T2SPO: TRAJECTORY-TO-STEP POLICY OPTIMIZATION FOR AGENTIC REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning enables large language model (LLM) agents to learn multi-step behaviors through interaction with their environments. However, rewards in many interactive tasks reflect only the final outcome, providing limited guidance on which intermediate decisions advance the task. Successful training trajectories contain intermediate states that can provide supervision for subsequent interactions. We introduce **Trajectory-to-Step Policy Optimization (T2SPO)**, a method that uses past interaction trajectories to provide step-level feedback for policy learning. T2SPO derives remaining-distance targets from successful trajectories and pairs them with representations of the states visited along the way. Conditioned on these examples, a pretrained TabPFN regressor estimates the remaining distance to success at each state of a new rollout. Changes in this distance estimate across consecutive states yield auxiliary credit for agent steps alongside task-level supervision. As training proceeds, newly completed trajectories refresh the estimator's context, incorporating new experience without updating its parameters. Experiments with 1.5B and 7B language models on ALFWorld and WebShop show that T2SPO consistently improves overall task success over GRPO.

1 INTRODUCTION

Reinforcement learning enables LLM agents to improve through environment interaction (Zhou et al., 2024; Wang et al., 2025). In multi-step tasks, the final outcome evaluates the whole trajectory but does not reveal how individual decisions contribute to success. **Estimating task progress and its changes across steps** helps connect individual decisions to task outcomes, providing fine-grained signals for credit assignment during policy optimization (Zhou et al., 2024; Li et al., 2026).

PPO uses a learned critic to estimate expected returns from intermediate states (Schulman et al., 2017). Combined with generalized advantage estimation, these predictions provide state-dependent training signals for individual actions (Schulman et al., 2016). This requires training a value model alongside the policy. GRPO avoids training a critic by estimating its baseline from the rewards of trajectories sampled for the same task (Shao et al., 2024). Under outcome-only supervision, all steps within a trajectory share the same outcome advantage, even when some decisions advance the task and others are redundant (Feng et al., 2025). If all trajectories in a group receive the same outcome reward, these advantages are zero. This motivates our central question: **Can historical experience supply fine-grained step credit without training an additional value model?**

Successful trajectories collected during policy training provide supervision beyond their final outcomes: each visited state can be paired with the number of turns remaining along its own successful continuation. These observed lengths serve as a proxy for task progress, providing supervision for estimating remaining distance at states encountered in later rollouts. Changes in the predicted distance across consecutive states can then supply auxiliary credit for individual agent steps.

In this paper, we introduce **Trajectory-to-Step Policy Optimization (T2SPO)**, which implements this approach with a frozen TabPFN regressor (Müller et al., 2022; Hollmann et al., 2025). For each rollout batch, the regressor predicts remaining distance using a shared context of historical state representations and completion-length labels. Adjacent-state predictions define a potential difference that provides auxiliary step credit (Ng et al., 1999). Newly completed trajectories update the context after scoring, allowing subsequent estimates to incorporate new experience while the

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.

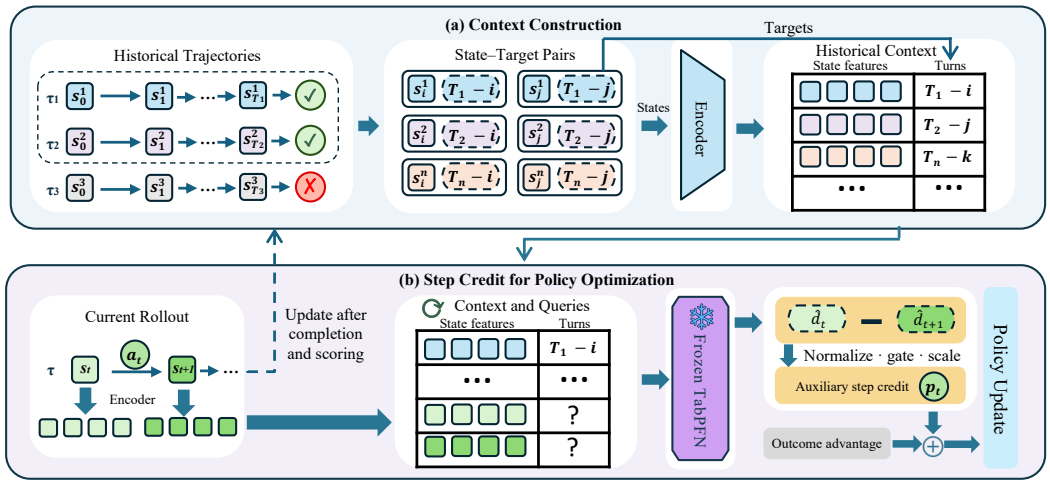


Figure 1: **T2SPO converts historical trajectories into step-level credit. (a) Context construction.** Successful historical trajectories pair each visited state with its own observed number of turns to completion (Section 4.1). State representations and these targets form a labeled context. **(b) Step credit for policy optimization.** A frozen TabPFN predicts the remaining distance at adjacent states of a current rollout using the same historical context. Their difference is normalized, gated, and scaled into auxiliary step credit, which augments the outcome advantage in GRPO. Current trajectories refresh the context only after completion and scoring.

encoder and regressor remain frozen. The credit augments outcome-based advantages in our main implementation and can also enter an actor–critic optimizer as an auxiliary reward (Figure 1).

Our contributions are:

- We propose **T2SPO**, a framework that reuses historical successful trajectories to supply auxiliary credit for individual agent steps while retaining task-level supervision.
- We develop a progress estimator with a frozen TabPFN regressor and online context updates, yielding auxiliary step credit without training an additional scoring model.
- Experiments on ALFWorld and WebShop show higher overall success than the reported GRPO baselines at both scales (Shridhar et al., 2021; Yao et al., 2022). Offline evaluations assess distance prediction across context sizes and SVD dimensions.

2 Related Work

Reinforcement learning for LLM agents. Reinforcement learning is widely used in language-model post-training, with feedback drawn from human preferences or verifiable task outcomes (Ouyang et al., 2022; DeepSeek-AI et al., 2025; Yu et al., 2025). For LLM agents, RL optimizes policies over multiple rounds of interaction, where earlier actions influence later decisions and eventual task outcomes (Zhou et al., 2024; Wang et al., 2025; Jin et al., 2025). Sparse or delayed feedback makes it difficult to determine which actions contribute to success, motivating methods that connect task outcomes to individual interaction steps (Feng et al., 2025).

