What Makes Two States the Same? Value-Aware State Matching for Multi-Turn Agent Credit Assignment

Yangyi Fang
Tsinghua University
Beijing, China

Haolin Shi
Tsinghua University
Beijing, China

Abstract

Credit assignment for multi-turn LLM agents comes down to estimating how good each intermediate state is, and so-called critic-free methods still make such an estimate: the group mean reward of GRPO is a Monte Carlo estimate of the value of the initial state. Recent step-level methods extend this estimate to later states by finding steps in other rollouts that reach the "same" state and averaging their returns, which amounts to a value lookup table keyed on raw observation text that ignores whether textual differences matter for future returns. Under simple noise assumptions, we show that the error of such a step baseline splits into bias and variance: matching on value-irrelevant features creates unnecessarily small groups (high variance), while ignoring value-relevant features merges states that should be distinguished (high bias). Guided by this analysis, we propose Structured State Abstraction (SSA), a value estimator without a critic that decomposes observations into typed fields using environment-provided metadata and learns per-field importance weights from return variance. Instead of exact text matching, SSA computes a kernel-smoothed baseline in which every state contributes to each step in proportion to its structured similarity, with a temperature that moves between exact matching and the task-level mean. On ALFWorld and WebShop, SSA improves over GiGPO and HGPO at both model scales, and kernel smoothing gives a comparison set to every step, including the 30–36% of steps that exact matching leaves as singletons without step-level credit.

1 Introduction

LLM agents are increasingly asked to do ordinary work: update records across business applications, answer tickets, edit shared documents, and coordinate with coworkers through dozens of tool calls [Drouin et al., 2024, Trivedi et al., 2024, Yao et al., 2024, Xu et al., 2024, Patwardhan et al., 2025, Shepard and Salimans, 2026]. Training them with reinforcement learning from outcome rewards [Ouyang et al., 2022, Guo et al., 2025, Chen et al., 2025] raises a credit assignment problem: the step that sent the right email should receive more credit than the steps that only looked around. The standard tool is a value function: the credit of a step is its advantage, the return that followed it minus the value V^π of the state it started from, so good credit needs a good estimate of V^π. PPO [Schulman et al., 2017] learns V^π with a critic, which for LLM agents is a second large model trained alongside the policy. GRPO [Shao et al., 2024] drops the critic but not the value function: the mean reward of a group of rollouts from the same prompt is a Monte Carlo estimate of V^π at the initial state, and the whole episode shares one advantage, however far the agent has progressed.

Step-level methods without a critic carry this Monte Carlo estimate to later states [Feng et al., 2025, He et al., 2026, Li et al., 2026], while other work spends extra rollouts or organizes them into trees [Kazemnejad et al., 2025, Cao et al., 2026], add step-level rewards [Xi et al., 2025,

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

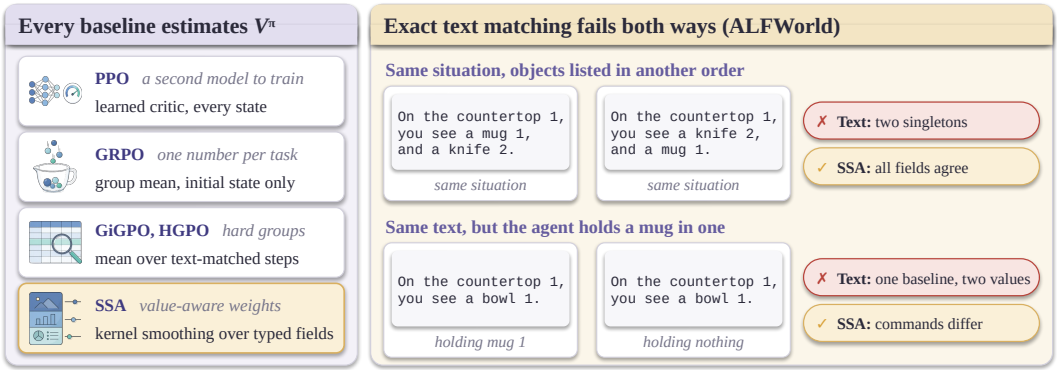

Figure 1: Left: the baseline of each method estimates V^π, and the methods differ in where they estimate it and which steps they average. Right: two pairs of ALFWorld observations. Exact text matching misses the first pair, which lists the same objects in another order, and merges the second, whose text is identical although the agent holds a mug in one. SSA compares typed fields, so the first pair shares a baseline and `admissible_commands` separates the second.

Zhang et al., 2026, Wang et al., 2026, Tan et al., 2026], or reshape turn-level signals [Wang et al., 2025a, Dong et al., 2025, 2026, Zeng et al., 2025, Li et al., 2025]. These critic-free baselines all estimate the same quantity and differ in where they estimate it: GRPO keeps one entry per task, and GiGPO [Feng et al., 2025], which averages the returns of steps that observe the same text, keeps a lookup table keyed on the observation string. Asking when two states are "the same" is asking what the table should be keyed on, a state abstraction problem [Li et al., 2006].

Exact text matching fails in two directions (Figure 1). Two observations can describe the same situation and differ only in surface text, such as the order in which objects are listed; on ALFWorld and WebShop, exact matching leaves 30–36% of steps in singleton groups with no step-level signal. Two observations can also be identical while the situations differ, for instance when the agent holds an object in one, and exact matching shares one estimate between states with different values. Treating the step baseline as an estimate of V^π makes this precise: under simple noise assumptions its mean squared error splits into a grouping bias, from merging states with different values, and an estimation variance, from averaging too few returns. Existing methods leave two choices open: which features the estimate depends on, and how sharply it separates states that differ in them.

Agent environments already answer much of the first question. Tool interfaces return structured responses, and ALFWorld and WebShop, the testbeds of prior step-level methods and of our experiments, return the admissible commands or available actions in their `info` dictionaries, yet existing step-level methods compare only the raw observation string. We propose SSA (Structured State Abstraction), a value estimator built from the rollouts that training already collects. SSA decomposes each observation into typed fields, learns per-field weights from return variance via ANOVA, and replaces exact matching with kernel smoothing over the resulting structured similarity. A temperature τ moves the estimate between exact matching ($\tau \to 0$) and the task-level mean over all steps ($\tau \to \infty$). The method needs no additional rollouts or models and little extra computation.

Our contributions:

- We view the baselines of critic-free agent RL, from the group mean of GRPO to step-level state matching, as Monte Carlo estimates of V^π that differ in which states they average over, and we derive a bias-variance decomposition of the step-baseline error, which identifies value-agnostic state comparison and hard grouping as two separate failure modes.
- We propose SSA, a value estimator in which structured state representations with learned, value-aware field weights replace text-level matching and kernel smoothing replaces hard grouping. Its temperature interpolates between exact matching and the task-level mean, and in terms of baseline MSE a tuned temperature is never worse than either endpoint.
- Experiments on ALFWorld and WebShop show that kernel smoothing gives every step a comparison set, including those that GiGPO leaves without step-level credit, and that SSA improves over GiGPO and HGPO on both benchmarks at both model scales.

2 Related Work

Group-based RL methods such as GRPO [Shao et al., 2024] assign a single trajectory-level advantage to all tokens in a rollout, providing no mechanism to distinguish steps that contributed positively from those that did not. While scaling RL training for LLMs has seen rapid progress [Guo et al., 2025, Team et al., 2025, Yu et al., 2025], credit assignment in multi-turn agent settings, from embodied navigation [Shridhar et al., 2021] to web interaction [Yao et al., 2022, Nakano et al., 2021], remains an open challenge [Wang et al., 2025b, Chen et al., 2025]. To enable finer-grained credit, GiGPO [Feng et al., 2025] introduces anchor state grouping: steps that visit the same environment state across different rollouts are grouped together, and advantages are normalized within each group. This provides meaningful step-level signal when groups are large, but relies on exact text matching to determine state identity, which produces singleton groups for 30–36% of steps. HGPO [He et al., 2026] mitigates this by incorporating historical context into the matching criterion, reducing singletons at the cost of a more complex grouping hierarchy. Both methods still compare raw observation text and do not ask whether a textual difference reflects a difference in value. SSA instead decomposes observations into typed fields, learns which fields predict returns, and computes a kernel-smoothed advantage over structured similarity rather than text similarity.

State abstraction, the problem of merging states with equivalent value functions, has a long history in tabular RL [Li et al., 2006, Abel et al., 2018, Sutton and Barto, 2018]. Bisimulation metrics [Ferns et al., 2004, Castro, 2020] formalize value equivalence for continuous state spaces. Deep bisimulation [Zhang et al., 2020] learns state representations where proximity reflects value equivalence. These methods typically require learning a separate encoder and are designed for low-dimensional continuous control. In the LLM domain, recent work on hierarchical multi-turn RL [Zhou et al., 2024, Wen et al., 2024, Putta et al., 2024], Monte Carlo credit assignment from intermediate states [Kazemnejad et al., 2025], and chain-of-thought reasoning [Wei et al., 2022, Yao et al., 2023] also grapples with how to evaluate individual actions within long episodes. SSA obtains a simple form of state abstraction for LLM agents from the structured state output that the environment already provides, rather than learning representations from scratch.

3 Preliminaries

We consider a language model agent that generates a trajectory $\boldsymbol{\tau} = \{(\mathbf{s}_t, \mathbf{a}_t, r_t)\}_{t=1}^{T}$ for a task x, where $\mathbf{s}_t \in \mathcal{S}$ is the observed state, $\mathbf{a}_t \sim \pi_\theta(\cdot \mid \mathbf{s}_t, x)$ the action, and r_t the reward [Sutton and Barto, 2018]. Most agent tasks provide only a terminal reward $R(\boldsymbol{\tau}) \in \{0, 1\}$.

