TrustFlow: Adaptive Trust Calibration for Language Model Guided Reinforcement Learning

Hao Chen[1,2], **Zhiwei Xu**[3], **Hu Fu**[4], **Yuelin Ma**[1,2], **Hangyu Mao**[5], **Yiqun Chen**[6], **Guoliang Fan**[1], **Bin Zhang**[1*]

[1] The Key Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences

[2] School of Artificial Intelligence, University of Chinese Academy of Sciences

[3] Shandong University

[4] Huazhong University of Science and Technology

[5] Institute of Microelectronics, Chinese Academy of Sciences

[6] Renmin University of China

{chenhao2022, mayuelin2023, guoliang.fan, zhangbin}@ia.ac.cn

zhiwei_xu@sdu.edu.cn, fuhu@hust.edu.cn, maohangyu@ime.ac.cn

chenyiqun990321@ruc.edu.cn

Abstract

Reinforcement learning (RL) often suffers from low sample efficiency and inefficient exploration in complex environments. Recent work leverages large language models (LLMs) as sources of prior knowledge for sequential decision-making. However, existing LLM-guided RL approaches typically rely on static or loosely coupled integration, failing to account for the evolving competence of the agent during training. As a result, LLM guidance may become redundant or even detrimental, leading to suboptimal learning dynamics, unnecessary dependence at inference time, and increased latency due to frequent LLM interaction. In this work, we propose TrustFlow, a unified framework that formulates LLM-RL integration as adaptive trust calibration. The key idea is to dynamically regulate the influence of LLM guidance based on learning progress, enabling a gradual transition from prior-driven exploration to LLM-free decision-making. Concretely, a preference representation is first constructed to connect sparse LLM rankings with policy representations. Trust calibration then balances LLM guidance with RL exploration. Building on this trust signal, value alignment is enforced, and a trust-modulated optimization scheme is introduced, incorporating preference structure into the learning process. Empirical results show that TrustFlow consistently improves sample efficiency and enables a smooth transition to fully LLM-free policies, eliminating the need for LLM access at inference time across diverse environments.

1 Introduction

Sequential decision-making lies at the core of many real-world applications, such as robotics [12] and autonomous systems [10]. Reinforcement learning provides a general framework for solving such problems through interaction with the environment. However, RL methods often suffer from inefficient exploration, high sample complexity, and limited generalization, especially in large and complex action spaces [41, 31, 11].

Recent advances in large language models offer a promising direction for addressing these challenges. Through large-scale pretraining, LLMs acquire rich prior knowledge and reasoning capabilities that

*Corresponding author

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

can guide RL agents in complex environments. A growing body of work has explored integrating LLMs into RL pipelines, including reward shaping [1, 38], task decomposition [17, 19], and action space reduction [18], showing improved learning efficiency.

Despite these advances, most existing approaches treat LLM guidance as static, overlooking the evolving competence of the RL agent during training. In practice, the role of external guidance should adapt over time: While strong guidance facilitates early-stage exploration, it may become redundant or even restrictive as the policy improves [15], similar to how a baby walker aids a toddler's first steps but eventually becomes a crutch that hinders independent movement, as illustrated in Figure 1. A fixed integration strategy, therefore, leads to a mismatch between the agent's competence and the level of guidance provided. Moreover, the lack of a principled mechanism to regulate this transition often results in either underutilization of LLM knowledge or persistent over-reliance on it, with many methods still requiring LLM access at inference time [35, 18], leading to increased latency and limited deployment efficiency.

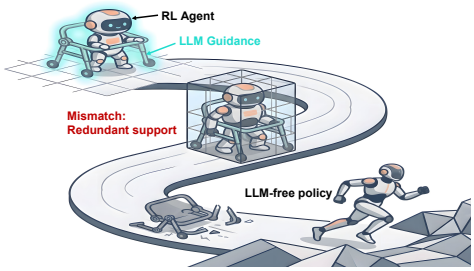


Figure 1: While LLM guidance aids early exploration, static integration becomes suboptimal as the agent improves. We dynamically calibrate LLM influence to enable LLM-free policy execution.

In this work, we propose TrustFlow, a unified framework that dynamically integrates LLM guidance with RL through adaptive trust calibration. Rather than treating LLM outputs as static guidance, TrustFlow models LLM-RL interaction as a time-varying trust modulation process, where the influence of LLM guidance adapts to the agent's evolving learning progress. To enable principled integration, LLM-derived action rankings are transformed into a Boltzmann-Gibbs preference distribution [14], providing a structured prior over the action space and incorporating LLM knowledge at the distribution level rather than through hard action selection. For trust calibration, a dynamic coefficient α balances the RL policy and the LLM prior, enabling a smooth transition from prior-guided exploration to LLM-free execution. For value alignment, a ranking objective is introduced to enforce consistency between Q-value ordering and LLM preferences, allowing the agent to internalize prior knowledge without rigid imitation. For trust-modulated optimization, a projection-based residual modulation scheme adjusts update magnitudes based on the agreement between RL signals and LLM guidance, leveraging useful supervision while maintaining stability under conflict. Overall, TrustFlow enables the agent to progressively internalize LLM-provided prior knowledge and ultimately operate as an independent policy without requiring LLM access at inference time.

Our main contributions are summarized as follows:

1. We propose TrustFlow, a unified framework that formulates the integration of LLMs and RL as an adaptive trust calibration process, enabling dynamic regulation of the influence of LLM guidance throughout training.
2. The framework incorporates multi-level alignment mechanisms, including trust calibration for policy fusion, value alignment via rank-consistent evaluation, and trust-modulated optimization with projection-based update modulation. These mechanisms enable selective and robust knowledge transfer across policy, value, and optimization stages.
3. Extensive experiments demonstrate that TrustFlow enables a smooth transition from LLM-guided learning to LLM-free policy execution, improving sample efficiency through effective trust calibration, while removing the need for LLM access at inference time.

2 Related Work

Both LLMs and RL have been widely applied to decision-making tasks. Some approaches, such as ProAgent [40] and Voyager [32], directly rely on LLMs for decision-making. However, their high inference latency limits applicability in real-time scenarios. Although smaller language models can reduce computational overhead, their limited capacity often degrades performance. In contrast, RL employs lightweight policies that can be executed efficiently, but struggles with complex tasks due

to sparse rewards and large exploration spaces. To bridge these gaps, recent work combines LLMs with RL in several complementary ways. One prominent direction focuses on improving reward design with LLMs. For example, EUREKA [20] and CARD [26] leverage environment source code to enhance state understanding: CARD estimates reward quality via trajectory preference without full training, while EUREKA performs closed-loop optimization through self-reflection and external metrics. Similarly, L2R [39], Text2Reward [34], and Auto MC-Reward [16] refine rewards using natural language, human feedback, and iterative failure signals. Another line of work incorporates LLMs as priors or teachers to guide policy learning. RL-PRIOR [35] improves exploration by pruning the action space, while InstructRL [6] introduces instruction-conditioned priors as regularization in RL objectives. Finally, LLM4Teach [43] adopts knowledge distillation and enables a gradual transition from imitation to independent policy optimization through an annealed weighting scheme. Complementing these RL-specific integrations, the broader field of Knowledge Distillation has also evolved from compressing model depth and hidden representations, as seen in early works like DistilBERT [24] and TinyBERT [8], to distilling higher-level capabilities in the LLM era. Recent advances focus on instruction distillation [29] and reasoning distillation [22], where student models learn to reproduce intermediate reasoning traces and chain-of-thought (CoT) processes [33]. By imitating the teacher's logical steps rather than just final outputs, these approaches significantly enhance the student's problem-solving consistency.

However, these methods overlook the evolving capability of the RL policy and fail to account for its progressive improvement during learning. In particular, when relying on imitation learning or LLM-generated supervision, existing methods often adopt a static supervision paradigm that enforces persistent adherence to external guidance throughout training. Such a fixed dependency fails to reflect the evolving learning dynamics of the agent, limiting its ability to overcome imperfect or noisy priors and preventing a smooth transition toward LLM-free policy learning.

3 Method

In this section, we present TrustFlow, a unified framework for integrating LLM guidance with RL through dynamic trust calibration. Rather than treating LLM feedback as fixed supervision, TrustFlow formulates the interaction between LLM guidance and RL optimization as an adaptive process, allowing the agent to progressively shift from LLM-guided exploration toward LLM-free decision-making. As shown in Figure 2, TrustFlow consists of the following components: Preference Representation, Adaptive Trust Calibration, Preference-Aware Value Alignment, and Trust-Modulated Optimization. Specifically, TrustFlow first transforms LLM-derived rankings into a probabilistic preference prior, then incorporates this prior into the RL policy through trust-aware calibration. It further aligns value estimation with the induced preference structure and adjusts the optimization process according to the consistency between RL learning signals and LLM guidance.