Step-level feedback and credit assignment. Step-level feedback can be obtained by training a model to evaluate intermediate states or actions. For mathematical reasoning, process reward models learn from human step annotations or automatically generated supervision: Let's Verify Step by Step uses human judgments, while Math-Shepherd derives labels from sampled continuations (Lightman et al., 2024; Wang et al., 2024). For LLM agents, ArCHer and Turn-PPO learn critics that estimate expected returns at the turn level and support policy updates (Zhou et al., 2024; Li et al., 2026). Another approach compares returns following different actions from a shared state. GiGPO iden-

tifies repeated environment states in collected trajectories and groups the corresponding actions to compute relative advantages (Feng et al., 2025). ARPO instead uses entropy to guide branching during rollout collection and assigns advantages according to the shared and branched portions of the resulting trajectories (Dong et al., 2026a). Neither method trains an auxiliary scoring model.

Tabular in-context learning for value estimation. Prior-data fitted networks learn to make supervised predictions from labeled context examples by pretraining over synthetic tasks (Müller et al., 2022). Tabular foundation models extend this approach to classification and regression, with TabPFN emphasizing prediction from small datasets and TabICL scaling in-context classification to larger tables (Hollmann et al., 2023; 2025; Qu et al., 2025). Closely related applications are V_0 and $V_{0.5}$, which combine semantic representations with TabPFN-based prediction conditioned on prior instruction–performance pairs (Zhang et al., 2026b;a). V_0 predicts success probability at the initial prompt to allocate rollout budgets and route instructions, while $V_{0.5}$ combines such predictions with sparse on-policy outcomes for baseline estimation and adaptive sampling. T2SPO instead conditions a frozen TabPFN regressor on representations of intermediate states paired with the remaining lengths of their successful source trajectories. Differences in predicted remaining distance between adjacent states supply auxiliary step credit for policy optimization.

3 PRELIMINARIES

We first define the interaction and optimization setting, then introduce the regression interface used to transfer trajectory-derived supervision to current states.

3.1 MULTI-TURN REINFORCEMENT LEARNING

LLM agents interact with their environments by selecting actions conditioned on observations and prior context (Yao et al., 2023). We use *step*, or *turn*, to denote one complete agent–environment interaction. Given an instruction q, let s_t denote the context available to the policy before turn t, typically consisting of the instruction, the current observation o_t, and processed interaction history (Dong et al., 2026b). The policy $\pi_\theta(a_t \mid s_t)$ generates a textual action a_t autoregressively, after which the environment returns reward r_t and observation o_{t+1}. A trajectory τ comprises T such transitions, with sparse task feedback provided by the outcome at the end of the episode.

Advantage estimation. Outcome-supervised GRPO samples a group of $G > 1$ trajectories for the same task and normalizes their final task scores R_i (Shao et al., 2024):

$$\hat{A}_i^{\text{out}} = \frac{R_i - \overline{R}}{\sigma_R + \varepsilon}, \tag{1}$$

where $\overline{R}$ and σ_R denote the group mean and standard deviation, and $\varepsilon > 0$ ensures numerical stability. This estimator assigns the same advantage to all generated decisions in trajectory i without a learned critic. For example, an above-average outcome reinforces both a redundant search and the successful purchase that follows it. When all group outcomes are equal, the outcome advantages are zero even if the trajectories differ in intermediate progress or redundant behavior.

Learned critics. PPO can obtain advantages that vary across turns through a learned value function and generalized advantage estimation (GAE) (Schulman et al., 2017; 2016). At a nonterminal turn with zero environment reward, its temporal-difference residual is $\delta_t = \gamma V_\psi(s_{t+1}) - V_\psi(s_t)$, where $\gamma \in (0, 1]$ is the discount factor. These values let task outcomes inform updates to earlier decisions.

Policy updates. We use the clipped surrogate (Schulman et al., 2017; Shao et al., 2024)

$$\ell_{\text{clip}}(\rho, \hat{A}) = \min\{\rho\hat{A}, \text{clip}_{[1-\epsilon,\,1+\epsilon]}(\rho)\hat{A}\}, \tag{2}$$

where $\text{clip}_{[l,u]}$ clips its argument to $[l, u]$, $\epsilon > 0$ is the clipping threshold, and ρ is the likelihood ratio between the current policy and the fixed rollout policy π_{old}. For GRPO, ratios are computed over generated tokens; environment observations only condition the policy. Section 4.4 describes the integration of auxiliary step credit. Appendix A.5 details the PPO extension, including its critic targets, GAE, and likelihood ratios for complete turns.

3.2 Prior-Data Fitted Networks

A prior-data fitted network (PFN) predicts the target for a query x from a labeled context $C = \{(x_j, y_j)\}_{j=1}^{m}$ (Müller et al., 2022). For regression, $x_j, x \in \mathbb{R}^d$ and $y_j \in \mathbb{R}$. PFNs are pretrained on synthetic supervised learning tasks to approximate posterior predictive inference under a task prior. At deployment, a pretrained regressor evaluates

$$\hat{y} = f_\phi(x; C), \tag{3}$$

with fixed parameters ϕ; adapting the context changes the prediction without optimizing those parameters. TabPFN implements this approach for tabular classification and regression (Hollmann et al., 2023; 2025). T2SPO uses this prediction interface with interaction states as queries and historical states paired with trajectory-derived labels as context. Section 4 specifies how to construct those labels and convert the resulting state predictions into auxiliary signals for policy optimization.

4 Trajectory-to-Step Policy Optimization

T2SPO pairs historical states with labels derived from successful trajectories, uses these examples to predict remaining distance in subsequent rollouts, and converts adjacent-state differences into auxiliary step credit (Figure 1). This section describes the target construction, context-based prediction, and integration with policy optimization.

4.1 Supervision from Historical Trajectories

A completed successful trajectory provides more than a terminal score: each visited state has an observed number of turns before success. For each successful trajectory τ_j of length T_j, we pair its decision state $s_{j,t}$ with the number of interactions remaining along that trajectory:

$$y_{j,t} = T_j - t, \qquad t = 0, \ldots, T_j - 1. \tag{4}$$

Each label records the remaining length of its own successful trajectory. These targets provide a success-conditioned distance proxy reflecting continuations observed under the collection policies.

Unsuccessful trajectories do not reveal a completion distance. The success-only variant excludes their states from regression. A failure-augmented variant assigns them the rollout horizon H as a heuristic target and caps their share of the context. This target is not an observed time to success; the main experiments use success-only context (Appendix A.1).

4.2 Estimating Current States from Past Trajectories

State representation. To use historical targets at newly visited states, we express both reference and query states in a shared feature space. Let h_t denote the instruction and interaction record available before turn t, including the current observation. A frozen text encoder embeds the instruction, current observation, and the selected history from h_t. The estimator's history window is configured separately from the policy context s_t (Appendix A.2). We concatenate numerical step and collection-round features, then apply standardization and truncated singular value decomposition to obtain $x_t \in \mathbb{R}^d$. Both transformations are fitted on the selected historical examples and shared by reference and query states. The current action a_t, future observations, and the episode outcome are excluded from x_t. This representation allows the estimator to compare states through their features without requiring exact state matches across trajectories.