GRPO [Shao et al., 2024] samples N trajectories per task and assigns a trajectory-level advantage via z-score normalization [Ahmadian et al., 2024, Kool et al., 2019]:

$$A^E(\boldsymbol{\tau}_i) = \frac{R(\boldsymbol{\tau}_i) - \mu}{\sigma}, \quad \mu = \frac{1}{N}\sum_{j=1}^{N} R(\boldsymbol{\tau}_j), \quad \sigma = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(R(\boldsymbol{\tau}_j) - \mu)^2}. \tag{1}$$

This assigns the same advantage to every token in trajectory i, regardless of individual step contributions.

GiGPO [Feng et al., 2025] refines credit to the step level by grouping all (action, return) pairs observed at a recurring state $\tilde{\mathbf{s}}$:

$$G^S(\tilde{\mathbf{s}}) = \left\{(\mathbf{a}_t^{(i)}, R_t^{(i)}) \;\middle|\; \mathbf{s}_t^{(i)} = \tilde{\mathbf{s}},\ 1 \le i \le N,\ 1 \le t \le T\right\}, \tag{2}$$

where state equality is determined by exact text matching of the raw observation string, and $R_t^{(i)} = \sum_{k=t}^{T} \gamma^{k-t} r_k^{(i)}$ is the return-to-go. The step-level advantage normalizes each return against its group baseline (see Appendix B for the full formulation), and the final advantage combines both levels as $A(\mathbf{a}_t^{(i)}) = A^E(\boldsymbol{\tau}_i) + \omega \cdot A^S(\mathbf{a}_t^{(i)})$, where ω weights the step-level term.

The quality of A^S depends on the groups G^S. Exact text matching discards structured metadata returned by the environment (e.g., `admissible_commands` in ALFWorld), and it produces many singleton groups when observation text varies across rollouts. The context-aware matching of HGPO reduces singletons but keeps the text-level comparison (Section 2).

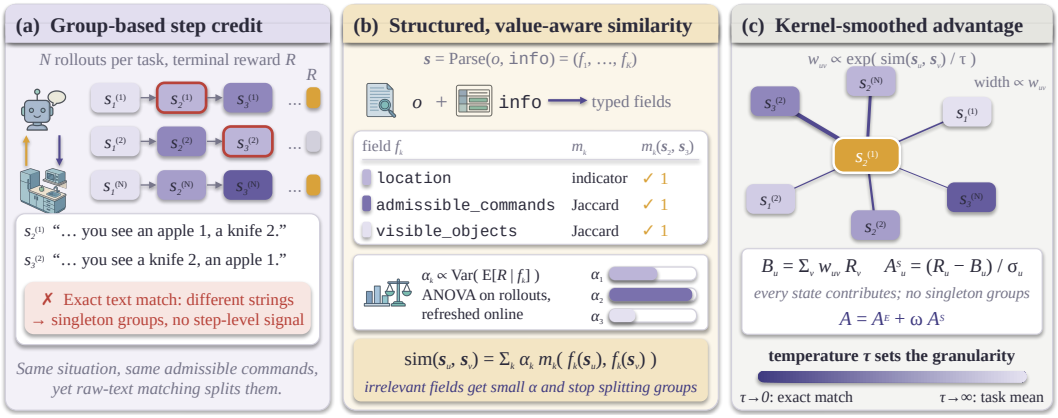

Figure 2: Overview of SSA. (a) Group-based methods match steps by raw observation text, so two steps that face the same situation but list objects in a different order fall into singleton groups and receive no step-level signal. (b) SSA parses each step into typed fields, compares them with field-specific metrics, and weights each field by how much it explains return variance. (c) The resulting similarity defines kernel weights over all steps, which give a smoothed baseline and a step advantage; the temperature τ moves between exact matching and the task-level mean.

4 Method: Structured State Abstraction

We present SSA, a step-level credit assignment method for multi-turn agent RL (Figure 2). It replaces text comparison with a structured, value-aware similarity and hard grouping with kernel smoothing, so that every step receives a comparative signal. We first formalize why value-aware matching matters, then describe each component.

4.1 Why Value-Aware Matching?

A step-level baseline should estimate $V^\pi(\mathbf{s})$, the expected return-to-go from state $\mathbf{s}$ under the current policy π. Let $\phi : \mathcal{S} \to \tilde{\mathcal{S}}$ be a state abstraction, and let the group $G_\phi(\mathbf{s})$ collect the steps of the same task whose abstract state equals $\phi(\mathbf{s})$. The group baseline $\hat{V}_\phi(\mathbf{s})$ is the mean return-to-go over $G_\phi(\mathbf{s})$. Assume that each observed return is $R = V^\pi(\mathbf{s}) + \eta$ with zero-mean noise of variance σ^2, independent across the members of a group, and that group members are distributed as states given their abstract state. Then the mean squared error of the baseline decomposes as

$$\mathrm{MSE}(\phi) \approx \underbrace{\mathbb{E}\big[\mathrm{Var}(V^\pi(\mathbf{s}) \mid \phi(\mathbf{s}))\big]}_{\text{grouping bias}} + \underbrace{\mathbb{E}\left[\frac{\sigma^2}{|G_\phi(\mathbf{s})|}\right]}_{\text{estimation variance}}, \tag{3}$$

where the first term measures how much the value varies inside a group and the second measures the noise of averaging over a finite group (Appendix E gives the derivation and the exact finite-group form). The two terms pull in opposite directions: a finer abstraction lowers the first term but shrinks groups and raises the second, and a coarser abstraction does the reverse.

The decomposition gives a simple matching criterion. A field that carries no information about V^π beyond the fields already matched leaves the bias term unchanged, yet matching on it splits groups and raises the variance term (Appendix D). A field that does carry such information lowers the bias term when it is matched. Good matching therefore attends to value-relevant fields and ignores the rest, which is what the field weights in Section 4.3 aim to do.

4.2 Structured State Representation

In multi-turn environments the observation is a text string produced by the environment engine. Prior methods treat it as opaque text, but it encodes a small number of *typed fields* with distinct semantic roles, and many of these fields are also returned as structured data in the `info` dictionary of the environment.

Definition 1 (Structured State)**.** *A structured state* $\mathbf{s} = (f_1, f_2, \ldots, f_K)$ *is a tuple of* K *typed fields, where each field* $f_k \in \mathcal{F}_k$ *is extracted from either the environment's metadata or the observation text:*

$$\mathbf{s} = \mathrm{Parse}(o, \texttt{info}) = \big(f_1(o, \texttt{info}),\ f_2(o, \texttt{info}),\ \ldots,\ f_K(o, \texttt{info})\big). \tag{4}$$

Fields taken from `info` need no text parsing. Table 5 in Appendix F lists the fields for both environments and marks which come from the environment API and which need lightweight text extraction. SSA treats the parsed fields as the value-relevant part of the state: history that the fields do not record is ignored, which is the information that HGPO adds through context. Assumption 1 in Appendix E.3 states this requirement formally.

4.3 Field-Importance Learning

Not all fields equally influence the value function. We learn per-field importance weights $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K) \in \Delta^{K-1}$ from rollout data, so that matching prioritizes value-relevant fields. If states that differ in field k have very different average returns, field k is value-relevant and should receive a high weight.

Return-variance decomposition. We measure this with a one-way ANOVA for each field. Given rollout data $\{(\mathbf{s}_t^{(i)}, R_t^{(i)})\}_{i,t}$, define the *between-group variance* when grouping by field k alone:

$$V_k = \mathrm{Var}\big(\mathbb{E}[R \mid f_k]\big) = \frac{1}{|\mathcal{D}|} \sum_{c \in \mathcal{F}_k} n_c \cdot (\bar{R}_c - \bar{R})^2, \tag{5}$$

where $\mathcal{D}$ is the rollout dataset, n_c is the number of steps with $f_k = c$, $\bar{R}_c$ is the mean return for steps with $f_k = c$, and $\bar{R}$ is the global mean return. V_k measures how much of the total return variance is explained by field k.

The importance weight for field k is the normalized between-group variance:

$$\alpha_k = \frac{V_k}{\sum_{k'=1}^{K} V_{k'}}. \tag{6}$$

Fields whose values predict returns therefore dominate the similarity, and fields unrelated to returns receive weights close to zero.

Practical considerations. For discrete fields, V_k is the standard one-way ANOVA statistic. ANOVA needs a partition of the steps, so set fields are discretized by a canonical hash (sort the elements and hash the resulting string) and text fields by hashing their token set; the match function in Eq. 7 still uses the Jaccard index, so partially overlapping sets count as similar. The computation needs no additional model, only variance statistics of the existing rollout batch. The estimate has two known weaknesses. The plug-in V_k is biased upward by roughly the number of distinct observed values times $\sigma^2/|\mathcal{D}|$, so a high-cardinality field can receive inflated weight when the batch is small (Appendix E.2). The score is also marginal: two correlated fields can both receive weight even when one of them would suffice.

Online update. Weights $\boldsymbol{\alpha}$ are updated every E training iterations using the most recent rollout batch, allowing adaptation as the policy evolves. Early in training, coarse fields (location) may dominate; later, fine-grained fields (specific object states) may become more relevant.