3.1 Problem Statement

We consider sequential decision-making in a text-based open environment $\mathcal{E}$, where each task $\mathcal{T} \in \mathcal{E}$ is formulated as a Partially Observable Markov Decision Process (POMDP) [9]. Specifically, a task is defined as $\mathcal{T} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \mathcal{Z}, \mathcal{O}, \mathcal{G}, \gamma \rangle$. In this language-based setting, each episode begins with a natural language instruction $g \in \mathcal{G}$ that specifies the task objective. The underlying state $s_t \in \mathcal{S}$ is not directly observable; instead, the agent receives textual observations $z_t \in \mathcal{Z}$ generated by the observation function $\mathcal{O}(z_t \mid s_t)$, providing partial descriptions of the environment, and $\mathcal{P}$ is the transition function. At each timestep, the environment exposes a set of admissible actions $\mathcal{A}_{\text{adm},t} \subseteq \mathcal{A}$, corresponding to valid natural language commands executable in the current state. To handle partial observability, the agent conditions its decision on the interaction history $\tau_t = (g, z_0, a_0, \ldots, z_t)$, which encodes both the instruction and past interactions. The goal is to learn a language-grounded policy $\pi(a_t|\tau_t)$ that selects $a_t \in \mathcal{A}_{\text{adm},t}$ to maximize the expected discounted return: $J(\pi) = \mathbb{E}_{\mathcal{T}\sim\mathcal{E},\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$.

3.2 Preference Representation from Ranking

To integrate LLM-derived guidance into reinforcement learning, a structured preference representation is constructed from high-level rankings. Instead of relying on sparse action suggestions or noisy

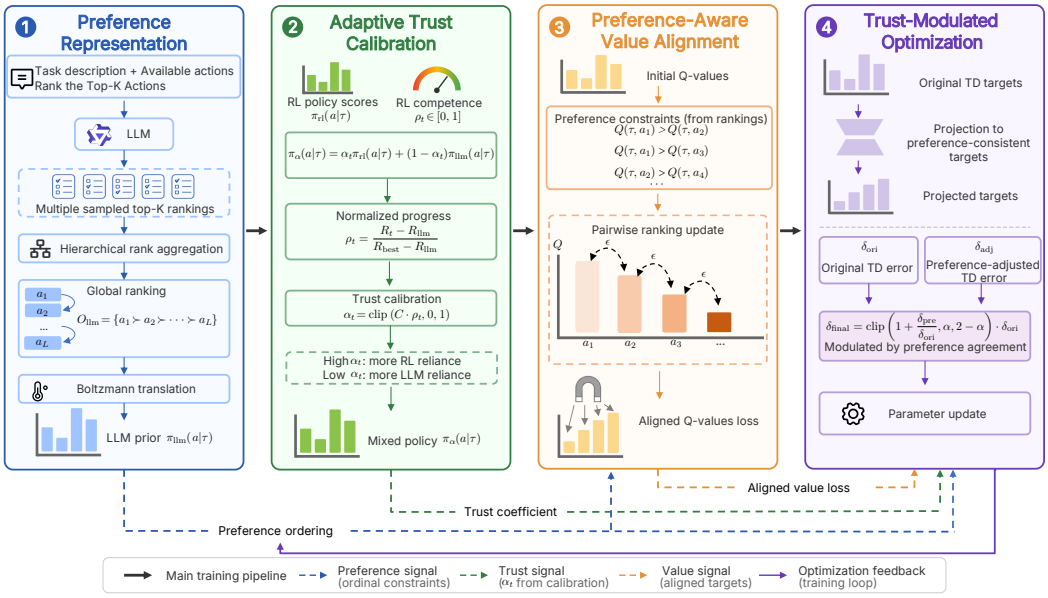


Figure 2: The overall framework of TrustFlow. (1) Preference Representation: bridging sparse LLM rankings and policy representations via probabilistic modeling; (2) Trust Calibration: dynamically balancing LLM guidance and RL exploration; (3) Value Alignment: enforcing consistency between Q-value ordering and preference rankings; (4) Trust-Modulated Optimization: grounding Bellman updates within trusted preference constraints via residual modulation.

scalar scores, preference rankings are treated as pairwise ordering constraints, providing a stable intermediate signal for downstream optimization.

Formally, let $f_{\text{llm}}(\tau, \phi)$ denote the LLM-induced ranking function. Given a trajectory τ and a ranking-oriented prompt ϕ, this function produces a partial ordering over a subset of the action space $\mathcal{A}_{\text{adm}}$. To mitigate generation variance, the LLM is queried N times, each yielding a ranked list of top-K candidate actions. These partial rankings are subsequently aggregated into a consensus ordering via a rank aggregation method similar to the Borda count [4]. The resulting ranking is defined as $\Omega_{\text{llm}} = \{a_1 \succ a_2 \succ \cdots \succ a_L\}$, where $\{a_1, \ldots, a_L\}$ denotes the set of L unique actions identified across the N samples. A detailed description of the aggregation procedure is provided in the Appendix. The value of L adapts to the diversity of LLM suggestions, with $1 \leq L \leq \min(NK, |\mathcal{A}|)$, thereby defining the subset of actions that receive explicit preference supervision. Actions not included in Ω_{llm} are grouped into a low-preference equivalence class, defined as $\mathcal{A}_{\text{tail}} = \mathcal{A}_{\text{adm}} \setminus \{a_1, \ldots, a_L\}$, and are not explicitly ordered among themselves.

To embed this ordinal structure into a utility-based framework, a surrogate utility function is defined:

$$u_{\text{llm}}(a) = \begin{cases} -i & \text{if } a = a_i \in \Omega_{\text{llm}} \\ -L - 1 & \text{if } a \in \mathcal{A}_{\text{tail}} \end{cases}, \tag{1}$$

which treats rank as a linear surrogate of preference strength while assigning all remaining actions to a shared low-preference equivalence class.

Both the LLM prior and the RL policy are then instantiated under a unified Boltzmann formulation:

$$\pi_{\text{llm}}(a|\tau) = \frac{\exp\left(u_{\text{llm}}(a)/\gamma_{llm}\right)}{\sum_{a'} \exp\left(u_{\text{llm}}(a')/\gamma_{llm}\right)}, \quad \pi_{\text{rl}}(a|\tau) = \frac{\exp\left(Q(\tau, a)/\gamma_{rl}\right)}{\sum_{a'} \exp\left(Q(\tau, a')/\gamma_{rl}\right)}, \tag{2}$$

where γ_{rl} and γ_{llm} are temperature parameters controlling the sharpness of the induced distributions, and $Q(\tau, a)$ denotes the action-value function learned by the RL agent.

This formulation can be viewed as an ordinal-to-utility embedding consistent with Plackett-Luce-style models [42], where rankings induce an exponential family distribution while preserving relative preference structure.

To combine these two decision distributions, we introduce a trust-guided fusion policy:

$$\pi_\alpha(a|\tau) = \alpha \cdot \pi_{\text{rl}}(a|\tau) + (1-\alpha) \cdot \pi_{\text{llm}}(a|\tau), \tag{3}$$

where $\alpha \in [0,1]$ is a dynamic trust coefficient that controls the relative contribution of the RL-induced policy. This formulation enables the agent to adaptively balance LLM-informed exploration and reward-driven exploitation, facilitating a smooth transition toward fully LLM-free policy learning.

3.3 Kalai–Smorodinsky-Inspired Calibration for Dynamic Trust

Leveraging the preference representation, we calibrate the trust weight α via a principled heuristic inspired by the Kalai–Smorodinsky (KS) solution [3]. This approach formulates trust allocation as a cooperative bargaining problem that balances alignment and performance utilities, thereby determining the relative reliance on LLM guidance versus RL-driven optimization.

Let $U_{\text{rl}} = \mathbb{E}_{a\sim\pi_{\text{rl}}}[\exp(u_{\text{llm}}(a)/\gamma)]$ and $U_{\text{llm}} = \mathbb{E}_{a\sim\pi_{\text{llm}}}[\exp(u_{\text{llm}}(a)/\gamma)]$ denote the preference utilities for the RL and LLM policies, respectively. The alignment utility is originally defined as $x_1(\alpha) = \mathbb{E}_{a\sim\pi_\alpha}[\exp(u_{\text{llm}}(a)/\gamma)]/U_{\text{llm}}$. To prevent potential suboptimality under perfect policy alignment ($x_1 = 1$), we introduce a relaxation $\lambda \in [0,1)$ and redefine the utility as $x_1(\alpha) = \alpha\frac{U_{\text{rl}}}{U_{\text{llm}}} + (1-\alpha)(1-\lambda)$. This anchors the LLM's contribution to a satisficed baseline, treating the prior as a sufficient reference rather than an absolute truth. The performance utility $x_2(\alpha)$ is empirically approximated by the current episodic return R_t. Let R_{llm} and R_{best} denote the return of the pure LLM policy and the best observed return, respectively.