Shared trajectory context. At training round n, a bounded historical memory $\mathcal{M}_n$ supplies the labeled context $C_n = \{(x_{j,t}, y_{j,t})\}_{(j,t)\in\mathcal{I}_n}$. Here $\mathcal{I}_n$ indexes the selected historical trajectory–turn pairs, with $|\mathcal{I}_n| = M_n$. Successful state examples are sampled from completed batches and retained in a finite pool, with older entries evicted at capacity. The failure-augmented variant additionally samples unsuccessful states subject to its quota. A pretrained TabPFN regressor f_ϕ conditions on this context to predict

$$\hat{d}_{n,t} = \mathrm{clip}_{[0, D_{\max}]}\big(f_\phi(x_t; C_n)\big), \tag{5}$$

where $D_{\max}$ bounds the numerical output. All query states in a batch share C_n and its fitted feature transformation, including the adjacent states used to form each step signal. Context updates refresh the labeled examples and their feature transformation, changing predictions without updating ϕ (Müller et al., 2022; Hollmann et al., 2025).

Prediction before insertion. We score all current rollouts using C_n before their outcomes can supply labels for C_{n+1}. This ordering lets a trajectory inform future updates while excluding its own outcome from the context used to evaluate it. The main experiments initialize the context from a separate trajectory archive (Appendix A.2). If memory is initially empty, training uses the base optimization signal until sufficient historical support is available.

4.3 From State Estimates to Step Credit

A remaining-distance estimate describes a state, whereas policy optimization requires a signal for the decision that changes it. T2SPO compares estimates before and after each transition under the same historical context. Define $\Phi_n(h_t) = -\hat{d}_{n,t}$ and omit n below when unambiguous. Using a potential difference (Ng et al., 1999), we obtain

$$F_t = \gamma \Phi_n(h_{t+1}) - \Phi_n(h_t) = \hat{d}_t - \gamma \hat{d}_{t+1}. \tag{6}$$

Here $\gamma \in (0, 1]$ is the discount factor. When $\gamma = 1$, a positive difference indicates a reduction in predicted remaining distance. Successful termination sets the successor potential to zero. Unsuccessful termination receives zero final process signal, avoiding credit caused solely by ending an unsuccessful episode. At truncation, the PPO extension evaluates the actual successor state; the GRPO integration sets the final process signal to zero.

To stabilize the auxiliary scale, let $z_t = \operatorname{arsinh}(F_t)$ and let $\mathcal{B}_n$ contain the batch turns selected by the normalization mask. We compute the root-mean-square scale and normalized signal as

$$\nu_n^2 = \frac{1}{|\mathcal{B}_n|} \sum_{u \in \mathcal{B}_n} z_u^2, \qquad \tilde{F}_t = \operatorname{clip}_{[-c,c]} \left(\frac{z_t}{\nu_n + \varepsilon} \right). \tag{7}$$

Here $c > 0$ limits individual contributions and $\varepsilon > 0$ ensures numerical stability. The step credit supplied to the optimizer is

$$p_{i,t} = w_n g_{i,t} \tilde{F}_{i,t}, \tag{8}$$

where w_n ramps up after burn-in and $g_{i,t} \in [0, 1]$ controls whether a prediction contributes to the update. With sufficient support, valid predictions have gate one; insufficient support or rejected predictions disable the contribution. The ALFWorld configuration also suppresses unchanged observations and excludes them from normalization. Turns excluded by the normalization mask receive zero credit, as does the whole batch if $\mathcal{B}_n$ is empty.

Normalization, gating, and terminal exceptions modify the potential difference, so the standard policy-invariance guarantee does not extend to the full T2SPO procedure. Algorithm 1 summarizes step-credit computation for the main success-only configuration.

4.4 Policy Optimization with Step Credit

The auxiliary term adds local information to the task-level learning signal. We retain the base optimizer and specify how the same step credit enters its training targets.

Group-relative optimization. Let $\hat{A}_{i,t}^{\text{base}}$ be the outcome-based advantage supplied by the GRPO implementation (Shao et al., 2024). T2SPO augments it at each turn:

$$\hat{A}_{i,t}^{\text{T2SPO}} = \hat{A}_{i,t}^{\text{base}} + p_{i,t}. \tag{9}$$

The resulting scalar is assigned to the generated tokens of that turn and used in the token-level clipped surrogate (Equation 2). The outcome component preserves task-level supervision, while $p_{i,t}$ introduces variation among decisions within each trajectory. We retain the base implementation's group normalization and KL regularization and introduce no learned critic. Appendix A.4 details the multi-turn aggregation convention.

Actor–critic extension. The same step credit can instead augment rewards before PPO advantage estimation:

$$r_{i,t}^{+} = r_{i,t}^{\text{env}} + p_{i,t}. \tag{10}$$

The environment reward includes task outcomes and any configured invalid-action penalty. PPO computes advantages from these augmented rewards using GAE and trains its critic on the corresponding return targets (Schulman et al., 2016; Li et al., 2026). This integration uses complete-turn likelihood ratios; Appendix A.5 specifies the advantages, bootstrap handling, and objective.

Algorithm 1 Computing Auxiliary Step Credit

Require: Memory $\mathcal{M}_n$, rollouts $\mathcal{T}_n$, frozen encoder and regressor f_ϕ, scale w_n
Ensure: Step credits $\{p_{i,t}\}$ and updated memory $\mathcal{M}_{n+1}$

```
Initialize p_{i,t} ← 0 for all turns in T_n
Select successful historical support states from M_n
Assign each support state its source trajectory's remaining length (Equation 4)
if sufficient support is available then
    Encode support and query states; append step and collection-round features
    Fit the feature transformation on support; apply it to both sets
    Form the labeled context C_n from transformed support states
    Predict current and successor distances with the same C_n (Equation 5)
    Set successor distance to zero at successful termination
    Compute F_{i,t} ← d̂_{i,t} − γ d̂_{i,t+1}
    Apply terminal and truncation rules from Section 4.3
    Determine gates g_{i,t} and normalization set B_n (Section 4.3)
    if B_n ≠ ∅ then
        z_{i,t} ← arsinh(F_{i,t}) for (i,t) ∈ B_n
        ν_n ← sqrt(|B_n|^{-1} Σ_{(i,t)∈B_n} z_{i,t}^2)
        p_{i,t} ← w_n g_{i,t} clip_{[−c,c]}(z_{i,t} / (ν_n + ε)) for (i,t) ∈ B_n
    end if
end if
Update memory with sampled successful states from T_n, obtaining M_{n+1}
return {p_{i,t}}, M_{n+1}
```

The two integrations use the same source of step credit but apply it at different points: GRPO augments advantages directly, whereas PPO propagates augmented rewards through GAE. Both keep the current batch's targets fixed during optimization and incorporate completed trajectories into subsequent contexts. The encoder and TabPFN receive no gradient updates.