4.4 Kernel-Smoothed Advantage Estimation

We index steps by $u = (i, t)$, meaning step t of rollout i, and write $\mathbf{s}_u = \mathbf{s}_t^{(i)}$ and $R_u = R_t^{(i)}$. Given the field weights, the *structured similarity* of two steps is

$$\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v) = \sum_{k=1}^{K} \alpha_k \cdot m_k\big(f_k(\mathbf{s}_u),\ f_k(\mathbf{s}_v)\big), \tag{7}$$

where $m_k : \mathcal{F}_k \times \mathcal{F}_k \to [0, 1]$ is a field-specific match function. For discrete fields (location, page type), $m_k(a, b) = \mathbb{1}[a = b]$. For set fields (admissible commands, visible objects), $m_k(A, B) = |A \cap B|/|A \cup B|$ (Jaccard index).

From hard grouping to kernel smoothing. One could group all states with $\mathrm{sim} \geq \delta$ for a threshold δ. A hard threshold fixes one tradeoff for all states: a strict δ leaves singleton groups that carry no step-level credit, and a loose δ merges states with different values and biases the baseline. The right granularity also changes across states and over training, so no single δ fits.

SSA uses *kernel smoothing* instead: every step of the same task contributes to the baseline of u with a weight that grows with its structured similarity to u, and no step is included or excluded outright. Let $\mathcal{G}(u)$ be the set of all steps in the N rollouts of the task of u, including u itself. The weights are

$$w_{uv} = \frac{\exp\big(\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v)/\tau\big)}{\sum_{v' \in \mathcal{G}(u)} \exp\big(\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_{v'})/\tau\big)}, \qquad v \in \mathcal{G}(u), \tag{8}$$

where $\tau > 0$ is the temperature. Because every m_k lies in $[0, 1]$ with $m_k(a, a) = 1$ and the weights α_k sum to one, the similarity is at most one and equals one exactly when two steps agree on every field with $\alpha_k > 0$. The temperature therefore interpolates between two known estimators. As $\tau \to 0$, the weights become uniform over the steps whose structured state matches $\mathbf{s}_u$ exactly, which is anchor-state grouping on the structured state; a step with no exact match keeps only itself, as a GiGPO singleton would. As $\tau \to \infty$, the weights become uniform over $\mathcal{G}(u)$, which gives a task-level mean in the spirit of the GRPO baseline. At intermediate τ, every step contributes according to its structured similarity, and a step without an exact match still has a comparison set.

The kernel-smoothed baseline and step-level advantage are:

$$B_u = \sum_{v \in \mathcal{G}(u)} w_{uv}\, R_v, \tag{9}$$

$$A^S(\mathbf{a}_u) = \frac{R_u - B_u}{\sigma_u + \epsilon}, \qquad \sigma_u = \sqrt{\textstyle\sum_{v \in \mathcal{G}(u)} w_{uv} \big(R_v - B_u\big)^2}, \tag{10}$$

where $\epsilon = 10^{-8}$ prevents division by zero. The normalization aligns the step-level advantage scale with the trajectory-level advantage.

The baseline B_u is a Nadaraya–Watson kernel regression [Nadaraya, 1964, Watson, 1964] of the return-to-go on the structured state, evaluated at $\mathbf{s}_u$. Under the assumptions of Eq. 3, its squared error given the states is $\big(\sum_v w_{uv}(V^\pi(\mathbf{s}_v) - V^\pi(\mathbf{s}_u))\big)^2 + \sigma^2 \sum_v w_{uv}^2$, so the effective group size $n_{\mathrm{eff},u} = 1/\sum_v w_{uv}^2$ takes the place of $|G_\phi|$ (Proposition 4). For scale, with $\tau = 0.1$ a step that differs from $\mathbf{s}_u$ in fields of total weight 0.3 receives $e^{-3} \approx 0.05$ of the weight of an exact match.

Proposition 1 (Temperature interpolates between exact matching and averaging)**.** *Assume bounded returns. Let $B(\tau)$ denote the kernel baseline of Eq. 9, let B^{exact} average R_v over the steps that match $\mathbf{s}_u$ exactly on the structured state, and let B^{avg} average R_v over all of $\mathcal{G}(u)$. Then $B(\tau) \to B^{\mathrm{exact}}$ as $\tau \to 0$ and $B(\tau) \to B^{\mathrm{avg}}$ as $\tau \to \infty$, and with MSE$(B) = \mathbb{E}\big[(B_u - V^\pi(\mathbf{s}_u))^2\big]$,*

$$\inf_{\tau > 0} \mathit{MSE}\big(B(\tau)\big) \leq \min\big\{\mathit{MSE}(B^{\mathrm{exact}}),\ \mathit{MSE}(B^{\mathrm{avg}})\big\}. \tag{11}$$

The proof is in Appendix C. We state the result for the baseline because it fixes the centering of the advantage, while the normalization only rescales it. Proposition 1 says that a tuned temperature is never worse than either endpoint; in between, high-similarity steps dominate the baseline while the remaining steps still reduce its variance. The same argument covers hard thresholds: with the truncated kernel $\exp(\mathrm{sim}/\tau)\mathbb{1}[\mathrm{sim} \geq \delta]$, the hard-threshold estimator at δ is the $\tau \to \infty$ endpoint, so the infimum of the MSE along that path is at most that of thresholding at δ. The value-aware weights $\boldsymbol{\alpha}$ determine which steps count as similar, so that at moderate τ the weight goes to steps with similar values rather than merely similar surface text. We discuss the relationship between SSA and existing methods (GiGPO, HGPO) in Appendix H.

4.5 Training Objective

The final advantage combines episode-level and step-level contributions:

$$A(\mathbf{a}_t^{(i)}) = A^E(\boldsymbol{\tau}_i) + \omega \cdot A^S_{\mathrm{SSA}}(\mathbf{a}_t^{(i)}), \tag{12}$$

where A^E is the GRPO episode-level advantage (Eq. 1) and A^S_{SSA} is the kernel-smoothed step-level advantage (Eq. 10). The policy is optimized via the standard clipped surrogate objective [Schulman et al., 2017]:

$$\mathcal{J}_{SSA}(\theta) = \mathbb{E}_{\substack{x\sim p(X) \\ \{\boldsymbol{\tau}_i\}_{i=1}^{N}\sim\pi_{\theta_{\text{old}}}}} \left[\frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\min\Big(\rho_t^{(i)}A(\mathbf{a}_t^{(i)}),\ \text{clip}\big(\rho_t^{(i)},1\pm\epsilon\big)A(\mathbf{a}_t^{(i)})\Big)\right] - \beta\,\mathbb{D}_{\text{KL}}\big(\pi_\theta\|\pi_{\text{ref}}\big), \tag{13}$$

where $\rho_t^{(i)} = \pi_\theta(\mathbf{a}_t^{(i)} \mid \mathbf{s}_t^{(i)}, x)/\pi_{\theta_{\text{old}}}(\mathbf{a}_t^{(i)} \mid \mathbf{s}_t^{(i)}, x)$ is the importance ratio.

5 Experiments

5.1 Setup

Benchmarks. We evaluate on two multi-turn agent environments with distinct state structure characteristics. ALFWorld [Shridhar et al., 2021] consists of embodied household tasks across 6 task types. State is relatively deterministic: the same game state produces consistent text, and the exact matching of GiGPO achieves $\sim$65% non-singleton rate. Structured fields include `admissible_commands` (from env), location, and visible objects (from text). WebShop [Yao et al., 2022] is an e-commerce web navigation environment. State is HTML-rendered text with more variation. Structured fields include `available_actions` (from env), page type, and selected options (from text).

Baselines. All RL methods use Qwen2.5-Instruct [Qwen et al., 2025] with the same rollout budget (N=8 per prompt).[1] We evaluate at two scales (1.5B and 7B) and compare against prompting baselines (ReAct [Yao et al., 2023], Reflexion [Shinn et al., 2024]), PPO [Schulman et al., 2017] (with critic), RLOO, GRPO [Shao et al., 2024] (episode-level), GiGPO [Feng et al., 2025] (exact text matching), and HGPO [He et al., 2026] (context-aware matching).

Implementation. SSA modifies the `build_step_group` function in the GiGPO codebase. The structured state parser for each environment is <100 lines of Python. Fields from the `info` dictionary are used directly; text-extracted fields use simple regex patterns. Field-importance weights $\boldsymbol{\alpha}$ are updated every E=10 iterations via one-way ANOVA on the latest rollout batch.

5.2 Main Results

Table 1 presents the main comparison on ALFWorld and WebShop across two model scales. All results are averaged over 3 random seeds.

SSA achieves the best performance on both benchmarks at both model scales. On ALFWorld, the improvements hold across all six subtask categories. The largest gains are on Look, Cool, and Pick2, where the admissible commands, the field with the highest learned weight, vary most across states that text matching would conflate. On simpler subtasks like Pick and Heat, where most methods already achieve high accuracy, the gains are smaller but still positive.

On WebShop, SSA improves over HGPO at both scales. WebShop observations contain HTML-rendered text with substantial surface variation across pages of different types, which makes text-based matching particularly noisy. By matching on the structured `available_actions` field rather than raw HTML, SSA avoids grouping states from different page types that happen to share vocabulary.

On both benchmarks, RL methods outperform prompting baselines by a wide margin, and among RL methods the step-level credit methods (GiGPO, HGPO, SSA) outperform the episode-level ones (PPO, RLOO, GRPO). SSA improves further over GiGPO and HGPO; it differs from them in the state representation (structured fields instead of text) and in the aggregation (kernel smoothing instead of hard groups).

[1] PPO, RLOO, GRPO, and GiGPO numbers at 1.5B from Feng et al. [2025]; at 7B, PPO and RLOO are reproduced, GiGPO from Feng et al. [2025]. HGPO (K=2) from He et al. [2026] at both scales.