KS-Inspired Trust Calibration. *Consider a cooperative bargaining game defined by utilities (x_1, x_2), disagreement point $d = (1-\lambda, R_{llm})$, and ideal point $m = (m_1, m_2)$, where $m_1 = \max(1-\lambda, \frac{U_{rl}}{U_{llm}})$ and $m_2 = R_{best}$. Based on the Kalai-Smorodinsky solution that balances normalized utility gains, the calibrated trust α_t^* is given by:*

$$\alpha_t^* = \text{clip}\left(\mathcal{C} \cdot \frac{R_t - R_{llm}}{R_{best} - R_{llm}}, 0, 1\right), \quad \textit{where } \mathcal{C} = \begin{cases} 1, & \frac{U_{rl}}{U_{llm}} > 1-\lambda, \\ 0, & \textit{otherwise}. \end{cases} \tag{4}$$

Detailed derivations and the formulation of the ideal point are provided in the Appendix. This formulation induces a gating effect via m_1. When RL alignment falls below the relaxed baseline, $\mathcal{C}$ vanishes and suppresses trust in RL. Once the threshold is exceeded, $\mathcal{C}$ becomes active and enables the trust weight to scale with performance. To ensure numerical stability near the singularity where the denominator vanishes, we set $\alpha_t^* = 0$ as a neutral fallback. This mechanism facilitates a smooth transition from LLM-guided exploration to RL-dominant optimization while remaining anchored to the preference structure. To stabilize policy updates, we initialize the trust weight at $\alpha_0 = 0.1$, favoring LLM guidance while retaining a small contribution from the RL policy. We employ a warm-up schedule during the initial training phase and apply momentum to the trust weight: $\alpha_t \leftarrow \beta\alpha_{t-1} + (1-\beta)\alpha_t^*$, where $\beta \in [0,1]$ is the momentum coefficient.

3.4 Preference-Aware Value Alignment

While policy-level trust governs action selection, it does not ensure that the Q-function captures the LLM's structural knowledge. This gap is addressed via value-level alignment, which enforces consistency with LLM-derived preference rankings.

Given a partial ordering $\Omega_{\text{llm}} = \{a_1 \succ a_2 \succ \cdots \succ a_L\}$ over a subset of actions, we extend the ordering by assuming that all actions in Ω_{llm} are preferred over actions outside this set. The ranking objective is defined as:

$$L_{\text{rank}}(\theta) = \sum_{a_i \succ a_j} \max\left(0, Q_\theta(\tau, a_j) - Q_\theta(\tau, a_i) + \epsilon\right), \tag{5}$$

where $\succ$ denotes both within-set ordering and the constraint that all actions in Ω_{llm} are preferred over actions in $\mathcal{A}_{\text{tail}}$, θ denotes the parameters of the Q-network, and $\epsilon > 0$ is a margin parameter. This encourages the learned Q-function to respect the LLM-induced ordering while remaining agnostic to absolute reward scaling. The objective is incorporated into the temporal-difference (TD) [27] loss as:

$$L_{\text{total}}(\theta) = L_{\text{TD}}(\theta) + (1-\alpha)\, L_{\text{rank}}(\theta), \tag{6}$$

where α is the trust coefficient, preventing over-commitment to potentially inaccurate LLM priors. The ranking term imposes a soft structural constraint on the value function, encouraging Q-values to respect the LLM-derived preference ordering while allowing deviations supported by reward and TD signals. The TD loss remains the primary learning signal, while the influence of LLM preferences is modulated by α, preventing them from being enforced as hard constraints.

3.5 Trust-Modulated Optimization

A trust-modulated optimization scheme is further introduced to inject the trust signal into temporal-difference learning. Instead of enforcing hard constraints, preference information is incorporated as a soft signal by modulating TD targets and gradient updates according to α, enabling structural guidance to influence learning dynamics while preserving the flexibility of standard TD optimization.

Preference-Grounded Target Construction For a transition (τ, a_t, r, τ'), the standard DQN [21] target $\hat{Q}(\tau, a_t) = r + \gamma \max_{a'} Q_{\text{target}}(\tau', a')$ can become less reliable when rewards are sparse or delayed. To inject preference structure, preference-consistent ordering over next-state action values is enforced. Specifically, we apply order-constrained isotonic projection [7] to $Q_{\text{target}}(\tau', \cdot)$, yielding a preference-consistent value vector $\tilde{Q}(\tau', \cdot)$ obtained by:

$$\tilde{Q}(\tau', \cdot) = \arg \min_{q \in \mathbb{R}^{|\mathcal{A}|}} \frac{1}{2} \sum_{a \in \mathcal{A}} \left(q(a) - Q_{\text{target}}(\tau', a)\right)^2 \quad \text{s.t.} \quad q(a_i) \geq q(a_j), \forall a_i \succ a_j, \tag{7}$$

where $q(a)$ denotes the adjusted action value, and $a_i \succ a_j$ includes both the LLM-provided ordering within Ω_{llm} and the assumption that all $a_i \in \Omega_{\text{llm}}$ are ranked above $a \in \mathcal{A}_{\text{tail}}$. This projection finds the nearest Euclidean vector satisfying Ω_{llm}, reshaping the value landscape to respect preferences with minimal deviation. The TD target is then constructed in a Double DQN [5] manner: $a^* = \arg\max_a Q_\theta(\tau', a), \hat{Q}_{\text{adj}}(\tau, a_t) = r + \gamma \tilde{Q}(\tau', a^*)$. The online network is used for action selection, while the preference-projected target values are used for action evaluation. This construction reduces the effect of preference-inconsistent bootstrap targets and yields a more stable and preference-aware Bellman backup.

Trust-Modulated Residual Scaling We further incorporate preference information into optimization by modulating the TD residual. Let $\delta_{\text{ori}} = \hat{Q}(\tau, a_t) - Q_\theta(\tau, a_t)$ and $\delta_{\text{pre}} = \hat{Q}_{\text{adj}}(\tau, a_t) - Q_\theta(\tau, a_t)$ denote the original and preference-adjusted TD residuals, respectively.

Since the TD gradient is proportional to $\delta \nabla_\theta Q_\theta(\tau, a)$, the residual directly determines the update magnitude. We therefore define a preference-aware scaling of the residual:

$$\delta_{\text{final}} = \text{clip}\left(1 + \frac{\delta_{\text{pre}}}{\delta_{\text{ori}}}, \alpha, 2 - \alpha\right) \cdot \delta_{\text{ori}}. \tag{8}$$

This formulation uses the trust coefficient α to regulate the influence of preference information on the optimization dynamics. The resulting scaled residual replaces the standard TD error in the Bellman update, thereby directly modulating the gradient signal during learning. When preference and TD signals agree, updates are amplified; when they conflict, updates are attenuated, allowing environment feedback to override unreliable priors.

Overall, the proposed framework forms a unified learning pipeline that progresses from LLM-derived rankings to structured preference representation, dynamic trust calibration, value-level alignment, and finally trust-modulated optimization. In this pipeline, preference information is progressively propagated from representation to policy selection, value estimation, and learning dynamics, ensuring consistency across all stages of training.

4 Experiments

4.1 Experimental Setup

To evaluate the effectiveness of our method, we conduct experiments across three text-based environments covering value-based, policy-based, and offline RL settings. We compare against: (1)

Standard RL (conventional methods without LLM guidance); (2) **LLM-PRIOR** (LLMs as standalone decision-makers); and (3) **RL-PRIOR** [35] (incorporating LLM knowledge to guide RL training).

In the value-based setting, we adopt DQN as the backbone and compare with DQN-PRIOR. In the policy-based setting, we build upon the GFLAN [2] framework, using T5-small [23] as the policy model. For offline RL, we consider CQL [13] and BC [30] as standard baselines, along with CQL-PRIOR. We use Qwen2.5-7B [36] as the default backbone for most experiments, and additionally include Qwen3 models [37] of different scales in the offline setting.

A key focus of our evaluation is the agent's ability to operate without LLM assistance at test time. To this end, LLM-assisted and LLM-free performance are explicitly distinguished when applicable. In value-based methods, where LLMs are often used for action pruning, two evaluation settings are reported: a mixed setting (denoted as *Mixed*), where the learned policy is executed with LLM assistance, and a fully LLM-free setting (denoted as *RL-only*), where the agent acts independently without any LLM intervention. In contrast, policy-based and offline RL settings involve LLMs only during training, and all reported results correspond to standalone performance at inference.

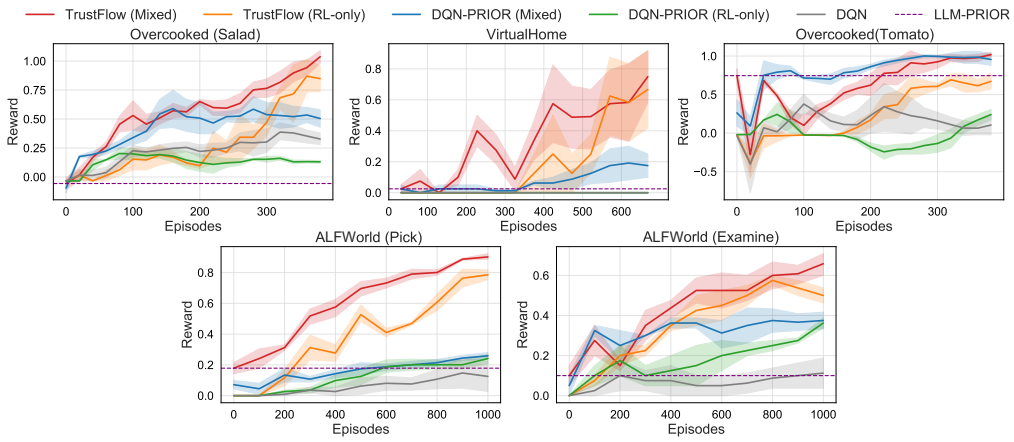


Figure 3: Comparison with value-based baselines. Shaded regions show the 75% confidence interval (CI) across 4 random seeds.