5 Experiments

5.1 Experimental Setup

Benchmarks and metrics. We evaluate on ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022), which require agents to complete tasks through multiple rounds of environment interaction. For ALFWorld, we report success rates for six task categories—Pick, Look, Clean, Heat, Cool, and Pick2—and overall success. For WebShop, we report the average task score on a 0–100 scale and the success rate. All success rates are expressed as percentages.

Models and baselines. We use Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct as policy models. Our primary comparison is with GRPO (Shao et al., 2024), whose outcome-based training signal is augmented by T2SPO. We also include the prompting-only base model, ReAct (Yao et al., 2023), PPO, RLOO, and GiGPO with standard-deviation normalization (Feng et al., 2025). All baseline results are taken from Feng et al. (2025).

Training details. The main T2SPO-GRPO configuration uses a context budget of 256 state examples drawn from historical successful trajectories. We evaluate checkpoints after 150 training updates. Training, estimator, and context settings are detailed in Appendix A.

5.2 Main Results

Comparison with baselines. Table 1 compares T2SPO-GRPO with published baseline results. Overall ALFWorld success is 74.7% for the 1.5B model and 83.6% for the 7B model, exceeding the reported GRPO means by 1.9 and 6.0 percentage points, respectively. On WebShop, the corresponding differences are 16.9 and 9.7 percentage points in success rate and 12.7 and 6.1 points in task score. T2SPO-GRPO also achieves higher WebShop scores and success rates than GiGPO at both model sizes, while GiGPO retains higher overall ALFWorld success.

Table 1: **Main results on ALFWorld and WebShop.** ALFWorld reports success rates (%); WebShop reports score (0–100) and success rate (%). RL results are means over three seeds; subscripts denote standard deviations. Shaded rows denote T2SPO-GRPO using only successful context examples; bold numbers indicate the highest reported mean within each model size. Δ reports the difference between T2SPO-GRPO and the GRPO reference within each model size: percentage points for success rates and points for WebShop score.

Type	Method	ALFWorld							WebShop	
		Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
Qwen2.5-1.5B-Instruct										
Prompting	Qwen2.5	5.9	5.5	3.3	9.7	4.2	0.0	4.1	23.1	5.2
Prompting	ReAct	17.4	20.5	15.7	6.2	7.7	2.0	12.8	40.1	11.3
RL Training	PPO (with critic)	$64.8_{\pm 3.5}$	$40.5_{\pm 6.9}$	$57.1_{\pm 4.9}$	$60.6_{\pm 6.6}$	$46.4_{\pm 4.0}$	$47.4_{\pm 1.9}$	$54.4_{\pm 3.1}$	$73.8_{\pm 3.0}$	$51.5_{\pm 2.9}$
RL Training	RLOO	$88.3_{\pm 3.0}$	$52.8_{\pm 8.6}$	$71.0_{\pm 5.9}$	$62.8_{\pm 8.7}$	$66.4_{\pm 5.5}$	$56.9_{\pm 4.7}$	$69.7_{\pm 2.5}$	$73.9_{\pm 5.6}$	$52.1_{\pm 6.7}$
RL Training	GRPO	$85.3_{\pm 1.5}$	$53.7_{\pm 8.0}$	$84.5_{\pm 6.8}$	$78.2_{\pm 7.9}$	$59.7_{\pm 5.0}$	$53.5_{\pm 5.6}$	$72.8_{\pm 3.6}$	$75.8_{\pm 3.5}$	$56.8_{\pm 3.8}$
RL Training	GiGPO	$\mathbf{94.4}_{\pm 5.9}$	$\mathbf{67.5}_{\pm 4.6}$	$\mathbf{94.8}_{\pm 3.8}$	$\mathbf{94.4}_{\pm 7.8}$	$79.8_{\pm 4.7}$	$\mathbf{76.4}_{\pm 5.4}$	$\mathbf{86.7}_{\pm 1.7}$	$83.1_{\pm 1.6}$	$65.0_{\pm 3.2}$
RL Training	**T2SPO-GRPO**	$87.4_{\pm 7.4}$	$56.8_{\pm 15.9}$	$90.3_{\pm 10.1}$	$65.9_{\pm 18.2}$	$\mathbf{86.5}_{\pm 3.0}$	$43.7_{\pm 23.9}$	$74.7_{\pm 10.0}$	$\mathbf{88.5}_{\pm 0.3}$	$\mathbf{73.7}_{\pm 3.3}$
	Δ vs. GRPO	+2.1	+3.1	+5.8	−12.3	+26.8	−9.8	+1.9	+12.7	+16.9
Qwen2.5-7B-Instruct										
Prompting	Qwen2.5	33.4	21.6	19.3	6.9	2.8	3.2	14.8	26.4	7.8
Prompting	ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2	46.2	19.5
RL Training	PPO (with critic)	$92.3_{\pm 4.0}$	$64.0_{\pm 8.4}$	$92.5_{\pm 2.4}$	$\mathbf{89.5}_{\pm 7.0}$	$80.3_{\pm 2.0}$	$68.8_{\pm 8.3}$	$80.4_{\pm 2.7}$	$81.4_{\pm 3.1}$	$68.7_{\pm 5.1}$
RL Training	RLOO	$87.6_{\pm 4.3}$	$78.2_{\pm 8.3}$	$87.3_{\pm 5.8}$	$81.3_{\pm 7.6}$	$71.9_{\pm 5.2}$	$48.9_{\pm 8.4}$	$75.5_{\pm 4.6}$	$80.3_{\pm 3.2}$	$65.7_{\pm 4.0}$
RL Training	GRPO	$90.8_{\pm 5.1}$	$66.1_{\pm 6.7}$	$89.3_{\pm 5.4}$	$74.7_{\pm 6.9}$	$72.5_{\pm 5.4}$	$64.7_{\pm 7.3}$	$77.6_{\pm 5.2}$	$79.3_{\pm 2.8}$	$66.1_{\pm 3.7}$
RL Training	GiGPO	$\mathbf{97.7}_{\pm 1.6}$	$\mathbf{82.7}_{\pm 7.9}$	$\mathbf{98.8}_{\pm 1.6}$	$83.7_{\pm 7.2}$	$\mathbf{89.3}_{\pm 8.2}$	$\mathbf{79.2}_{\pm 6.6}$	$\mathbf{90.8}_{\pm 1.3}$	$84.4_{\pm 2.9}$	$72.8_{\pm 3.2}$
RL Training	**T2SPO-GRPO**	$91.3_{\pm 5.7}$	$73.8_{\pm 2.1}$	$97.3_{\pm 2.4}$	$84.5_{\pm 16.8}$	$78.4_{\pm 7.7}$	$63.5_{\pm 10.5}$	$83.6_{\pm 6.4}$	$\mathbf{85.4}_{\pm 4.1}$	$\mathbf{75.8}_{\pm 3.6}$
	Δ vs. GRPO	+0.5	+7.7	+8.0	+9.8	+5.9	−1.2	+6.0	+6.1	+9.7

Qwen2.5, ReAct, PPO, RLOO, GRPO, and GiGPO results are from Feng et al. (2025).