Table 1: Performance on ALFWorld and WebShop. ALFWorld reports per-subtask and overall success rate (%); WebShop reports score and success rate (%). Best in **bold**, second best underlined within each scale.

Type	Method	ALFWorld (SR%)							WebShop	
		Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
Qwen2.5-1.5B-Instruct										
Prompting	ReAct	17.4	20.5	15.7	6.2	7.7	2.0	12.8	40.1	11.3
Prompting	Reflexion	35.3	22.2	21.7	13.6	19.4	3.7	21.8	55.8	21.9
RL Training	PPO	64.8	40.5	57.1	60.6	46.4	47.4	54.4	73.8	51.5
RL Training	RLOO	88.3	52.8	71.0	62.8	66.4	56.9	69.7	73.9	52.1
RL Training	GRPO	80.0	50.0	75.0	88.9	63.2	50.0	70.3	73.1	52.2
RL Training	GiGPO	95.3	80.2	92.9	92.7	70.6	78.5	86.7	83.1	65.0
RL Training	HGPO	95.5	86.8	93.2	93.5	84.5	84.8	91.2	86.2	72.5
RL Training	**SSA**	**96.8**	**89.5**	**95.2**	**94.8**	**87.2**	**87.5**	**93.1**	**88.5**	**75.2**
Qwen2.5-7B-Instruct										
Prompting	ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2	46.2	19.5
Prompting	Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	42.7	58.1	28.8
RL Training	PPO	92.3	64.0	92.5	89.5	80.3	68.8	80.4	81.4	68.7
RL Training	RLOO	87.6	78.2	87.3	81.3	71.9	48.9	75.5	80.3	65.7
RL Training	GRPO	90.7	66.2	94.1	91.2	78.9	70.5	79.8	79.2	67.2
RL Training	GiGPO	97.5	83.2	95.8	90.2	89.3	83.5	93.2	86.8	76.5
RL Training	HGPO	97.8	85.5	96.2	92.8	91.5	86.2	94.5	88.2	78.8
RL Training	**SSA**	**98.5**	**88.8**	**97.5**	**94.5**	**93.8**	**89.2**	**96.5**	**89.8**	**80.5**

Singleton rate and group size. Figure 3(a) compares the effective singleton rate between GiGPO and SSA. Although SSA uses kernel smoothing rather than hard groups, we report the fraction of steps where no other state achieves $\mathrm{sim} > 0.5$ as an effective singleton metric. On ALFWorld, GiGPO produces singletons for over a third of steps, while SSA reduces this substantially through structured field matching. On WebShop, the improvement is similar: structured matching by available actions and page type yields more informative comparisons than raw text. The group size distribution in Figure 3(b) shows that SSA shifts mass from singletons to larger groups.

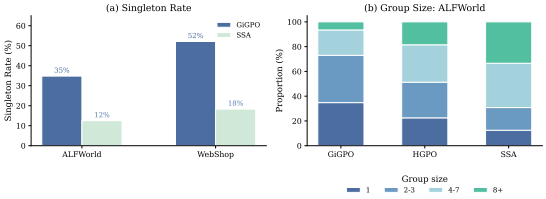

Figure 3: (a) Effective singleton rate: SSA reduces singletons across both environments. (b) Group size distribution on ALFWorld.

Learned field importance. Figure 4 shows the learned field-importance weights α_k for each environment. The weights match what the tasks suggest. In ALFWorld, admissible commands dominate, followed by location; the former directly constrains the agent's action space at each step. In WebShop, available actions receive the highest weight, followed by page type. In both environments, decision-constraining fields (what actions are available) receive higher importance than state-describing fields (what objects are visible), suggesting that the constraints on an agent's choices are more predictive of future returns than descriptions of the environment.

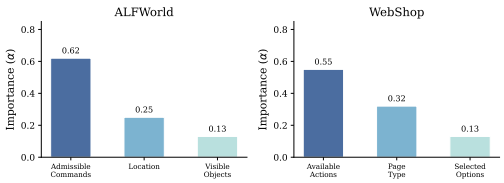

Figure 4: Learned field-importance weights α_k. Decision-constraining fields dominate.

5.3 Training Dynamics

Figure 5 compares the training dynamics of GRPO, GiGPO, HGPO, and SSA across four metrics. SSA converges faster and to a higher final episode success rate than all baselines. The KL divergence

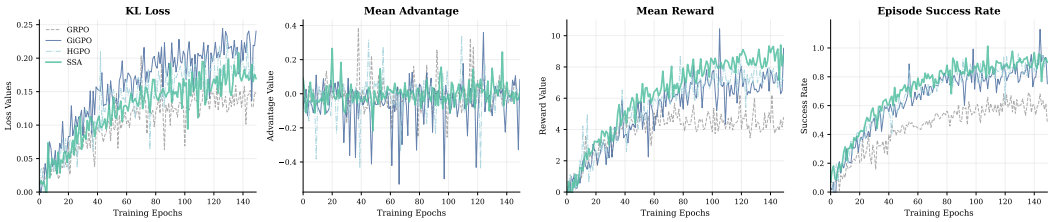

Figure 5: Training dynamics on ALFWorld.

Table 2: Ablation study. Each row adds one component to the GRPO baseline (Qwen2.5-7B-Instruct).

Variant	ALFWorld (All)	WebShop (Succ.)
GRPO (baseline)	$79.8_{\pm 2.3}$	$67.2_{\pm 1.8}$
+ structured fields, uniform α	$90.5_{\pm 1.5}$	$75.2_{\pm 1.4}$
+ learned α (ANOVA)	$93.8_{\pm 1.1}$	$78.0_{\pm 1.2}$
+ kernel smoothing (full SSA)	$\mathbf{96.5}_{\pm 0.8}$	$\mathbf{80.5}_{\pm 1.0}$
w/o online α update	$95.0_{\pm 1.0}$	$79.2_{\pm 1.1}$

of SSA sits between GRPO (lowest, weakest policy update) and GiGPO/HGPO (highest), indicating that SSA makes effective policy updates without excessive divergence from the reference policy. Mean advantage oscillations are smaller for SSA than for GiGPO and HGPO, suggesting that the kernel-smoothed estimator produces more stable advantage signals. Mean reward rises fastest for SSA and reaches a higher asymptote, consistent with the main results in Table 1.

5.4 Ablation Study

Table 2 isolates the contribution of each SSA component by incrementally adding them to the GRPO baseline (Qwen2.5-7B-Instruct).

Structured fields with uniform weights already give a large gain over GRPO, so typed field matching helps even before any weighting. Learning the weights with ANOVA adds a further gain, which indicates that return variance picks out the value-relevant fields. Kernel smoothing adds another improvement, in line with Proposition 1: with continuous weights the estimator uses comparisons that a hard boundary would discard. Fixing $\boldsymbol{\alpha}$ at its initial estimate lowers performance, so the weights should follow the policy as the value-relevant fields change during training. Additional ablations on field source are in Appendix I.

5.5 Computational Efficiency

Following GiGPO [Feng et al., 2025], we measure per-iteration overhead on ALFWorld using Qwen2.5-1.5B-Instruct (2×H100). Table 3 shows the breakdown. Shared components (rollouts, old/ref probability computation, policy updates) total 425.6s per iteration. SSA replaces the hashmap lookup of GiGPO with structured field extraction (0.05s), weighted similarity computation (0.62s), and adds periodic ANOVA updates (0.18s amortized). The step-relative advantage computation (0.53s) is shared with GiGPO. The total overhead of SSA relative to GRPO is 1.38s, representing $<$0.4% of the per-iteration training time.

6 Conclusion

Viewing critic-free baselines as value estimates, this paper presents SSA, which addresses two limitations of existing step-level estimates: value-agnostic state comparison and hard grouping. With typed fields, learned field weights, and kernel smoothing, SSA improves over GiGPO and HGPO on ALFWorld and WebShop at two model scales. The main limitation is the need for environment-specific field definitions, though most value-relevant fields come directly from the environment's `info` dictionary; future work could explore automatic field discovery via LLMs and tool-calling work benchmarks [Trivedi et al., 2024, Yao et al., 2024, Shepard and Salimans, 2026].

Table 3: Per-iteration overhead on ALFWorld (Qwen2.5-1.5B, 2×H100). Shared components (rollouts, probability computation, policy updates) total 425.6s.

Component	Time (s)
GiGPO: anchor state grouping (hashmap)	0.01
GiGPO: step-relative advantage	0.53
SSA: structured field extraction	0.05
SSA: weighted similarity computation	0.62
SSA: step-relative advantage	0.53
SSA: ANOVA field-importance (amortized)	0.18
GiGPO total add. vs. GRPO	**0.54 (<0.2%)**
SSA total add. vs. GRPO	**1.38 (<0.4%)**

Acknowledgments and Disclosure of Funding

This work received no third-party funding or support. The authors declare no competing interests.

References

David Abel, Dilip Arumugam, Lucas Lehnert, and Michael Littman. State abstractions for lifelong reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *PMLR*, pages 10–19, 2018.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, 2024.

Lang Cao, Hui Ruan, Yongqian Li, Peng Chao, Wu Ning, Haonan Song, Renhong Chen, and Yitong Li. Treeadv: Tree-structured advantage redistribution for group-based rl, 2026. URL `https://arxiv.org/abs/2601.03703`.

Pablo Samuel Castro. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10069–10076, 2020.

Kevin Chen, Marco Cusumano-Towner, Brody Huval, Aleksei Petrenko, Jackson Hamburger, Vladlen Koltun, and Philipp Krähenbühl. Reinforcement learning for long-horizon interactive llm agents. *arXiv preprint arXiv:2502.01600*, 2025.