We evaluate our method across three representative environments. **Overcooked** [28] is a partially observable text-based cooking environment with up to 8 macro-actions and dense rewards, evaluated on the *Tomato* (simple) and *Salad* (multi-ingredient coordination) tasks. **ALFWorld** [25] is a text-based environment with up to 50 high-level actions per step and diverse tasks, evaluated on the *pick* and *examine* categories. **VirtualHome** [28] is a household simulation environment with up to 17 actions and sparse rewards, evaluated on the *Entertainment* scenario.

5 Experimental Results and Analysis

Online Performance The online experimental results are shown in Figure 3. Overall, our method consistently outperforms all baselines across diverse tasks, demonstrating improved sample efficiency and more stable convergence. In environments with relatively constrained action spaces, such as *Overcooked Tomato* and *Salad*, DQN-PRIOR is able to learn reasonable policies, but requires substantially more interactions. In contrast, our method reaches strong performance much earlier, benefiting from more effective integration of LLM guidance during training. In more challenging settings with large action spaces and sparse rewards (e.g., *ALFWorld* and *VirtualHome*), baseline methods often struggle to learn effective policies. In contrast, our method remains robust, driven by its trust calibration mechanism that dynamically balances LLM guidance and learned policy behavior. This is achieved through structured preference signals that regularize value estimation, together with consistency-aware updates that modulate optimization dynamics. Beyond assisted performance, our method also leads to stronger LLM-free policies. Across all tasks, the RL-only performance

consistently surpasses competing approaches, and in several cases approaches the performance achieved with LLM assistance. This suggests that the proposed alignment mechanism effectively transfers LLM knowledge into the RL policy, enabling a smooth transition from guided learning to independent decision-making.

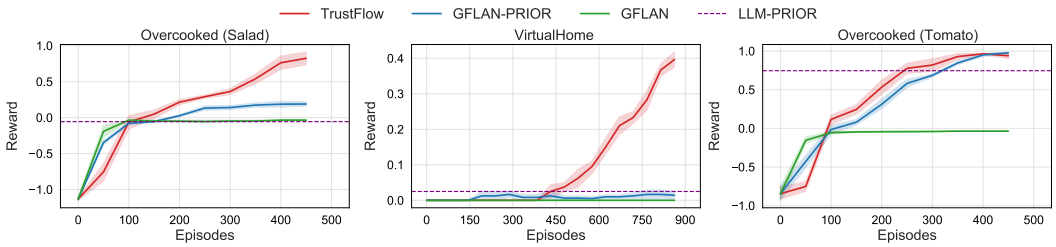


Figure 4: Comparison with policy-based baselines. Shaded regions show 75% CI.

Policy-based and Offline RL Evaluation A similar trend is observed in the policy-based setting. As shown in Figure 4, TrustFlow achieves performance comparable to GFLAN-PRIOR on simpler tasks such as *Tomato*. However, on more complex tasks (e.g., *Salad* and *VirtualHome*), GFLAN-PRIOR exhibits noticeable performance degradation, whereas TrustFlow remains stable, highlighting its advantage in more challenging decision-making scenarios.

Table 1: Offline performance on the *Salad* task using a 2k-transition dataset. We compare TrustFlow with baseline methods across different LLM backbones. All results are reported under standalone evaluation without test-time LLM assistance.

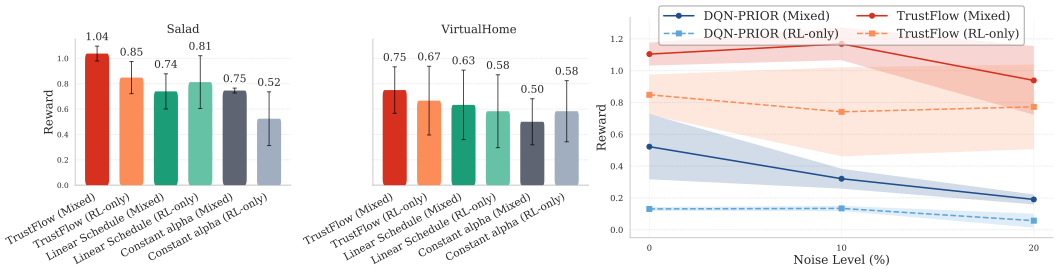
LLM-based Methods			**Standard RL (No LLM)**		
LLM Backbone	**Ours (TrustFlow)**	**CQL-PRIOR**	**DQN**	**BC**	**CQL**
Qwen2.5-3B	0.295 ± 0.008	0.044 ± 0.008	0.194 ± 0.022	0.542 ± 0.299	0.176 ± 0.025
Qwen2.5-7B	1.337 ± 0.000	0.048 ± 0.008			
Qwen3-8B	1.262 ± 0.050	-0.036 ± 0.000			
Qwen3-14B	1.085 ± 0.191	-0.007 ± 0.009			

In the offline RL setting, the results are consistent with our online observations. Table 1 reports performance on the *Salad* task using a 2k-transition dataset. All policies are evaluated independently without LLM assistance at inference time, across different LLM scales (from Qwen2.5-3B to Qwen3-14B). Overall, TrustFlow outperforms baseline methods under most model configurations. Moreover, we find that TrustFlow achieves comparable performance when using 7B–14B LLM backbones to provide preference supervision during training, further demonstrating the robustness and stability of our method. Additional results on the 1k dataset are provided in the Appendix.

Table 2: Test-Time Cost per LLM Invocation and Final Task Performance.

Method	**In Tokens**	**Out Tokens**	**Latency (s)**	**Performance**
TrustFlow	83.17	9.22	0.4006	1.038
DQN Prior	73.82	7.46	0.3724	0.505
TrustFlow (RL-only)	–	–	0.0099	0.848

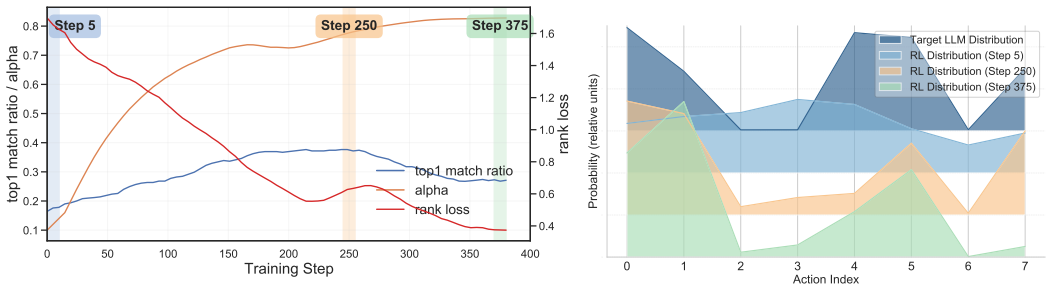
Test-Time Efficiency. As shown in Table 2, TrustFlow incurs an increase of 23% in output token usage and 7.5% in latency, while yielding a 105% performance improvement over DQN-PRIOR. Notably, even without LLM assistance at inference time, TrustFlow (RL-only) still achieves a 67% performance gain, demonstrating the effectiveness of the learned policy.



(a) Ablation study on trust calibration strategies.

(b) Ablation study on noise injection.

Figure 5: Ablation studies on trust calibration and noise robustness. Error bars and shaded regions denote the 75% confidence interval.



(a) Dynamics of core metrics.

(b) Action probability distribution evolution.

Figure 6: Evolution of key alignment metrics.

5.1 Ablation Study

We conduct a series of ablation studies to analyze the key components of TrustFlow, focusing on dynamic trust calibration, robustness to imperfect LLM guidance, and the evolution of value alignment during training. Additional ablations on the ranking objective and the trust-modulated residual update are provided in the Appendix.

Ablation on Dynamic Trust Calibration We first evaluate the role of dynamic trust calibration by comparing our adaptive coefficient with two heuristic alternatives: (i) a fixed coefficient $\alpha = 0.5$, and (ii) a linear schedule where α increases from 0 to 1 over training. As shown in Figure 5a, both heuristic variants consistently underperform TrustFlow. In contrast, our method achieves superior results in both the mixed-policy setting and the LLM-free DQN setting. This result highlights that trust should not be manually specified. Instead, adaptively calibrating the reliance on LLM guidance based on the agent's competence is essential for effective learning.

Robustness to Noisy LLM Priors To assess robustness to imperfect LLM guidance, noise is injected into LLM-generated preference rankings by randomly perturbing the ordering with 10% and 20% noise ratios. TrustFlow is compared against the DQN-PRIOR baseline under identical conditions. As shown in Figure 5b, TrustFlow remains robust under noisy preferences, with slight improvement in the mixed-policy setting at 10% noise. In contrast, DQN-PRIOR (Mixed) degrades significantly, with performance drops of 38% and 63% under 10% and 20% noise, respectively. While TrustFlow exhibits increased variance as noise grows, its average performance remains consistently strong.

Evolution of Value Alignment During Training. We further analyze training dynamics from a value alignment perspective. Figure 6a illustrates the evolution of the trust coefficient α, the ranking loss, and the Top-1 action agreement between RL and LLM policies. In early stages, α remains low, indicating strong reliance on LLM guidance. As training progresses, it gradually increases and stabilizes, reflecting a shift toward RL-driven decision-making. Meanwhile, the ranking loss steadily decreases, suggesting that the value function becomes increasingly aligned with LLM-induced

preferences. This trend is further supported by the distributional evolution shown in Figure 6b. Interestingly, the Top-1 agreement first rises and then declines, revealing a two-stage behavior: the agent initially follows LLM priors and subsequently refines them based on its own experience. This pattern indicates that TrustFlow facilitates selective knowledge transfer rather than rigid imitation.