Table 2: **Failure-augmented context on ALFWorld and WebShop.** Means over three seeds; subscripts report standard deviations. Bold numbers indicate the higher mean within each model-size pair.

		ALFWorld	WebShop	
Model	Context examples	Succ. (%)	Score	Succ. (%)
Qwen2.5-1.5B-Instruct	Success only	$\mathbf{74.7}_{\pm 10.0}$	$\mathbf{88.5}_{\pm 0.3}$	$\mathbf{73.7}_{\pm 3.3}$
	With 25% failure examples	$74.0_{\pm 5.9}$	$85.0_{\pm 0.6}$	$68.2_{\pm 3.0}$
Qwen2.5-7B-Instruct	Success only	$83.6_{\pm 6.4}$	$85.4_{\pm 4.1}$	$\mathbf{75.8}_{\pm 3.6}$
	With 25% failure examples	$\mathbf{89.3}_{\pm 3.0}$	$\mathbf{86.3}_{\pm 0.6}$	$74.7_{\pm 6.1}$

Failure-augmented context. Table 2 compares success-only context with a variant that admits up to 25% failure examples, labeled by the rollout horizon. Both configurations use a context budget of 256 and a process weight of 0.10. On ALFWorld, failure augmentation raises mean success from 83.6% to 89.3% at 7B, while the 1.5B mean remains similar at 74.0% versus 74.7%. On WebShop, mean success is lower at both model sizes; task score decreases at 1.5B but increases slightly at 7B. The effect of failure examples therefore varies across benchmarks and model sizes. Appendix A.3 reports the ALFWorld task-category breakdown.

5.3 Offline Distance Prediction

We evaluate remaining-distance prediction on 1,951 states from successful WebShop trajectories generated by Qwen2.5-1.5B-Instruct. Context and query trajectories have disjoint task identities, and both use the per-trajectory targets in Equation 4. Mean absolute error (MAE) is averaged over query states and measured in turns. Inputs contain the task description, current observation, history, step index, and pre-rollout metadata, excluding the current action and future information.

Using the same queries, we test all nine combinations of context budgets 128, 256, and 512 and SVD dimensions 32, 64, and 128. Larger contexts retain all examples from smaller contexts.

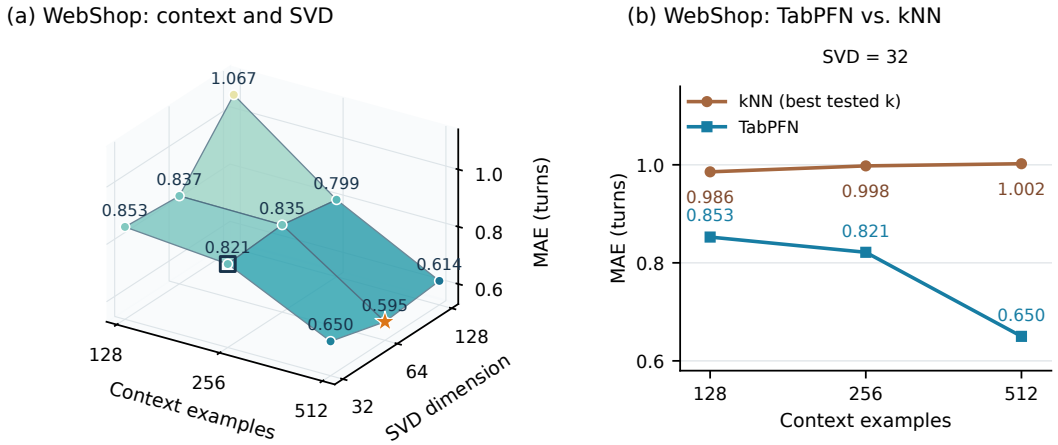


Figure 2: **Offline distance prediction on WebShop.** (a) Remaining-distance MAE across nine context/SVD settings; parameter axes use logarithmic spacing, and facets connect evaluated points. The square marks the main RL configuration; the star marks the lowest observed MAE. (b) TabPFN and the lowest kNN MAE among five tested neighbor counts at each context budget, using the same context examples, SVD32 features, and query states.

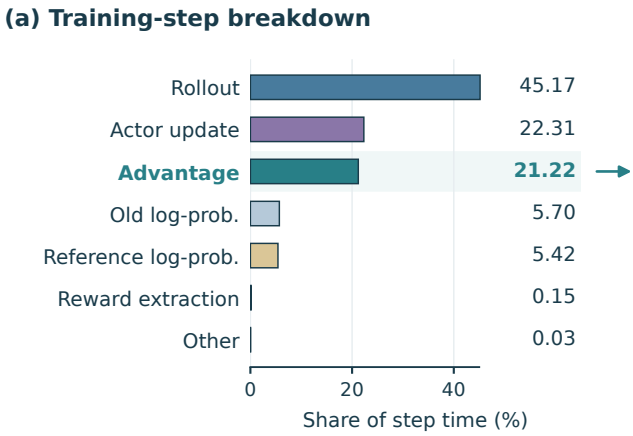
Increasing the context budget from the main RL setting of 256 to 512 reduces offline MAE at every tested SVD dimension (Figure 2a). At SVD32, MAE decreases from 0.8215 to 0.6500 turns, a 20.9% reduction. The lowest observed MAE is 0.5948 with 512 context examples and SVD64, while increasing feature dimension does not improve prediction uniformly (Appendix B).

5.4 Comparison with Nearest Neighbors

We compare TabPFN with kNN regression at SVD32 across context budgets $M \in \{128, 256, 512\}$. At each budget, both estimators use the same labeled context, fitted feature transformation, and query states. The kNN baseline averages labels of the nearest context states under cosine similarity. We test $k \in \{8, 16, M/4, M/2, M\}$ and plot the lowest observed kNN MAE at each budget (Figure 2b). TabPFN's MAE decreases from 0.8526 to 0.6500, while the best tested kNN MAEs are 0.9856, 0.9978, and 1.0022. The corresponding MAE reductions are 13.5%, 17.7%, and 35.1%, respectively. Appendix B reports the full grid, tested neighbor counts, and offline evaluation protocol.

5.5 Computation Cost

We profile three WebShop Qwen2.5-1.5B-Instruct runs with the main context-256/SVD32 configuration. Across 360 optimizer steps excluding evaluation and checkpointing, mean step time is 380.21 s. Figure 3 shows that rollout generation, actor updates, and advantage computation account for 45.17%, 22.31%, and 21.22% of summed step time, respectively. Within advantage computation, state estimation accounts for 21.06% of step time, while preparing the next context takes 0.075%. State estimation includes embedding, context construction, SVD, and TabPFN fitting and prediction. Appendix C details the timing protocol and per-stage measurements.