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. *arXiv preprint arXiv:2507.19849*, 2025.

Guanting Dong, Licheng Bao, Zhongyuan Wang, Kangzhi Zhao, Xiaoxi Li, Jiajie Jin, Jinghan Yang, Hangyu Mao, Fuzheng Zhang, Kun Gai, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Toward generalized web agent training: A deep dive into entropy-balanced reinforcement learning. In *Proceedings of the ACM Web Conference 2026*, pages 2126–2137, 2026.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. WorkArena: How capable are web agents at solving common knowledge work tasks? In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *PMLR*, pages 11642–11662, 2024.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=QXEhBMNrCW`.

Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *UAI*, volume 4, pages 162–169, 2004.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=T8Dev99qnz.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! In *ICLR 2019 Workshop*, 2019.

Chenliang Li, Adel Elmahdy, Alex Boyd, Zhongruo Wang, Siliang Zeng, Alfredo Garcia, Parminder Bhatia, Taha Kass-Hout, Cao Xiao, and Mingyi Hong. Stabilizing off-policy training for long-horizon LLM agent via turn-level importance sampling and clipping-triggered normalization. *arXiv preprint arXiv:2511.20718*, 2025.

Jiazheng Li, Yawei Wang, Qiaojing Yan, Yijun Tian, Zhichao Xu, Huan Song, Panpan Xu, and Lin Lee Cheong. SALT: Step-level advantage assignment for long-horizon agents via trajectory graph. In Vera Demberg, Kentaro Inui, and Lluís Marquez, editors, *Findings of the Association for Computational Linguistics: EACL 2026*, pages 4709–4725, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-386-9. doi: 10.18653/v1/2026.findings-eacl.247. URL https://aclanthology.org/2026.findings-eacl.247/.

Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a unified theory of state abstraction for mdps. *ISAIM*, 2006.

Elizbar A. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1): 141–142, 1964.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, et al. GDPval: Evaluating AI model performance on real-world economically valuable tasks. *arXiv preprint arXiv:2510.04374*, 2025.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent Q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*, 2024.

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, 2025. URL https://arxiv.org/abs/2412.15115.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Daniel Shepard and Robin Salimans. AutomationBench. *arXiv preprint arXiv:2604.18934*, 2026.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

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

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2603.08754*, 2026.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1.5: Scaling reinforcement learning with LLMs. *arXiv preprint arXiv:2501.12599*, 2025.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16022–16076, 2024.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. *arXiv preprint arXiv:2510.14967*, 2026.

Jiawei Wang, Jiacai Liu, Yuqian Fu, Yingru Li, Xintao Wang, Yuan Lin, Yu Yue, Lin Zhang, Yang Wang, and Ke Wang. Harnessing uncertainty: Entropy-modulated policy gradients for long-horizon LLM agents. *arXiv preprint arXiv:2509.09265*, 2025a.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, et al. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025b.

Geoffrey S. Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, 26(4):359–372, 1964.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

Muning Wen, Ziyu Wan, Jun Wang, Weinan Zhang, and Ying Wen. Reinforcing LLM agents via policy optimization with action decomposition. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Yajie Yang, Wenxiang Chen, Zhihao Zhang, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, Tao Ji, Tao Gui, Qi Zhang, and Xuanjing Huang. Agentprm: Process reward models for LLM agents via step-wise promise and progress. *arXiv preprint arXiv:2511.08325*, 2025.

Frank F. Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, et al. TheAgentCompany: Benchmarking LLM agents on consequential real world tasks. *arXiv preprint arXiv:2412.14161*, 2024.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 20744–20757. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/82ad13ec01f9fe44c01cb91814fd7b8c-Paper-Conference.pdf.

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 *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, and Mingyi Hong. Reinforcing multi-turn reasoning in LLM agents via turn-level credit assignment. *arXiv preprint arXiv:2505.11821*, 2025.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

Yaocheng Zhang, Haohuan Huang, Zijun Song, Zijie Zhao, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. CriticSearch: Fine-grained credit assignment for search agents via a retrospective critic. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 12272–12290, 2026.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn rl. In *International Conference on Machine Learning*, pages 62178–62209. PMLR, 2024.

A Algorithm

Algorithm 1 summarizes the SSA training procedure. The only modifications to a standard GRPO training loop are: (1) the `Parse` function that decomposes observations into typed fields (line 5), (2) the ANOVA-based field-importance update (lines 6–8), and (3) the structured similarity and kernel-smoothed advantage computation (lines 9–13).

Algorithm 1 SSA Training

Require: Policy π_θ, task set $\mathcal{X}$, rollout budget N, temperature τ, update interval E

1: Initialize field-importance weights $\boldsymbol{\alpha} \leftarrow$ uniform
2: **for** iteration $= 1, 2, \ldots$ **do**
3: Sample task $x \sim \mathcal{X}$; collect N trajectories $\{\boldsymbol{\tau}_i\}_{i=1}^{N}$ from π_θ
4: Compute episode-level advantage $A^E(\boldsymbol{\tau}_i)$ via GRPO z-score (Eq. 1)
5: Parse each step observation: $\mathbf{s}_t^{(i)} \leftarrow \texttt{Parse}(o_t^{(i)}, \texttt{info}_t^{(i)})$
6: **if** iteration mod $E = 0$ **then**
7: Update $\boldsymbol{\alpha}$ via ANOVA on $\{(\mathbf{s}_t^{(i)}, R_t^{(i)})\}$ (Eqs. 5–6)
8: **end if**
9: Compute pairwise structured similarity $\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v) = \sum_k \alpha_k \, m_k(\cdot)$ for steps u, v of the same task (Eq. 7)
10: Compute kernel weights $w_{uv} = \mathrm{softmax}_{v \in \mathcal{G}(u)}(\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v)/\tau)$ (Eq. 8)
11: Compute kernel-smoothed baseline $B_u = \sum_v w_{uv} R_v$ (Eq. 9)
12: Compute step-level advantage A^S_{SSA} (Eq. 10)
13: Combine: $A(\mathbf{a}_t^{(i)}) = A^E(\boldsymbol{\tau}_i) + \omega \cdot A^S_{\mathrm{SSA}}(\mathbf{a}_t^{(i)})$
14: Update θ via clipped surrogate objective (Eq. 13)
15: **end for**

B Preliminaries: Full Formulation

For completeness, we reproduce the full step-level credit formulation from GiGPO [Feng et al., 2025], which our method builds upon. The formulation extends the standard policy gradient framework [Sutton and Barto, 2018, Schulman et al., 2017] to multi-turn agent settings.

Discounted return-to-go. At step t of trajectory i, the return-to-go is defined as:

$$R_t^{(i)} = \sum_{k=t}^{T} \gamma^{k-t} \, r_k^{(i)}. \tag{14}$$

Step-level advantage (GiGPO). Within each step-level group $G^S(\tilde{\mathbf{s}})$ (Eq. 2), GiGPO computes the step-level advantage as:

$$A^S(\mathbf{a}_t^{(i)}) = \frac{R_t^{(i)} - \mathrm{mean}\big(\{R_t^{(j)} \mid (\mathbf{a}_t^{(j)}, R_t^{(j)}) \in G^S(\tilde{\mathbf{s}})\}\big)}{F_{\mathrm{norm}}\big(\{R_t^{(j)} \mid (\mathbf{a}_t^{(j)}, R_t^{(j)}) \in G^S(\tilde{\mathbf{s}})\}\big)}, \tag{15}$$

where F_{norm} is a normalization function (typically std or 1). The final training advantage combines episode-level and step-level contributions:

$$A(\mathbf{a}_t^{(i)}) = A^E(\boldsymbol{\tau}_i) + \omega \cdot A^S(\mathbf{a}_t^{(i)}), \tag{16}$$

with hyperparameter ω controlling the step-level contribution. SSA thus changes two parts of GiGPO: text matching becomes structured similarity, and hard grouping becomes kernel smoothing.

The state matching bottleneck. GiGPO determines group membership via exact text matching of the raw observation string. This has two limitations. First, the environment returns structured metadata alongside text (e.g., `admissible_commands` in ALFWorld, `available_actions` in WebShop), but the matching anchor discards it. Second, states sharing location and inventory but differing in irrelevant details are placed in separate groups. HGPO [He et al., 2026] extends the matching criterion with historical context, but still treats observations as unstructured text.

C Proof of Proposition 1

Proof. Fix the rollouts of a task and a step u. Since $m_k \in [0,1]$, $m_k(a,a) = 1$, and $\sum_k \alpha_k = 1$, we have $\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v) \le 1 = \mathrm{sim}(\mathbf{s}_u, \mathbf{s}_u)$. Let $M_u = \{v \in \mathcal{G}(u) : \mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v) = 1\}$, the steps that match u on every field with $\alpha_k > 0$; it contains u and is therefore nonempty. Dividing the numerator and denominator of Eq. 8 by $\exp(1/\tau)$ gives

$$w_{uv}(\tau) = \frac{\exp\big((\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v) - 1)/\tau\big)}{\sum_{v' \in \mathcal{G}(u)} \exp\big((\mathrm{sim}(\mathbf{s}_u, \mathbf{s}_{v'}) - 1)/\tau\big)}. \tag{17}$$

As $\tau \to 0$, each term with $v \in M_u$ equals one and each term with $v \notin M_u$ tends to zero, so $w_{uv}(\tau) \to \mathbb{1}[v \in M_u]/|M_u|$ and $B_u(\tau) \to B_u^{\mathrm{exact}}$. As $\tau \to \infty$, every term tends to one, so $w_{uv}(\tau) \to 1/|\mathcal{G}(u)|$ and $B_u(\tau) \to B_u^{\mathrm{avg}}$. Both limits hold for every realization of the rollouts.