6 Conclusion

We introduce TrustFlow, a framework for integrating LLM guidance into reinforcement learning that improves adaptability and reduces reliance on LLMs at test time. TrustFlow dynamically adjusts the influence of LLM-derived preferences through an adaptive trust mechanism, balancing external prior knowledge with learned experience. This design enables more effective use of LLM signals during training while allowing the agent to gradually rely less on them as learning progresses. Experiments across value-based, policy-based, and offline RL settings show that TrustFlow consistently improves performance over strong baselines, while remaining fully independent of LLMs at inference time. In future work, we will explore reducing the reliance on repeated LLM-based action ranking by using partial preference signals and improving the efficiency of preference utilization during training.

7 Limitations

Although TrustFlow improves learning efficiency through adaptive trust calibration, it introduces additional training-time overhead due to repeated LLM queries for action ranking, and its performance may degrade when the number of sampled rankings is insufficient. Additionally, the current framework is designed for finite discrete action spaces and has not been evaluated in continuous action spaces or open-ended action generation scenarios. Furthermore, our evaluation is limited to a relatively small set of tasks in text-based environments, and its effectiveness in more diverse decision-making scenarios remains to be investigated.

References

[1] Siddhant Bhambri, Amrita Bhattacharjee, Durgesh Kalwar, Lin Guan, Huan Liu, and Subbarao Kambhampati. Extracting heuristics from large language models for reward shaping in reinforcement learning, 2024. URL https://arxiv.org/abs/2405.15194.

[2] Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[3] G. De Marco and J. Morgan. Kalai-smorodinsky bargaining solution equilibria. *J. Optim. Theory Appl.*, 145(3):429–449, June 2010. ISSN 0022-3239. doi: 10.1007/s10957-010-9697-y. URL https://doi.org/10.1007/s10957-010-9697-y.

[4] Peter Emerson. The original borda count and partial voting. *Social Choice and Welfare*, 40, 02 2013. doi: 10.1007/s00355-011-0603-9.

[5] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2094–2100. AAAI Press, 2016.

[6] Hengyuan Hu and Dorsa Sadigh. Language instructed reinforcement learning for human-ai coordination. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[7] Chu in Charles Lee. The min-max algorithm and isotonic regression. *Annals of Statistics*, 11: 467–477, 1983. URL https://api.semanticscholar.org/CorpusID:120277518.

[8] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online, November 2020. Association for Computational

Linguistics. doi: 10.18653/v1/2020.findings-emnlp.372. URL https://aclanthology.org/2020.findings-emnlp.372/.

[9] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(98)00023-X. URL https://www.sciencedirect.com/science/article/pii/S000437029800023X.

[10] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2022. doi: 10.1109/TITS.2021.3054625.

[11] Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of zero-shot generalisation in deep reinforcement learning. *J. Artif. Int. Res.*, 76, May 2023. ISSN 1076-9757. doi: 10.1613/jair.1.14174. URL https://doi.org/10.1613/jair.1.14174.

[12] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *Int. J. Rob. Res.*, 32(11):1238–1274, September 2013. ISSN 0278-3649. doi: 10.1177/0278364913495721. URL https://doi.org/10.1177/0278364913495721.

[13] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

[14] David A. Lavis. Boltzmann, gibbs, and the concept of equilibrium. *Philosophy of Science*, 75 (5):682–696, 2008. doi: 10.1086/594514.

[15] Dapeng Li, Zhiwei Xu, Bin Zhang, Guangchong Zhou, Zeren Zhang, and Guoliang Fan. From explicit communication to tacit cooperation: A novel paradigm for cooperative marl. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '24, page 2360–2362, Richland, SC, 2024. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9798400704864.

[16] Hao Li, Xue Yang, Zhaokai Wang, Xizhou Zhu, Jie Zhou, Yu Qiao, Xiaogang Wang, Hongsheng Li, Lewei Lu, and Jifeng Dai. Auto MC-Reward: Automated Dense Reward Design with Large Language Models for Minecraft. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16426–16435, Los Alamitos, CA, USA, June 2024. IEEE Computer Society. doi: 10.1109/CVPR52733.2024.01554. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.01554.

[17] Yuchen Liu, Luigi Palmieri, Ilche Georgievski, and Marco Aiello. DELTA: Decomposed efficient long-term robot task planning using large language models. In *2nd CoRL Workshop on Learning Effective Abstractions for Planning*, 2024. URL https://openreview.net/forum?id=O1uIKPpyFI.

[18] Zhihao Liu, Xianliang Yang, Zichuan Liu, Yifan Xia, Wei Jiang, Yuanyu Zhang, Lijuan Li, Guoliang Fan, Lei Song, and Jiang Bian. Knowing what not to do: Leverage language model insights for action space pruning in multi-agent reinforcement learning. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=T49vPTkIt5.

[19] Zhirong Luan, Yujun Lai, Rundong Huang, Shuanghao Bai, Yuedi Zhang, Haoran Zhang, and Qian Wang. Enhancing robot task planning and execution through multi-layer large language models. *Sensors*, 24(5), 2024. ISSN 1424-8220. doi: 10.3390/s24051687. URL https://www.mdpi.com/1424-8220/24/5/1687.

[20] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=IEduRUO55F.

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

[22] Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4, 2023. URL https://arxiv.org/abs/2306.02707.

[23] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), January 2020. ISSN 1532-4435.

[24] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019*, 2019. URL http://arxiv.org/abs/1910.01108.

[25] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, 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.

[26] Shengjie Sun, Runze Liu, Jiafei Lyu, Jing-Wen Yang, Liangpeng Zhang, and Xiu Li. A large language model-driven reward design framework via dynamic feedback for reinforcement learning. *Know.-Based Syst.*, 326(C), September 2025. ISSN 0950-7051. doi: 10.1016/j.knosys.2025.114065. URL https://doi.org/10.1016/j.knosys.2025.114065.

[27] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988. URL https://api.semanticscholar.org/CorpusID:3349598.

[28] Weihao Tan, Wentao Zhang, Shanqi Liu, Longtao Zheng, Xinrun Wang, and Bo An. True knowledge comes from practice: Aligning large language models with embodied environments via reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hILVmJ4Uvu.

[29] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models*, 3(6):7, 2023.

[30] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation, 2018. URL https://arxiv.org/abs/1805.01954.

[31] Andrew Wagenmaker, Kevin Huang, Liyiming Ke, Kevin Jamieson, and Abhishek Gupta. Overcoming the sim-to-real gap: Leveraging simulation to learn to explore for real-world RL. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=JjQl8hXJAS.

[32] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=ehfRiF0R3a.

[33] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

[34] Tianbao Xie, Siheng Zhao, Chen Henry Wu, Yitao Liu, Qian Luo, Victor Zhong, Yanchao Yang, and Tao Yu. Text2reward: Reward shaping with language models for reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tUM39YTRxH.

[35] Xue Yan, Yan Song, Xidong Feng, Mengyue Yang, Haifeng Zhang, Haitham Bou Ammar, and Jun Wang. Efficient reinforcement learning with large language model priors. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=e2NRNQ0sZe.

[36] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[37] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[38] Hu Yifan, Hu Bin-Bin, Yuan Bowen, and Zhang Hai-Tao. Llm coach: Reward shaping for reinforcement learning-based navigation agent. In *2025 Joint International Conference on Automation-Intelligence-Safety (ICAIS) & International Symposium on Autonomous Systems (ISAS)*, pages 1–6, 2025. doi: 10.1109/ICAISISAS64483.2025.11051677.

[39] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montserrat Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, et al. Language to rewards for robotic skill synthesis. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 374–404. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/yu23a.html.

[40] Ceyao Zhang, Kaijie Yang, Siyi Hu, Zihao Wang, Guanghe Li, Yihang Sun, Cheng Zhang, Zhaowei Zhang, Anji Liu, Song-Chun Zhu, et al. Proagent: building proactive cooperative agents with large language models. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i16.29710. URL https://doi.org/10.1609/aaai.v38i16.29710.

[41] Wenbo Zhang and Hengrui Cai. Where to intervene: Action selection in deep reinforcement learning. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=D3au9XkWuy.

[42] Zhibing Zhao, Peter Piech, and Lirong Xia. Learning mixtures of plackett-luce models. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2906–2914, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/zhaob16.html.

[43] Zihao Zhou, Bin Hu, Chenyang Zhao, Pu Zhang, and Bin Liu. Large language model as a policy teacher for training reinforcement learning agents. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24, 2024. ISBN 978-1-956792-04-1. doi: 10.24963/ijcai.2024/627. URL https://doi.org/10.24963/ijcai.2024/627.

A KS-Inspired Derivation of the Trust Weight

In this section, we offer a KS-inspired heuristic for adaptive trust calibration, leveraging the intuition of bargaining theory to provide a principled justification for our reference points. We then present the step-by-step derivation within this framework.