(b) Advantage stage

Component	Mean (s)	Share (%)
State estimation	**80.084**	**21.063**
Context prep.	0.285	0.075
Other adv. work	0.314	0.083
Total	**80.682**	**21.221**

Figure 3: **Online computation cost on WebShop.** (a) Mutually exclusive training stages across three seeds and 360 ordinary optimizer steps. (b) Substages within advantage computation; times are means in seconds. Both panels report shares of summed full-step time. Other advantage work includes GRPO advantage computation, shaping, and context commit.

6 Conclusion

T2SPO converts historical completion lengths into auxiliary step credit through a frozen distance estimator whose context evolves with training. Adjacent-state differences augment GRPO advantages or PPO rewards, with PPO retaining its learned critic. T2SPO-GRPO achieves higher overall ALFWorld success and higher WebShop scores and success rates than the reported GRPO baselines at both model sizes. In the offline WebShop evaluation, larger contexts reduce prediction error against observed remaining lengths. At SVD32, TabPFN achieves lower MAE than the best tested kNN settings at all three context budgets.

Limitations and future work. The mixed effects of a fixed proportion of failure examples motivate supervision that adapts to the task and the agent's experience. For tasks with irreversible dead ends, such as Sokoban, the estimator should distinguish recoverable detours from states where success is no longer possible. Historical context could evolve through soft updates based on weighted combinations of feature vectors and remaining-distance targets from semantically compatible states. When successful trajectories are scarce, memory updates and the strength of auxiliary credit could adapt to the coverage and reliability of available experience. For more complex tasks, training or fine-tuning the compact progress predictor could improve task-specific progress estimation while keeping the trainable model substantially smaller than a separate LLM-based critic.

AI Use Statement

AI tools assisted with manuscript editing, literature search, figure preparation, and the implementation of parts of the code. They also supported discussions of experimental design and result interpretation. The authors developed the core research idea, specified the algorithmic procedure, and take responsibility for the final manuscript, references, code, and reported results.

REFERENCES

DeepSeek-AI et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. arXiv preprint arXiv:2501.12948, 2025. URL https://arxiv.org/abs/2501.12948v1.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization. In *International Conference on Learning Representations*, 2026a.

Guanting Dong, Xiaoshuai Song, Yuyang Hu, Jiajie Jin, Chenghao Zhang, Yifei Chen, Xiaoxi Li, Huaying Yuan, Xinyu Yang, Tongyu Wen, Jiejun Tan, Hongjin Qian, Shijue Huang, Junting Lu, Zhenyu Li, Wanjun Zhong, Yutao Zhu, Tat-Seng Chua, Zhicheng Dou, and Ji-Rong Wen. Towards long-horizon agents: A survey. *Preprints*, July 2026b. doi: 10.20944/preprints202607.1328.v1.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, volume 38, pp. 46375–46408. Curran Associates, Inc., 2025. doi: 10.52202/085713-1544.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025. doi: 10.1038/s41586-024-08328-6.

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

Junbo Li, Peng Zhou, Rui Meng, Meet P. Vadera, Lihong Li, and Yang Li. Turn-PPO: Turn-level advantage estimation with PPO for improved multi-turn RL in agentic LLMs. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 6227–6243. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-eacl.328.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do Bayesian inference. In *International Conference on Learning Representations*, 2022.

Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, pp. 278–287, 1999.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/068431-2011.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 50817–50847. PMLR, 2025.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017. URL https://arxiv.org/abs/1707.06347.

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

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

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.510.

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

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pp. 20744–20757. Curran Associates, Inc., 2022. doi: 10.52202/068431-1508.

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

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, pp. 113222–113244. Curran Associates, Inc., 2025. doi: 10.52202/085713-3775.

Yi-Kai Zhang, Yueqing Sun, Hongyan Hao, Qi Gu, Xunliang Cai, Long Chen, De-Chuan Zhan, and Han-Jia Ye. $V_{0.5}$: Generalist value model as a prior for sparse RL rollouts. In *Third Conference on Language Modeling*, 2026a.

Yi-Kai Zhang, Zhiyuan Yao, Hongyan Hao, Yueqing Sun, Qi Gu, Hui Su, Xunliang Cai, De-Chuan Zhan, and Han-Jia Ye. V_0: A generalist value model for any policy at state zero. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62178–62209. PMLR, 2024.

A T2SPO Implementation and Evaluation Details

This appendix details policy training, evaluation, context construction, and optimizer integration.

A.1 Training and Evaluation Protocol

Table 3 summarizes the main T2SPO-GRPO training settings for the Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct models. Each update collects eight trajectories for each of 16 task instances, giving 128 rollouts. The policy receives the instruction, current observation, available actions, and up to two preceding observation–action pairs. The estimator's history window is configured separately, as detailed in Appendix A.2.

Table 3: Policy training and evaluation settings shared by both model sizes.

Setting	ALFWorld	WebShop
Task instances per update	16	16
Rollouts per task instance	8	8
Actor learning rate	10^{-6}	10^{-6}
KL loss coefficient	0.01	0.01
Invalid-action penalty	0.10	0.10
Policy history window	2 turns	2 turns
Maximum prompt length (tokens)	2,048	4,096
Maximum response length (tokens)	512	512
Rollout horizon (turns)	50	15
Training updates	150	150
Evaluation interval (updates)	5	5
Episodes per evaluation	128	128
Evaluation temperature	0.4	0.4

ALFWorld uses the environment's training and in-distribution evaluation splits. WebShop reserves the first 500 instruction indices for evaluation and draws training tasks from the remaining indices. Each WebShop evaluation samples 128 instructions without replacement from the reserved set. Evaluation uses stochastic decoding at temperature 0.4, and reported results use the checkpoint after update 150. For both the success-only and failure-augmented T2SPO-GRPO results, we report the mean and sample standard deviation over seeds 0, 1, and 2. Baseline results in Table 1 are taken from Feng et al. (2025).

A.2 Estimator and Context Configurations

Table 4 specifies the success-only T2SPO-GRPO configurations in Table 1. All use a frozen Qwen3-Embedding-0.6B encoder and the official TabPFN V2 regressor with one ensemble member and a random seed matched to the training run. Both ALFWorld model sizes use a structured state summary without an additional action suffix. WebShop uses raw observations at both model sizes. The estimator includes the full available interaction history for both benchmarks. Step and collection-round features are added before context-fitted standardization and SVD.

Context initialization and refresh. Each benchmark starts with a fixed archive of 128 training trajectories collected from a separate policy checkpoint at update 100. ALFWorld uses a separate archive for each model size, while WebShop shares one frozen archive across both sizes. Each archive is shared across training seeds. Successful states from this archive fill the context while the online pool is small. Each completed training batch contributes at most 64 sampled successful states to a pool of capacity 256, with older entries evicted at capacity. Context selection takes online states first and fills any remaining positions from the archive, so archived examples are gradually replaced as online experience accumulates. Both sources use each state's own successful continuation length from Equation 4. The next batch shares this context and its fitted feature transformation.