Let $|R| \le R_{\max}$. Then $|B_u(\tau)| \le R_{\max}$ and $|V^\pi| \le R_{\max}$, so $(B_u(\tau) - V^\pi(\mathbf{s}_u))^2 \le 4R_{\max}^2$. By dominated convergence, $\mathrm{MSE}(B(\tau))$ tends to $\mathrm{MSE}(B^{\mathrm{exact}})$ as $\tau \to 0$ and to $\mathrm{MSE}(B^{\mathrm{avg}})$ as $\tau \to \infty$. An infimum over $\tau > 0$ is no larger than any limit of the function along τ, which gives the stated inequality.

For the truncated kernel $\exp(\mathrm{sim}/\tau)\mathbb{1}[\mathrm{sim} \ge \delta]$ with $\delta \le 1$, the support $\{v : \mathrm{sim}(\mathbf{s}_u, \mathbf{s}_v) \ge \delta\}$ contains u, and the same argument shows that the $\tau \to \infty$ limit is the uniform average over this support, which is the hard-threshold baseline at δ. Hence the infimum over τ on the truncated path is at most the MSE of hard thresholding at δ. □

The proposition compares the kernel baseline with its own endpoints. It does not claim that the untruncated kernel used in SSA dominates every hard threshold, and the gain at interior τ depends on how well sim tracks value similarity.

D Proof: Value-Irrelevant Fields Increase MSE

Let ϕ be an abstraction that does not use field k, and let $\phi^{+k}(\mathbf{s}) = (\phi(\mathbf{s}), f_k(\mathbf{s}))$ add exact matching on field k. We call field k value-irrelevant given ϕ if $\mathbb{E}[V^\pi(\mathbf{s}) \mid \phi(\mathbf{s}), f_k(\mathbf{s})] = \mathbb{E}[V^\pi(\mathbf{s}) \mid \phi(\mathbf{s})]$, that is, field k carries no information about V^π beyond what ϕ already matches. This is stronger than $\mathrm{Var}(\mathbb{E}[V^\pi \mid f_k]) = 0$, which only rules out marginal dependence.

Proposition 2. *If field k is value-irrelevant given ϕ, then under the assumptions of Eq. 3, adding field k to the matching criterion leaves the grouping bias unchanged and weakly increases the estimation variance, so MSE$(\phi^{+k}) \ge$ MSE(ϕ).*

Proof. By the law of total variance conditioned on $\phi(\mathbf{s})$,

$$\mathbb{E}\big[\mathrm{Var}(V^\pi \mid \phi)\big] = \mathbb{E}\big[\mathrm{Var}(V^\pi \mid \phi, f_k)\big] + \mathbb{E}\big[\mathrm{Var}\big(\mathbb{E}[V^\pi \mid \phi, f_k] \mid \phi\big)\big]. \tag{18}$$

Under value irrelevance the inner conditional mean does not depend on f_k, so the last term is zero and the bias terms of ϕ and ϕ^{+k} coincide. Adding a matching condition can only remove members from a group, so $G_{\phi^{+k}}(\mathbf{s}) \subseteq G_\phi(\mathbf{s})$ and $\sigma^2/|G_{\phi^{+k}}(\mathbf{s})| \ge \sigma^2/|G_\phi(\mathbf{s})|$ for every $\mathbf{s}$. Taking expectations gives the result. □

In SSA the same reasoning applies in soft form: a field with $\alpha_k > 0$ that is value-irrelevant lowers the similarity between steps that differ only in that field, which shifts kernel weight away from them without reducing the bias of the baseline.

E Detailed Theoretical Analysis

E.1 Derivation of the MSE Decomposition (Eq. 3)

We derive the MSE of the group baseline under a state abstraction ϕ. Let $V^\pi(\mathbf{s})$ denote the expected return-to-go from $\mathbf{s}$ under the current policy, and let $\hat{V}_\phi(\mathbf{s})$ average the observed returns over the group $G_\phi(\mathbf{s})$ of steps from the same task with $\phi(\mathbf{s}') = \phi(\mathbf{s})$:

$$\hat{V}_\phi(\mathbf{s}) = \frac{1}{|G_\phi(\mathbf{s})|} \sum_{\mathbf{s}' \in G_\phi(\mathbf{s})} R(\mathbf{s}'). \tag{19}$$

We assume $R(\mathbf{s}') = V^\pi(\mathbf{s}') + \eta(\mathbf{s}')$, where the noise has zero mean and variance σ^2 given the state and is independent across group members (homoscedastic noise; the relaxation is discussed below). Returns-to-go of steps in the same trajectory are correlated in practice, so this is an idealization that isolates the effect of group composition.

Conditioning on the group members, the error at $\mathbf{s}$ splits into a deterministic and a noise part,

$$\hat{V}_\phi(\mathbf{s}) - V^\pi(\mathbf{s}) = \underbrace{\big(\bar{V}_\phi(\mathbf{s}) - V^\pi(\mathbf{s})\big)}_{\text{bias}} + \underbrace{\frac{1}{|G_\phi(\mathbf{s})|} \sum_{\mathbf{s}' \in G_\phi(\mathbf{s})} \eta(\mathbf{s}')}_{\text{noise}}, \tag{20}$$

where $\bar{V}_\phi(\mathbf{s}) = \frac{1}{|G_\phi(\mathbf{s})|} \sum_{\mathbf{s}' \in G_\phi(\mathbf{s})} V^\pi(\mathbf{s}')$. The noise has zero mean given the members, so the cross term vanishes, and its variance is $\sigma^2/|G_\phi(\mathbf{s})|$. For the bias, averaging $(\bar{V}_\phi(\mathbf{s}) - V^\pi(\mathbf{s}))^2$ over the members $\mathbf{s}$ of a group of size n gives the empirical within-group variance of V^π, whose expectation is $(1 - 1/n)\,\mathrm{Var}(V^\pi \mid \phi(\mathbf{s}))$ when the members are distributed as states given their abstract state. Combining the two parts,

$$\mathrm{MSE}(\phi) = \mathbb{E}\left[\Big(1 - \frac{1}{|G_\phi(\mathbf{s})|}\Big)\mathrm{Var}\big(V^\pi \mid \phi(\mathbf{s})\big)\right] + \mathbb{E}\left[\frac{\sigma^2}{|G_\phi(\mathbf{s})|}\right]. \tag{21}$$

Eq. 3 drops the factor $1 - 1/|G_\phi|$, which only lowers the bias of very small groups and does not change the direction of the tradeoff.

Relaxing the homoscedasticity assumption. When the return variance is state-dependent, $\mathrm{Var}(R(\mathbf{s}') \mid \mathbf{s}') = \sigma^2(\mathbf{s}')$, the variance term becomes $\mathbb{E}[\bar{\sigma}^2_\phi/|G_\phi|]$ where $\bar{\sigma}^2_\phi = \frac{1}{|G_\phi|}\sum_{\mathbf{s}' \in G_\phi} \sigma^2(\mathbf{s}')$ is the average within-group noise. The bias term is unchanged. The qualitative conclusions (finer grouping reduces bias but increases variance) remain the same.

E.2 Why ANOVA Identifies Value-Relevant Fields

The field-importance weight $\alpha_k = V_k / \sum_{k'} V_{k'}$ uses the between-group variance $V_k = \mathrm{Var}(\mathbb{E}[R \mid f_k])$ as a proxy for value relevance. We justify this choice by connecting it to the MSE decomposition.

Proposition 3. *Let $\phi_k(\mathbf{s}) = f_k(\mathbf{s})$ group states by field k alone, and assume $\mathbb{E}[\eta \mid \mathbf{s}] = 0$. Then $\mathbb{E}[R \mid f_k] = \mathbb{E}[V^\pi \mid f_k]$, and the grouping bias under ϕ_k satisfies*

$$\mathbb{E}\big[\mathit{Var}(V^\pi \mid \phi_k(\mathbf{s}))\big] = \mathit{Var}(V^\pi) - V_k. \tag{22}$$

Proof. By the tower property, $\mathbb{E}[R \mid f_k] = \mathbb{E}[\mathbb{E}[V^\pi + \eta \mid \mathbf{s}] \mid f_k] = \mathbb{E}[V^\pi \mid f_k]$, so $V_k = \mathrm{Var}(\mathbb{E}[V^\pi \mid f_k])$. The law of total variance gives $\mathrm{Var}(V^\pi) = \mathbb{E}[\mathrm{Var}(V^\pi \mid f_k)] + \mathrm{Var}(\mathbb{E}[V^\pi \mid f_k])$, and substituting yields the result. □

At the population level, ranking fields by V_k is therefore the same as ranking them by how much single-field grouping reduces the grouping bias. Two gaps separate this from what SSA computes. First, SSA uses the plug-in estimate from Eq. 5. For a field with c_k distinct observed values and within-group return variance σ^2_R, standard one-way ANOVA gives $\mathbb{E}[\hat{V}_k] \approx V_k + (c_k - 1)\,\sigma^2_R/|\mathcal{D}|$, so the estimate is inflated for high-cardinality fields when $|\mathcal{D}|$ is small; set fields such as visible objects are the most exposed to this effect, and online re-estimation on fresh batches averages part of it out. Second, V_k is a marginal score and does not account for interactions or redundancy among fields, so it is a heuristic for the multi-field abstraction rather than its optimum.

E.3 The Kernel Baseline as Soft Grouping

Eq. 3 concerns hard groups. The kernel baseline admits the same decomposition with soft weights. Fix the task and a step u, write $V_v = V^\pi(\mathbf{s}_v)$, and treat the field weights $\boldsymbol{\alpha}$ as fixed, for example estimated on an earlier batch, so that the kernel weights depend on the states but not on the returns being averaged.