A.1 Formulation of the Reference Points

The Kalai-Smorodinsky (KS) bargaining solution defines an agreement point by equalizing the ratios of maximal attainable gains across objectives. In our setting, the two objectives correspond to Alignment Utility (x_1) and Performance Utility (x_2). While matching the LLM-guided policy distribution yields $x_1(\alpha) = 1$ under the original definition, such distributional alignment does not necessarily lead to optimal downstream performance. We therefore introduce a trust relaxation parameter $\lambda \in [0, 1)$, which effectively reduces the alignment reference level from 1 to $1 - \lambda$, creating a relaxed interpolation anchor for the bargaining process.

- **Disagreement Point** $d = (d_1, d_2) = (1 - \lambda, R_{\textbf{llm}})$**.** The disagreement point encodes the minimal acceptable performance under each objective, serving as the lower bound for the bargaining set.
  - $d_2 = R_{\text{llm}}$ ensures that RL-driven policies are only considered if they do not underperform the zero-shot LLM baseline.
  - $d_1 = 1 - \lambda$ establishes a satisficing threshold for alignment. By shifting the reference from 1 to $1 - \lambda$, we acknowledge that the LLM distribution is a relaxed prior rather than an absolute constraint.
- **Ideal Point** $m = (m_1, m_2) = \left(\max\left(1 - \lambda, \frac{U_{\text{rl}}}{U_{\textbf{llm}}}\right), R_{\textbf{best}}\right)$**.** The ideal point represents the attainable upper bounds of utilities within the current feasible set.
  - $m_2 = R_{\text{best}}$ provides a dynamic normalization based on the highest observed environmental return to date.
  - m_1 is derived from the linear structure of the mixed policy under the relaxed interpolation anchor. Since $\pi_\alpha = \alpha\pi_{\text{rl}} + (1 - \alpha)\pi_{\text{llm}}$, and we treat the LLM's contribution as satisfying the relaxed baseline $1 - \lambda$, the alignment utility is formulated as:

    $$x_1(\alpha) = \alpha \frac{U_{\text{rl}}}{U_{\text{llm}}} + (1 - \alpha)(1 - \lambda).$$

    This formulation linearly interpolates between the relaxed LLM baseline $(1 - \lambda)$ and the RL-induced relative alignment $\frac{U_{\text{rl}}}{U_{\text{llm}}}$.

    To maintain consistency with this anchor, we define $m_1 = \max(1 - \lambda, \frac{U_{\text{rl}}}{U_{\text{llm}}})$. This construction ensures that the bargaining range $[d_1, m_1]$ remains mathematically sound. When the RL policy's alignment $\frac{U_{\text{rl}}}{U_{\text{llm}}}$ is below the threshold, the bargaining space for x_1 collapses to a single point $1 - \lambda$, correctly directing all "bargaining effort" toward performance gains. Conversely, if $\frac{U_{\text{rl}}}{U_{\text{llm}}} > 1 - \lambda$, the ideal point m_1 scales to reward the RL agent for maintaining high preference consistency while improving performance.

A.2 Derivation of the Trust Weight

The KS solution $x^* = (x_1^*, x_2^*)$ is determined by equalizing the normalized gains across both objectives relative to the bargaining boundaries:

$$\frac{x_1(\alpha) - d_1}{m_1 - d_1} = \frac{x_2(\alpha) - d_2}{m_2 - d_2}. \tag{9}$$

Substituting the disagreement point $d = (1 - \lambda, R_{\text{llm}})$ and the ideal point $m = (m_1, R_{\text{best}})$, we obtain the following condition. Since the complete mapping $R(\alpha)$ is generally unavailable a priori during online RL, we approximate the performance utility using the empirically observed return R_t under the current trust weight α:

$$\frac{x_1(\alpha) - (1 - \lambda)}{m_1 - (1 - \lambda)} = \frac{R_t - R_{\text{llm}}}{R_{\text{best}} - R_{\text{llm}}}. \tag{10}$$

To solve for the trust weight α, we expand the alignment utility $x_1(\alpha)$ using the relaxed interpolation anchor. Under this formulation, the mixed policy's alignment is anchored at $1-\lambda$ when $\alpha = 0$:

$$x_1(\alpha) = \alpha \frac{U_{\text{rl}}}{U_{\text{llm}}} + (1-\alpha)(1-\lambda). \tag{11}$$

Subtracting the disagreement level $d_1 = 1-\lambda$ from both sides, we isolate the alignment gain as a linear function of α:

$$\begin{aligned} x_1(\alpha) - (1-\lambda) &= \left[\alpha \frac{U_{\text{rl}}}{U_{\text{llm}}} + (1-\lambda) - \alpha(1-\lambda)\right] - (1-\lambda) \\ &= \alpha \left(\frac{U_{\text{rl}}}{U_{\text{llm}}} - (1-\lambda)\right). \end{aligned} \tag{12}$$

Substituting this linear expansion back into the KS condition (10) yields:

$$\frac{\alpha \left(\frac{U_{\text{rl}}}{U_{\text{llm}}} - (1-\lambda)\right)}{m_1 - (1-\lambda)} = \frac{R_t - R_{\text{llm}}}{R_{\text{best}} - R_{\text{llm}}}. \tag{13}$$

To relate the environmental progress to the intrinsic alignment capacity of the RL policy, we define the scaling coefficient $\mathcal{C}$ as the ratio between the attainable bargaining range and the current policy's alignment gap:

$$\mathcal{C} = \frac{m_1 - (1-\lambda)}{\frac{U_{\text{rl}}}{U_{\text{llm}}} - (1-\lambda)}. \tag{14}$$

Isolating α finalizes the calibrated trust weight for the RL agent:

$$\alpha_t^* = \text{clip}\left(\mathcal{C} \cdot \frac{R_t - R_{\text{llm}}}{R_{\text{best}} - R_{\text{llm}}}, 0, 1\right). \tag{15}$$

Boundary Cases: When evaluating the calibrated trust weight α_t^*, we handle edge cases under a natural veto mechanism to guarantee physical consistency:

1. **Alignment Gate:** If $\frac{U_{\text{rl}}}{U_{\text{llm}}} \leq 1-\lambda$, the RL policy does not exceed the relaxed alignment baseline, and the alignment bargaining range degenerates with $m_1 = 1-\lambda$. We therefore set $\mathcal{C} = 0$ to withhold trust in RL, yielding $\alpha_t^* = 0$.
2. **Performance Stagnation:** If $R_t \leq R_{\text{llm}}$, the performance utility provides no gain relative to the LLM baseline. The calculated raw weight satisfies $\alpha_t^* \leq 0$, which is clipped to 0.
3. **Degenerate Performance Interval:** If $R_{\text{best}} = R_{\text{llm}}$, the performance normalization interval vanishes, making the normalized performance gain undefined. In this case, we set $\alpha_t^* = 0$ as a conservative fallback.

B Effect of Ranking Loss and Assisted Update

We further ablate the ranking loss and the assisted update mechanism on the *Salad* and *VirtualHome* tasks.

As shown in Figures 7 and 8, removing either component leads to consistent performance degradation. Both the ranking loss and the assisted update improve performance under the mixed-policy setting as well as in the standalone RL-only setting.

C Additional Results on 1k Datasets

To further validate the scalability of TrustFlow with respect to dataset size, we present additional results on the *Salad* task using a 1k-transition dataset. These experiments complement the main results reported on the 2k dataset in the main text. As shown in Table 3, we observe a similar trend: performance improves as the model size increases. This behavior aligns with our observations on the 2k dataset. We attribute this improvement to the higher-quality preference signals provided by larger LLMs, which offer more accurate guidance and thus better facilitate policy learning. We also observe that the 8B model performs worse than the 7B model in this setting, which we conjecture is due to the increased randomness and variance introduced by the relatively small dataset size.

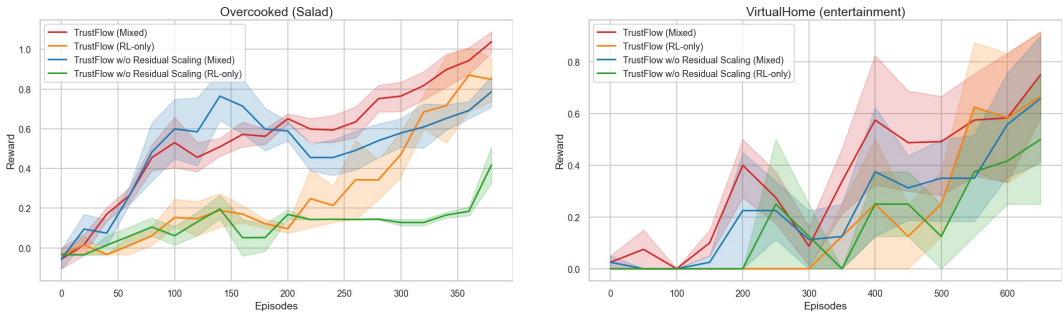


Figure 7: Ablation Studies on Residual Scaling.