Failure-augmented context. The variants in Table 2 keep the context budget at 256 and the process weight at 0.10 on both benchmarks. Unsuccessful states receive the horizon label $H = 50$ on ALFWorld and $H = 15$ on WebShop, and occupy at most 25% of the context; successful states fill

Table 4: Estimator settings for the main T2SPO-GRPO runs at both model sizes. History refers to the estimator input; context and successful-state pool budgets count state rows.

Shared settings	**Value**	
Estimator history window	Full prefix	
TabPFN package version	8.3.0	
Shared context budget	256	
Successful-state pool capacity	256	
New successful states per round	≤ 64	
Recent trajectory window	256	
Minimum support rows	8	
Process clip c	5	
Final process weight	0.10	
Burn-in / linear ramp (rounds)	1 / 4	
Discount γ	1.0	
Benchmark-specific settings	**ALFWorld**	**WebShop**
State text	Structured summary	Raw
SVD dimension	64	32
Suppress unchanged observations	Yes	No

the remaining positions. Each variant retains the corresponding success-only run's policy training and state representation settings.

A.3 Failure-Augmented Results by Task Category

Table 5 gives the complete ALFWorld category results for the success-only and failure-augmented configurations. At 7B, all six category means increase with failure examples; at 1.5B, the changes vary across categories. The overall ALFWorld and WebShop comparisons are reported in Table 2.

Table 5: **Failure-augmented context across ALFWorld task categories.** Success rates (%), averaged over three seeds; subscripts denote sample standard deviations. T2SPO-GRPO uses success-only context, while T2SPO-GRPO + Fail admits up to 25% failure examples. Bold numbers indicate the higher mean within each model-size pair.

Type	Method	ALFWorld						
		Pick	Look	Clean	Heat	Cool	Pick2	All
Qwen2.5-1.5B-Instruct								
RL Training	**T2SPO-GRPO**	**87.4**$_{\pm 7.4}$	56.8$_{\pm 15.9}$	**90.3**$_{\pm 10.1}$	65.9$_{\pm 18.2}$	**86.5**$_{\pm 3.0}$	43.7$_{\pm 23.9}$	**74.7**$_{\pm 10.0}$
RL Training	**T2SPO-GRPO + Fail**	81.6$_{\pm 4.6}$	**65.2**$_{\pm 17.0}$	82.9$_{\pm 13.1}$	**70.2**$_{\pm 11.4}$	77.9$_{\pm 2.6}$	**54.1**$_{\pm 21.2}$	74.0$_{\pm 5.9}$
Qwen2.5-7B-Instruct								
RL Training	**T2SPO-GRPO**	91.3$_{\pm 5.7}$	73.8$_{\pm 2.1}$	97.3$_{\pm 2.4}$	84.5$_{\pm 16.8}$	78.4$_{\pm 7.7}$	63.5$_{\pm 10.5}$	83.6$_{\pm 6.4}$
RL Training	**T2SPO-GRPO + Fail**	**94.1**$_{\pm 2.9}$	**75.0**$_{\pm 12.5}$	**98.7**$_{\pm 2.2}$	**87.3**$_{\pm 14.5}$	**83.3**$_{\pm 2.9}$	**82.5**$_{\pm 11.0}$	**89.3**$_{\pm 3.0}$

A.4 Algorithm and Reproducibility Details

Update order. For each training round, the policy first collects a rollout batch. The estimator predicts remaining distances for the batch using only the existing context, after which the step credits and policy-training targets are computed. The completed trajectories can then be inserted into memory and the next predictor context prepared. This insertion does not change the predictions or targets already computed for the current batch. The actor is optimized using these fixed targets; in the PPO extension the critic is updated as well. There are no gradient updates to the encoder or TabPFN.

GRPO aggregation. The base multi-turn implementation computes within-task outcome statistics over expanded turn rows. Consequently, a trajectory with more turns contributes more entries to the group mean and standard deviation. This convention differs from the standard estimator in

Equation 1, which contributes one score per trajectory. T2SPO preserves the base convention and adds its process term only after the base advantages have been computed.

Validity checks. The direct predictor requires sufficient support, finite input features, a valid projection, and finite predictions. Insufficient support disables the process contribution; rejected predictions receive zero gate in numerical fallback paths. Strict error handling can instead stop an update on an estimator error. For the ALFWorld configuration, unchanged observations set both the process gate and normalization mask to zero. If no turns remain active, the process signal is zero.

A.5 Actor–Critic Extension

We detail the PPO integration from Section 4.4; Appendix D reports its ALFWorld results.

Turn probabilities. A complete response $a_t = (a_{t,1}, \ldots, a_{t,L_t})$ has autoregressive probability

$$\pi_\theta(a_t \mid s_t) = \prod_{k=1}^{L_t} \pi_\theta(a_{t,k} \mid s_t, a_{t,<k}). \tag{11}$$

Environment observations condition the policy but are excluded from its action likelihood and loss.

Advantage estimation. PPO uses a learned critic $V_\psi(s_t)$ to approximate the expected discounted return from s_t. Generalized advantage estimation (GAE) forms discounted sums of temporal-difference residuals (Schulman et al., 2016); for a rollout segment of K turns (Li et al., 2026),

$$\begin{aligned} \delta_t &= r_t + \gamma V_\psi(s_{t+1}) - V_\psi(s_t), \\ \hat{A}_t^{\mathrm{GAE}} &= \sum_{\ell=0}^{K-t-1} (\gamma\lambda)^\ell \delta_{t+\ell}, \end{aligned} \tag{12}$$

where $\gamma \in (0, 1]$ is the discount factor and $\lambda \in [0, 1]$ controls the bias–variance tradeoff. At a true terminal state, the successor value is zero; for a rollout truncated before termination, the final residual instead uses a bootstrapped successor value. For T2SPO, the augmented rewards r^+ from Equation 10 replace r_t in these residuals to obtain $\hat{A}^+$. The critic targets are $\hat{V}_{i,t}^{\mathrm{target}} = \hat{A}_{i,t}^+ + V_\psi(s_{i,t})$, formed before advantage whitening. At truncation, the PFN also evaluates the actual successor state when constructing the auxiliary reward.

Policy updates. Following turn-level optimization (Li et al., 2026), define the likelihood ratio of the complete generated response and its clipped objective as

$$\begin{aligned} \rho_{i,t}(\theta) &= \frac{\pi_\theta(a_{i,t} \mid s_{i,t})}{\pi_{\mathrm{old}}(a_{i,t} \mid s_{i,t})}, \\ \mathcal{J}_{\mathrm{turn}}(\theta) &= \frac{1}{N_{\mathrm{tok}}} \sum_{i,t} \ell_{\mathrm{clip}}\big(\rho_{i,t}(\theta), \bar{A}_{i,t}^+\big). \end{aligned} \tag{13}$$

Here $\bar{A}^+$ is whitened over valid turns, and N_{tok} is the total number of valid generated tokens in the batch. Each turn contributes one clipped surrogate. KL regularization is retained from the base optimizer. The complete-turn log-ratio is the sum of generated-token log-ratios, clipped to $[-20, 20]$ before exponentiation for numerical stability.