Proposition 4 (Weighted decomposition)**.** *Under the noise assumptions of Eq. 3, conditioned on the states of $\mathcal{G}(u)$,*

$$\mathbb{E}\big[(B_u - V_u)^2 \mid \textit{states}\big] = \Big(\sum_{v\in\mathcal{G}(u)} w_{uv}\,(V_v - V_u)\Big)^2 + \sigma^2 \sum_{v\in\mathcal{G}(u)} w_{uv}^2. \tag{23}$$

For uniform weights over a hard group G, this reduces to $(\bar{V}_G - V_u)^2 + \sigma^2/|G|$, the per-step form of Eq. 3. The quantity $n_{\mathrm{eff},u} = 1/\sum_v w_{uv}^2$ is the effective group size.

Proof. Given the states, $B_u - V_u = \sum_v w_{uv}(V_v - V_u) + \sum_v w_{uv}\eta_v$ because the weights sum to one. The first sum is fixed, and the second has zero mean and variance $\sigma^2 \sum_v w_{uv}^2$ since the noise terms are independent with variance σ^2. The cross term vanishes in expectation. □

To relate the bias term to the similarity, we need the similarity to be value-aware in the following sense.

Assumption 1 (Value-aware similarity)**.** *There is $L \geq 0$ such that $|V^\pi(\mathbf{s}) - V^\pi(\mathbf{s}')| \leq L\big(1 - \mathrm{sim}(\mathbf{s},\mathbf{s}')\big)$ for all structured states $\mathbf{s},\mathbf{s}'$ of a task.*

The assumption says two things. Steps with $\mathrm{sim} = 1$, which agree on every field with $\alpha_k > 0$, have the same value, so the parsed fields are sufficient for V^π. Away from exact matches, the value changes at most in proportion to the lost similarity. The ANOVA weights aim to make the assumption hold with a small L by assigning little weight to fields along which the value does not change.

Proposition 5 (Temperature tradeoff)**.** *Let $\bar{s}_u(\tau) = \sum_v w_{uv}(\tau)\,\mathrm{sim}(\mathbf{s}_u,\mathbf{s}_v)$ be the mean similarity under the kernel weights. Under Assumption 1 and the conditions of Proposition 4,*

$$\mathbb{E}\big[(B_u(\tau) - V_u)^2 \mid \textit{states}\big] \leq L^2\big(1 - \bar{s}_u(\tau)\big)^2 + \frac{\sigma^2}{n_{\mathrm{eff},u}(\tau)}. \tag{24}$$

Moreover, $\bar{s}_u(\tau)$ is nonincreasing in τ and $n_{\mathrm{eff},u}(\tau)$ is nondecreasing in τ.

Proof. For the bias, $\big|\sum_v w_{uv}(V_v - V_u)\big| \leq \sum_v w_{uv}|V_v - V_u| \leq L\sum_v w_{uv}(1 - \mathrm{sim}(\mathbf{s}_u,\mathbf{s}_v)) = L(1-\bar{s}_u)$, and Proposition 4 gives the bound. For monotonicity, let $\beta = 1/\tau$, $s_v = \mathrm{sim}(\mathbf{s}_u,\mathbf{s}_v)$, and $Z(\beta) = \sum_v e^{\beta s_v}$, so that $w_{uv} = e^{\beta s_v}/Z(\beta)$ and $\bar{s}_u = \mathbb{E}_\beta[s]$. Then $\frac{d}{d\beta}\mathbb{E}_\beta[s] = \mathrm{Var}_\beta(s) \geq 0$, so $\bar{s}_u$ is nondecreasing in β, that is, nonincreasing in τ. For the effective size, $\sum_v w_{uv}^2 = Z(2\beta)/Z(\beta)^2$ and

$$\frac{d}{d\beta}\log\frac{Z(2\beta)}{Z(\beta)^2} = 2\,\mathbb{E}_{2\beta}[s] - 2\,\mathbb{E}_\beta[s] \geq 0, \tag{25}$$

because $\mathbb{E}_\beta[s]$ is nondecreasing in β and $2\beta \geq \beta$. Hence $\sum_v w_{uv}^2$ is nondecreasing in β and $n_{\mathrm{eff},u}$ is nondecreasing in τ. □

Lowering τ therefore tightens the bias bound and shrinks the effective group size, and raising τ does the reverse. At $\tau \to 0$ the bias bound is zero and the variance term is $\sigma^2/|M_u|$, the exact-match estimator of Proposition 1; at $\tau \to \infty$ the variance term is $\sigma^2/|\mathcal{G}(u)|$ and the bias bound is largest. The bound is a guide rather than an exact description: when the similarity tracks value poorly, L is large and the bias bound is loose. In practice, $\tau \in [0.05, 0.2]$ works well in both environments (Appendix G).

E.4 Remark on Gradient Bias

As in GRPO and GiGPO, whose group means include the return of the step being evaluated, the kernel baseline B_u includes u itself and the later steps of the same trajectory, whose states and returns depend on the action $\mathbf{a}_u$. A baseline that depends on the action makes the policy gradient biased in general. The leading effect of the self term is a shrinkage: the coefficient of R_u in $R_u - B_u$ is $1 - w_{uu}$, and w_{uu} is large exactly when $\mathbf{s}_u$ has few similar neighbors, so unusual steps receive a damped rather than a zero step-level signal. Restricting $\mathcal{G}(u)$ to steps from the other rollouts of the task gives a baseline that depends only on $\mathbf{s}_u$ and on independently sampled rollouts, which is action-independent in the usual sense; the normalization by σ_u and ratio clipping remain, as in GRPO. All results in this paper use the estimator as defined in Eq. 9.

E.5 Connection to Classical State Abstraction Theory

Li et al. [2006] order exact state abstractions from finest to coarsest as model irrelevance, Q^π-irrelevance, Q^*-irrelevance, a^*-irrelevance, and π^*-irrelevance; each class is a special case of the next. Classical guarantees concern planning or learning on the abstract MDP, for example that Q^*-irrelevance preserves an optimal policy.

SSA uses abstraction for a narrower purpose. Matching only forms a baseline, while the policy still conditions on the full observation, so an imperfect abstraction affects the bias and variance of the advantage but does not restrict the policy class. The relevant target is therefore value equivalence under the current policy: steps should share a baseline when their values V^π are close. This is implied by Q^π-irrelevance but is weaker than it, and it is not one of the classes above. The field weights approximate this target from rollout data: a field with $V_k \approx 0$ shows no marginal association with returns under π and receives low weight, subject to the caveats of Section E.2. Unlike classical abstraction, SSA does not need the MDP dynamics or a learned value function, and it makes no claim of preserving optimal policies; its justification is the bias-variance tradeoff of the baseline.

F Extended Experimental Details

Training configuration. All experiments follow the GiGPO [Feng et al., 2025] training pipeline with the following hyperparameters:

Table 4: Hyperparameter configuration.

Parameter	Value
SSA-specific	
Kernel temperature (τ)	0.1
Field-importance update interval (E)	10 iterations
Following GiGPO	
Step discount (γ)	0.95
Step-level weight (ω)	1.0
Group size (N)	8
Learning rate	10^{-6}
Clip ratio (ϵ)	0.2
KL coefficient (β)	0.001

Environment details. ALFWorld uses 3,827 training instances across 6 task types (Pick, Look, Clean, Heat, Cool, Pick2) with a maximum of 50 interaction steps, prompt length 2048, response length 512, and reward 10 (success) / -0.1 (invalid action). WebShop has 1.1M products and 12K user instructions, with a maximum of 15 interaction steps, prompt length 4096, response length 512, and reward 10 (success) / -0.1 (invalid action).

Computing details. Following GiGPO [Feng et al., 2025], ALFWorld and WebShop experiments with Qwen2.5-7B-Instruct are run on 4×H100 GPUs for 150 iterations.

Observation examples. To illustrate how SSA decomposes observations, we show concrete examples from each environment.

ALFWorld. A typical observation at step t:

```
You are in the middle of a room. Looking quickly around you, you
see a countertop 1, a cabinet 3, a cabinet 2, a cabinet 1, a drawer
1, a fridge 1, a garbagecan 1, a microwave 1, a sinkbasin 1, a
stoveburner 1.
```

The `info` dictionary additionally provides `admissible_commands`: `{"go to countertop 1", "go to cabinet 3", "go to fridge 1", "open fridge 1", ...}`. The `Parse` function extracts: `location` = "middle of a room" (regex), `admissible_commands` = the full set from `info` (direct), `visible_objects` = {countertop, cabinet, drawer, fridge, garbagecan, microwave,

sinkbasin, stoveburner} (regex). Two observations from the same location but with different admissible commands (e.g., after picking up an object) receive low similarity because the admissible commands field, which has the highest learned weight ($\alpha = 0.62$), differs substantially.

WebShop. A typical observation on a product page:

```
[Back to Search] [< Prev] Page 1 of 3 [Next >]
Brand: OXO, Price: $12.99, Rating: 4.5
Color: [Red] [Blue] [Green] | Size: [Small] [Medium]
```

The `info` dictionary provides `available_actions`: {"click[Red]", "click[Blue]", "click[Green]", "click[Small]", "click[Medium]", "click[Buy Now]", "click[Back to Search]", "click[Next >]"}. The `Parse` function extracts: `page_type` = "product" (keyword), `available_actions` = the full set from `info` (direct), `selected_options` = {} (regex). A search results page and a product page will have low similarity despite potentially overlapping vocabulary, because they differ in both `page_type` and `available_actions`.