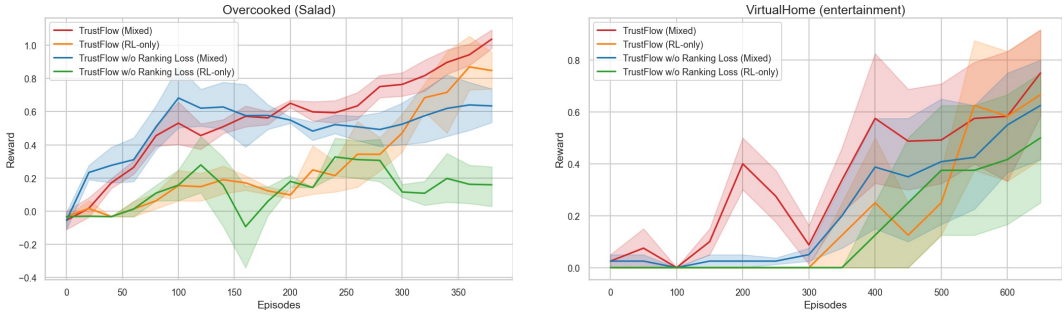


Figure 8: Ablation Studies on Ranking Loss.

D Additional Experimental Analysis

D.1 Ablation Study on Trust-Modulated Optimization

To investigate the contributions of the two components in the Trust-Modulated Optimization module, namely preference-grounded target construction and TD residual scaling, we conduct an ablation study by progressively removing these components. Specifically, we compare three configurations: (1) standard TD learning without Trust-Modulated Optimization, (2) removing preference projection and directly setting the adjusted TD residual to its upper bound, and (3) the full TrustFlow method. The results are reported in Table 5.

The results show that removing either component leads to a performance drop compared with the full TrustFlow configuration. In particular, retaining TD residual scaling without preference-grounded target construction yields only a limited improvement over standard TD learning. This indicates that residual scaling alone is insufficient to fully exploit the LLM-provided preferences. Incorporating preference-grounded target construction further improves performance, suggesting that aligning value targets with the LLM preference order provides complementary benefits to trust-modulated residual scaling. Overall, the two components work together to improve value learning under LLM guidance.

D.2 Sensitivity Analysis of Preference Representation

We investigate the sensitivity of TrustFlow to three hyperparameters governing preference representation: the number of Top-K actions returned by each LLM query (K), the number of sampled rankings (N), and the resulting aggregated ranking length (L). We also report the average time per LLM call to examine the computational cost associated with different configurations.

Sensitivity to Top-K. We first vary K while fixing $N = 5$. The results are presented in Table 6.

TrustFlow achieves comparable performance with $K = 1$ and $K = 2$, indicating that its performance is relatively stable within this range. As K increases further, performance tends to decrease. A larger K expands the aggregated preference set but may also introduce less confident actions into the ranking, resulting in noisier preference supervision. Meanwhile, the average time per LLM call

Table 3: Offline performance on the *Salad* task using a 1k-transition dataset. We compare TrustFlow with baseline methods across different LLM backbones. All results are reported under standalone evaluation without test-time LLM assistance.

LLM-based Methods			**Standard RL (No LLM)**		
LLM Backbone	**Ours**	**CQL-PRIOR**	**DQN**	**BC**	**CQL**
Qwen2.5-3B	0.294 ± 0.009	0.007 ± 0.009	0.144 ± 0.011	0.153 ± 0.000	0.176 ± 0.025
Qwen2.5-7B	0.830 ± 0.235	-0.059 ± 0.001			
Qwen3-8B	0.454 ± 0.351	0.061 ± 0.009			
Qwen3-14B	1.081 ± 0.075	0.010 ± 0.009			

Table 4: Maximum horizon for each environment.

Environment	**Horizon**
ALFWorld (Pick)	60
ALFWorld (Examine)	60
Overcooked (Tomato)	30
Overcooked (Salad)	30
VirtualHome	15

increases with K. These results suggest that a relatively small Top-K value provides a favorable balance between preference coverage, supervision quality, and computational cost.

Sensitivity to the Number of Samples (N). We further vary the number of sampled rankings while fixing $K = 2$. The results are shown in Table 7.

Reducing the number of sampled rankings from $N = 5$ to $N = 2$ leads to a substantial performance drop. With fewer samples, the aggregated ranking becomes shorter and potentially less representative, resulting in less reliable preference supervision. In contrast, increasing N from 5 to 8 yields comparable performance, suggesting diminishing returns beyond a moderate sampling budget. Although a larger N provides a longer aggregated ranking, it also incurs additional computational cost. Overall, these results indicate that a moderate sampling budget, such as $N = 5$, is sufficient to obtain effective preference representations without introducing unnecessary query overhead.

E Environment Details

The maximum horizon for each environment is shown in Table 4.

We briefly describe the environments used in our experiments as follows:
Overcooked. Overcooked is a partially observable, text-based kitchen environment in which the agent receives the positions of visible objects at each time step and must execute a sequence of actions to complete cooking tasks. We consider two tasks for evaluation. In *Overcooked (Tomato)*, the agent prepares and delivers a dish consisting of chopped tomatoes. In *Overcooked (Salad)*, the task is more complex, requiring the preparation and delivery of a salad made from chopped tomatoes and lettuce.

The agent operates over a set of up to eight macro-actions, including movement, picking and placing ingredients, chopping, and delivering. The environment provides dense shaping rewards, including positive rewards for successful subgoals (e.g., chopping) and penalties for incorrect deliveries and time steps, in addition to a terminal reward upon successful completion.

Table 5: Ablation study on Trust-Modulated Optimization.

Method	TD Residual	Preference Projection	Residual Scaling	Salad (Mixed / RL-only)
Standard TD (w/o Trust-Modulated Optimization)	δ_{ori}	×	×	0.787 ± 0.130 / 0.418 ± 0.176
w/o Preference Projection	δ_{ori} (clipped to upper bound)	×	✓	0.823 ± 0.304 / 0.537 ± 0.109
Full TrustFlow	δ_{final}	✓	✓	1.038 ± 0.102 / 0.849 ± 0.220

Table 6: Sensitivity to the number of Top-K actions.

K	Avg. L	TrustFlow	TrustFlow (RL-only)	Time per Call (s)
1	3.87	1.014 ± 0.157	0.913 ± 0.162	0.2871
2	4.71	1.038 ± 0.102	0.849 ± 0.220	0.4006
3	5.68	0.888 ± 0.446	0.743 ± 0.540	0.5556
4	6.48	0.853 ± 0.174	0.812 ± 0.362	0.6975

For Overcooked (Salad)

Input: Your task is to serve the dish of a bowl only containing chopped tomato and lettuce. Current Observation: There are two fixed cutting boards in the room. You notice a tomato, a lettuce and an onion on the different tables. Currently you don't have anything in hand.

Admissible Actions: 'pick up the lettuce', 'take the empty bowl', 'pick up the onion', 'walk to the first cutting board', 'walk to the second cutting board', 'serve nothing', 'pick up the tomato', 'chop nothing'

ALFWorld. We focus on the text-based component of ALFWorld, where the action space consists of high-level commands (e.g., "go to a room"). Although the set of admissible actions is finite, it can be relatively large, with up to 50 possible actions at each step. We consider two task categories: *ALFWorld (Pick)* and *ALFWorld (Examine)*, selecting 28 and 10 tasks, respectively.

Tasks include instructions such as "put the cellphone on the armchair" or "examine the laptop with the desk lamp." The environment provides sparse rewards, with a reward of 1.0 only upon successful completion of the task and no intermediate rewards.

For ALFWorld (Examine) from Qwen2.5-7B

Input: Task: Your task is to: examine the book with the desklamp. Current observation:You are in the middle of a room. Looking quickly around you, you see a bed 1, a drawer 10, a drawer 9, a drawer 8, a drawer 7, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a dresser 1, a garbagecan 1, a shelf 7, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, and a shelf 1.

Admissible Actions: go to drawer 5, go to shelf 3, go to drawer 7, go to drawer 3, inventory, go to drawer 9, go to drawer 10, go to bed 1, go to drawer 8, go to shelf 7, go to dresser 1, go to shelf 5, go to shelf 6, go to drawer 6, go to drawer 4, go to drawer 2, go to shelf 4, go to drawer 1, go to garbagecan 1, go to shelf 1, look, go to shelf 2.

VirtualHome. VirtualHome simulates a realistic household environment with multiple rooms and diverse interactive objects. In *VirtualHome (Entertainment)*, the agent is required to retrieve items (e.g., milk and chips) from the kitchen, bring them to the living room, and then execute a sequence of actions such as turning on the TV, sitting on the sofa, and watching television.

The action space includes up to 17 possible actions. The environment provides sparse rewards, granting a reward of +1 only upon successful completion of the entire task. This sparse reward structure makes the task significantly more challenging for learning.

Table 7: Sensitivity to the number of sampled rankings.

N	Avg. L	TrustFlow	TrustFlow (RL-only)	Time per Call (s)
2	3.26	0.767 ± 0.179	0.476 ± 0.142	0.3005
5	4.71	1.038 ± 0.102	0.849 ± 0.220	0.4006
8	5.97	1.009 ± 0.223	0.836 ± 0.403	0.4610

For VirtualHome (Entertainment) from Qwen2.5-7B

Input: Your task is to enjoy the chips and the milk while watching TV. Current Observation: There are four rooms: the kitchen, bathroom, bedroom, and living room. You are in the kitchen and notice chips and milk. But they are not close to you. Currently, you are not grabbing anything in hand. **Admissible Actions:** 'walk to the bathroom', 'walk to the living room', 'walk to the bedroom', 'walk to the chips', 'walk to the milk'.

F Implementation Details

Our method runs on NVIDIA 80G A100 GPU and Intel(R) Xeon(R) Gold 5218 CPU, CUDA 12.8, PyTorch 2.1.2. The hyperparameters in the training of TrustFlow are shown in Table 8.