B Offline Distance Prediction

Table 6 reports the WebShop context/SVD grid from Section 5.3. The query set contains 514 trajectories and 3,073 states, with regression metrics computed on its 1,951 successful-trajectory states. Context and query targets use each successful trajectory's own remaining length. Tasks are split between context and query sets with seed 20260812. A query-independent ordering of 2,115 successful context states (seed 20260920) defines nested pools of 128, 256, and 512.

All configurations encode raw observations with recent two-turn history using Qwen3-Embedding-0.6B and append the step and collection-round features. Standardization and SVD are fitted on the selected context and then applied to the query states. The regressor uses the official TabPFN V2 checkpoint, one ensemble member, and random seed 0. The same cached 1,024-dimensional embeddings and held-out query states are reused for every context budget and SVD dimension.

Table 6: **Offline remaining-distance prediction on WebShop.** All rows use TabPFN and evaluate the same 1,951 successful query states on held-out tasks. MAE and RMSE are measured in turns; ρ denotes Spearman correlation. Bold numbers mark the best value for each metric.

Context	SVD	MAE ↓	RMSE ↓	ρ ↑
128	32	0.8526	1.4784	0.7293
128	64	0.8370	1.4714	0.7500
128	128	1.0666	1.6504	0.7240
256	32	0.8215	1.4698	0.7486
256	64	0.8345	1.4681	0.7838
256	128	0.7987	1.4477	0.7815
512	32	0.6500	1.3799	0.8094
512	64	**0.5948**	1.3486	**0.8359**
512	128	0.6138	**1.3366**	0.8333

B.1 Nearest-Neighbor Comparison

Table 7 reports the full kNN comparison underlying Figure 2b. At each context budget, both methods use the same context examples, SVD32 feature transformation, and 1,951 successful query states. The kNN estimates are unweighted means of labels selected by cosine similarity in the transformed feature space. The figure uses the lowest observed kNN MAE among five tested neighbor counts at each budget: $k = 32$, 64, and 8 for context sizes 128, 256, and 512, respectively.

Table 7: **TabPFN and kNN on WebShop.** All results use SVD32; columns give context budgets M. MAE is in turns, with the lowest value in bold. At $k = M$, kNN returns the context mean.

Estimator	$M = 128$	$M = 256$	$M = 512$
kNN ($k = 8$)	1.1042	1.0804	1.0022
kNN ($k = 16$)	1.0429	1.0948	1.0401
kNN ($k = M/4$)	0.9856	0.9978	1.0093
kNN ($k = M/2$)	1.0874	1.0882	1.1083
kNN ($k = M$)	1.4719	1.4678	1.4669
TabPFN	**0.8526**	**0.8215**	**0.6500**

C COMPUTATION COST

We measure online computation in the three success-only WebShop Qwen2.5-1.5B-Instruct runs, each using two NVIDIA H20 GPUs, a context budget of 256, SVD32, and a 15-turn horizon. Each optimizer step collects 128 trajectories and queries 640–1,760 states, with a median of 928. We retain log records from the final training process of each run and exclude steps containing evaluation or checkpoint saving, leaving 120 steps per seed and 360 steps in total. Table 8 reports descriptive timing statistics over these steps. Each percentage is the sum of the component's times divided by the sum of full-step times.

Table 8: **Online training-step timing on WebShop.** Means and standard deviations are over 360 optimizer steps from three seeds; times are in seconds. Shares use summed full-step time. The lower block expands the advantage row in the upper block.

Component	Mean ± SD	Median	Share (%)
Rollout generation	171.729 ± 34.274	165.729	45.167
Reward extraction	0.560 ± 0.215	0.515	0.147
Old-policy log-probability	21.657 ± 6.640	19.245	5.696
Reference-policy log-probability	20.616 ± 6.416	18.093	5.422
Advantage computation	80.682 ± 13.747	79.454	21.221
Actor update	84.830 ± 28.138	73.509	22.312
Between-stage gaps	0.131 ± 0.094	0.119	0.034
Full training step	380.206 ± 61.672	363.305	100.000
Within advantage computation			
State estimation	80.084 ± 13.825	78.910	21.063
Online-context preparation	0.285 ± 0.223	0.237	0.075
Other advantage work	0.314 ± 0.134	0.281	0.083

Timing hierarchy. The six top-level stage timers are mutually exclusive and together cover 99.966% of recorded step time; the residual consists of between-stage work and timer overhead. State estimation and online-context preparation are nested within advantage computation. The state-estimation timer covers feature and embedding handling, prediction-context construction, SVD, TabPFN fitting, and prediction. Preparing context updates for subsequent batches is timed separately. Together, state estimation and context preparation account for 21.138% of step time. The remaining advantage time includes base GRPO computation, shaping, validation, and context commit. This runtime has no separate weight-synchronization timer, and GRPO uses no critic. These timings describe component shares within the measured runs, rather than a matched-control slowdown.

D Additional Turn-PPO Results

We evaluate the actor–critic integration in Appendix A.5 on ALFWorld with both model sizes. Our Turn-PPO control and T2SPO-Turn-PPO retain the same learned critic, GAE, and turn-level policy objective; T2SPO adds auxiliary rewards with weight 0.10. Each update collects one rollout for each of 128 task instances, with a horizon of 50 turns and 150 training updates. Both use actor and critic learning rates of 10^{-6} and 10^{-5}, respectively, discount $\gamma = 0.99$, GAE parameter $\lambda = 0.9$, one PPO epoch, and clipping ratio 0.2. The estimator uses raw state text with recent two-turn history and full admissible-action descriptions, SVD64, a context of 256 successful state examples with trajectory-local $T - t$ targets, and the official TabPFN V2 regressor from package version 2.2.1.

Table 9: **Turn-PPO with and without T2SPO on ALFWorld.** Success rates (%) are means with sample standard deviations over the same two training seeds, 0 and 1 ($n = 2$). Each final checkpoint is evaluated on 128 episodes after 150 updates; Δ is the change in mean success in percentage points.

Model	Turn-PPO	T2SPO-Turn-PPO	Δ
Qwen2.5-1.5B-Instruct	$91.4_{\pm 6.6}$	$91.4_{\pm 3.3}$	0.0
Qwen2.5-7B-Instruct	$90.2_{\pm 9.4}$	$91.4_{\pm 1.1}$	+1.2

Table 9 shows unchanged mean success at 1.5B and a 1.2-point increase at 7B. At each scale, one seed improves and the other declines.