Structured field parsers. The field definitions for each environment are specified manually as part of the experimental setup. This is a one-time effort per environment: the practitioner inspects the environment's `info` dictionary to identify available structured fields, and optionally adds lightweight text extraction patterns for fields not directly exposed by the API. Each parser is implemented in $<$100 lines of Python. For ALFWorld, location is extracted via regex matching on "You arrive at" / "You are in" patterns; visible objects are extracted from "you see" clauses; admissible commands are taken directly from `info['admissible_commands']`. For WebShop, page type is detected by keyword matching; available actions come from `info['available_actions']`. We emphasize that SSA does not automate field discovery: the framework's contribution is in formalizing observations as typed field tuples and learning value-aware importance weights over user-specified fields, not in automatically identifying which fields to use. Future work could explore automatic field discovery, for example by prompting an LLM to identify semantically distinct components of an observation string.

Complete field specification. Table 5 provides the full specification of all structured fields used by SSA across both environments, including the extraction source, data type, match function, and semantic role. We classify each field as either *decision-constraining* (directly restricting or determining the agent's available choices at the current step) or *state-describing* (characterizing the environment's appearance without directly constraining the action space). As discussed in Section 5.2, the learned importance weights α_k consistently assign higher values to decision-constraining fields, suggesting that the agent's decision context is more predictive of future returns than the perceptual details of the observation.

Table 5: Complete structured field specification across both environments. "Source" indicates whether the field comes from the environment's `info` dictionary (API) or requires text extraction (Regex). "Role" classifies each field as decision-constraining (DC) or state-describing (SD). α_k values are the learned importance weights at convergence.

Env	Field	Source	Type	Match m_k	Role	α_k
ALFWorld	`admissible_commands`	API	Set⟨str⟩	Jaccard	DC	0.62
	`location`	Regex	Discrete	Exact	DC	0.25
	`visible_objects`	Regex	Set⟨str⟩	Jaccard	SD	0.13
WebShop	`available_actions`	API	Set⟨str⟩	Jaccard	DC	0.55
	`page_type`	Regex	Discrete	Exact	DC	0.32
	`selected_options`	Regex	Set⟨str⟩	Jaccard	SD	0.13

The pattern holds in both environments: decision-constraining fields (admissible commands, available actions, location, page type) receive a combined weight of 0.87 across both environments, while state-describing fields (visible objects, selected options) receive only 0.13. This is consistent with the theoretical expectation from Section 4.1: fields that constrain the agent's action space directly affect the optimal policy (and hence the value function), making them value-relevant in the sense of the MSE decomposition (Eq. 3).

G Additional Analysis

Sensitivity to temperature τ. We vary the kernel temperature $\tau \in \{0.02, 0.05, 0.1, 0.2, 0.5, 1.0\}$ on ALFWorld and WebShop with Qwen2.5-7B-Instruct (Figure 6). Performance changes little across $\tau \in [0.05, 0.2]$, with the best result at $\tau = 0.1$. Very small τ (0.02) concentrates weight too aggressively on near-identical states, approaching hard matching behavior; very large τ (1.0) distributes weight too uniformly, diluting the step-level signal toward the task-level mean.

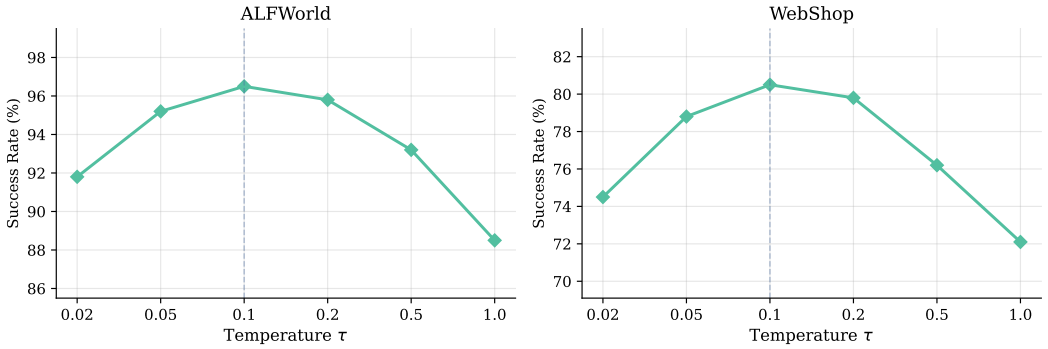

Figure 6: Sensitivity to kernel temperature τ on Qwen2.5-7B-Instruct. Performance peaks at τ=0.1 (dashed line) and changes little across $\tau \in [0.05, 0.2]$.

Field importance dynamics during training. Figure 7 tracks $\boldsymbol{\alpha}$ across training iterations on ALFWorld. Early in training (iterations 1–20), location dominates as the agent learns basic navigation. As training progresses, admissible commands become increasingly important, reflecting the agent's growing sophistication in action selection. By iteration 100, admissible commands receive the highest weight ($\alpha_{\text{cmd}} \approx 0.62$), consistent with the converged values reported in Figure 4. This confirms the value of online $\boldsymbol{\alpha}$ updates: a fixed initial estimate would over-weight location and under-weight admissible commands throughout training.

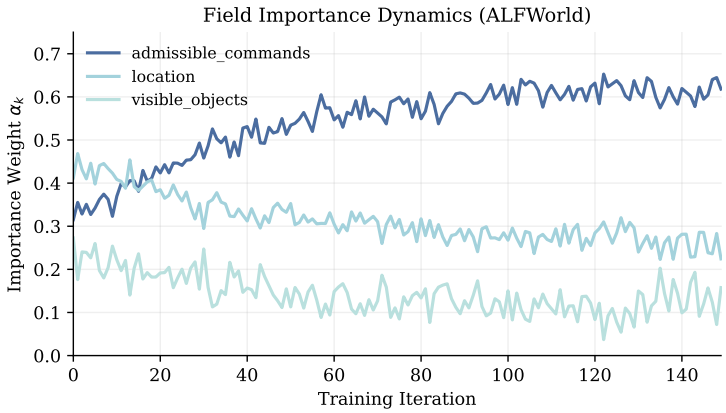

Figure 7: Field importance weights α_k over training iterations on ALFWorld. Admissible commands overtake location as the dominant field around iteration 40.

H Relationship to Existing Methods

SSA places existing methods in one family. The exact matching of GiGPO corresponds to the $\tau \to 0$ limit of Eq. 8 with a single field, the raw observation string, compared by an indicator. HGPO extends the matching criterion with context but retains hard grouping. SSA differs by (1) decomposing observations into typed fields with value-aware importance weights, (2) using typed match functions (indicator for discrete fields, Jaccard for set fields) suited to the semantics of each

field, and (3) applying kernel smoothing over structured similarity rather than text similarity. This framework also connects to broader efforts in LLM agent training via RL [Chen et al., 2025, Wang et al., 2025b, Zhou et al., 2024], where the credit assignment challenge is universal: how to distribute a sparse episode reward across the many actions that constitute a multi-turn interaction. While methods like ArCHer [Zhou et al., 2024] address this through hierarchical RL with learned critics, and VinePPO [Kazemnejad et al., 2025] through Monte Carlo rollouts from intermediate states, SSA takes the complementary approach of improving the state comparison primitive within the group-based framework, which requires no additional models or rollouts.

I On the Cost of Field Definitions

SSA needs environment-specific field definitions, which may seem to limit its use in new environments. We discuss three points.

Most fields require no parsing. The majority of value-relevant fields come directly from the environment's `info` dictionary and need no text extraction. In ALFWorld, the most important field (`admissible_commands`, α=0.62) is returned verbatim by the environment API. A practitioner applying SSA to a new environment can therefore start from the keys of the `info` dictionary and add text-extracted fields only where needed.

The cost is comparable to baselines. Every step-level credit method requires environment-specific configuration. GiGPO requires defining which portion of the observation text to hash as the anchor state, which involves inspecting the observation format and writing extraction logic. HGPO requires specifying the context window size. SSA replaces these choices with a structured field list, in which each field has a named semantic role, a typed match function, and a learned importance weight, so the configuration can be read and checked directly. The total implementation effort for each environment is under 100 lines of Python, comparable to the baseline configuration required by GiGPO.

Automatic field discovery is feasible. Field discovery with an LLM is a natural extension. Given a small sample of observations from a new environment, a language model can be prompted to identify semantically distinct components (e.g., "this observation contains a location, a list of objects, and a set of available actions") and generate extraction code. We leave systematic evaluation of this approach to future work, but the contribution of SSA is the framework of typed fields, learned importance weights, and structured matching, not the specific parsers.

J Limitations and Broader Impact

Limitations. SSA requires environment-specific field definitions to construct structured state representations. While we show in Appendix I that this cost is modest and comparable to baseline configuration, extending SSA to environments without structured `info` dictionaries may require additional parsing effort.

Broader impact. This work proposes a training-time credit assignment method for reinforcement learning agents and does not introduce new model capabilities or deployment-facing systems. We do not foresee direct negative societal impacts arising from this methodological contribution.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Sections 4 and 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Appendix J and Appendix I.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: See Appendices C, D, and E.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code in supplementary; will be open-sourced upon acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 5.1 and Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Results averaged over 3 seeds; std in Table 2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: No human subjects or private data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Appendix J. This is a methodological contribution with no direct negative societal impact.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No model or dataset released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All benchmarks and models cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Code with documentation in supplementary.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs (Qwen2.5-1.5B/7B-Instruct) are the policy models trained and evaluated. Described in §5.1.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.