Table 8: The hyperparameters of TrustFlow.

Parameter	**TrustFlow**	**TrustFlow (GFLAN)**	**TrustFlow (CQL)**
Learning rate	5e-4	1e-6	5e-4
Batch size	128	64	128
LLM temperature	0.5	0.5	0.5
DQN temperature	0.02 (Salad) 0.01 (others)	–	0.02
Momentum	0.9	0.9	0.9
Margin parameter	1e-4	–	1e-4
Trust relaxation	0.8	0.8	–
Epochs per iteration	40	4	–

Here we provide an example prompt, where the model is asked to select the top 2 actions. Since the model needs to generate a ranked list of multiple actions, this increases the generation time and consequently prolongs the overall training process. To mitigate this issue, we simplify the output by having the model produce a ranking of action IDs instead of full action descriptions.

For ALFWorld (Examine) from Qwen2.5-7B

Input: Task: Your task is to: examine the book with the desklamp. Current observation: You are in the middle of a room. Looking quickly around you, you see a bed 1...
You are allowed to take the following actions: A0: go to drawer 5, A1: go to shelf 3, A2: go to drawer 7, A3: go to drawer 3, A4: inventory, A5: go to drawer 9, A6: go to drawer 10, A7: go to bed 1, A8: go to drawer 8, A9: go to shelf 7, A10: go to dresser 1, A11: go to shelf 5, A12: go to shelf 6, A13: go to drawer 6, A14: go to drawer 4, A15: go to drawer 2, A16: go to shelf 4, A17: go to drawer 1, A18: go to garbagecan 1, A19: go to shelf 1, A20: look, A21: go to shelf 2.
Please select the top 2 actions from the admissible actions by their IDs (A0, A1, etc.). List only the action IDs in order of preference, without any additional details or names. For example: Best Actions: 1. 'A0', 2. 'A1'. Best Actions:
Output: 1. A17, 2. A11
1. A0, 2. A19
1. A4, 2. A0
1. A0, 2. A5
1. A11, 2. A8

Hierarchical Rank Aggregation (HRA). To construct a stable global ranking from multiple preference samples, a Hierarchical Rank Aggregation (HRA) strategy is adopted. Let $\mathcal{A}$ denote the candidate action set. For each action $a \in \mathcal{A}_{adm}$, N independent preference samples are collected, each producing a ranked action list. The ranking behavior of a is summarized by a frequency vector:

$$V_a = [c_1, c_2, \ldots, c_L], \tag{16}$$

where c_i denotes the number of times action a is ranked at position i across the N samples, and L is the maximum rank considered.

Actions are then ordered through lexicographic comparison of their ranking profiles under a highest-rank-first principle. Specifically, let $i_{\min}(a) = \min\{i \mid c_i > 0\}$ denote the best observed rank of action a. If $i_{\min}(a) < i_{\min}(b)$, then $a \succ b$. For actions sharing the same best rank i^*, the corresponding frequencies c_{i^*} are compared, with larger values indicating higher preference. Remaining ties are resolved recursively by comparing lower-priority frequencies $c_{i^*+1}, c_{i^*+2}, \ldots$ until a difference is found.

Offline Training In the offline setting, we do not have access to environment interaction and thus cannot compute the calibrated trust weight α_t^*. To address this issue, we replace the adaptive trust mechanism with a simple linear schedule that gradually increases reliance on the learned policy. Specifically, we set

$$\alpha_t = \frac{t}{T},$$

where T is the total number of training steps, so that α_t increases linearly from 0 to 1. This provides a smooth transition from the LLM prior to the learned policy in the absence of online performance feedback.

Pseudocode The pseudocode of TrustFlow is presented in Algorithm 1.

Algorithm 1: Algorithm of TrustFlow

Input: Task T; frozen LLM ranking function $f_{\text{llm}}(\cdot, \phi)$; Q-network Q_θ and target $\hat{Q}$; replay buffer $\mathcal{B}$; trust coefficient α.
Initialization
Initialize Q_θ; $\hat{Q} \leftarrow Q_\theta$; $\mathcal{B} \leftarrow \emptyset$; $\alpha \leftarrow \alpha_{\text{init}}$; $R_{\text{best}} \leftarrow -\infty$; $R_{\text{worst}} \leftarrow +\infty$
// The LLM is fixed ⇒ its reference return is computed once
$R_{\text{LLM}} \leftarrow$ mean episodic return of 5 rollouts of the LLM-only policy
Main Training Loop
for iteration $e = 1, 2, \ldots$ **do**
  // (A) Preference Representation + Rollout (§3.2)
  Reset task T
  for step $t = 0, 1, \ldots$ until done **do**
    Obtain admissible actions $\mathcal{A}_{\text{adm},t}$
    // Collect N Top-K rankings and aggregate
    Query $f_{\text{llm}}(\tau_t, \phi, N) \to N$ Top-K rankings over $\mathcal{A}_{\text{adm},t}$
    Aggregate the N rankings $\to \Omega_{llm} = \langle a_1 \succ a_2 \succ \cdots \succ a_L \rangle$ *// L unique actions*
    // Ordinal ranking → utility → Boltzmann LLM policy
    $u_{\text{llm}}(a_i) \leftarrow$ rank-derived utility, $a_i \in \Omega_{llm}$
    $\pi_{\text{llm}}(a|\tau_t) \propto \exp(u_{\text{llm}}(a))$; unranked $a \in \mathcal{A}_{\text{adm},t}$ share a low-utility tail
    // RL policy and trust-calibrated fusion
    $\pi_{\text{rl}}(a|\tau_t) \leftarrow \text{softmax}(Q_\theta(\tau_t, \cdot))$
    $\pi_\alpha(a|\tau_t) \leftarrow$ fusion of $\pi_{\text{llm}}(\cdot|\tau_t)$ and $\pi_{\text{rl}}(\cdot|\tau_t)$ with weight α *// §3.3*
    // Act
    $a_t \sim \pi_\alpha(\cdot|\tau_t)$
    Execute a_t, observe $r_t, \tau_{t+1}, \text{done}$
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{(\tau_t, a_t, r_t, \tau_{t+1}, \text{done}, \pi_{\text{llm}}(\cdot|\tau_t), \pi_{\text{llm}}(\cdot|\tau_{t+1}))\}$
  end for
  $R \leftarrow$ episodic return; $R_{\text{best}} \leftarrow \max(R_{\text{best}}, R)$; $R_{\text{worst}} \leftarrow \min(R_{\text{worst}}, R)$
  // (B) Adaptive Trust Calibration (§3.3)
  $\alpha^* \leftarrow$ calibrated trust weight
  $\alpha \leftarrow \beta \cdot \alpha + (1 - \beta) \cdot \alpha^*$ *// Momentum-smoothed*

// (C) Preference-Aware Value Alignment & Trust-Modulated Optimization (§3.4–3.5)

for gradient step $j = 1, \ldots, J$ **do**

Sample mini-batch from $\mathcal{B}$

// Preference-Grounded Target Construction

$\hat{Q}(\tau', \cdot) \leftarrow$ target value

$\widetilde{Q}(\tau', \cdot) \leftarrow$ projection of $\hat{Q}(\tau', \cdot)$ onto the preference constraints defined by $\Omega_{\text{llm}}(\tau')$

$a' \leftarrow \arg\max_a Q_\theta(\tau', a)$

// TD Residual Scaling

$\delta_{\text{pre}} \leftarrow r + \gamma \cdot \widetilde{Q}(\tau', a') - Q_\theta(\tau, a)$

$\delta_{\text{ori}} \leftarrow r + \gamma \cdot \hat{Q}(\tau', a') - Q_\theta(\tau, a)$

$\delta_{\text{final}} \leftarrow \text{scale}(\delta_{\text{ori}}, \alpha, \delta_{\text{pre}})$ *// TD Residual Scaling*

// Losses

$\mathcal{L}_{\text{TD}}(\theta) \leftarrow \frac{1}{2}(\delta_{\text{final}})^2$ *// Trust-modulated TD loss*

$\mathcal{L}_{\text{Rank}}(\theta) \leftarrow \sum_{a_i \succ a_j} \max(0, Q_\theta(\tau, a_j) - Q_\theta(\tau, a_i) + \epsilon)$

// Preference-Aware Value Alignment (ranking loss), margin ϵ

$\mathcal{L}_{\text{total}}(\theta) \leftarrow \mathcal{L}_{\text{TD}}(\theta) + (1 - \alpha) \cdot \mathcal{L}_{\text{Rank}}(\theta)$

$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\text{total}}(\theta)$

end for

end for

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: The Methodology and Experiments sections align with the claims made in the Abstract and Introduction.

   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: The limitation is discussed in Section 7.

   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: [N/A]

Justification: The paper does not include theoretical results

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: The implementation details are described in the Appendix.

   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: [No]

Justification: The code and experimental results will be released once they are fully organized.

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: In the Experiments section, we provide a detailed account of our experimental procedures, testing methodologies, and model employed; furthermore, additional implementation details are included in the Appendix.

   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: Multiple sets of random seeds are considered in the Experiments section.

   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: We specify the hardware and software environments used in the Appendix.

   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: Our research fully complies with the NeurIPS Code of Ethics.

   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: [N/A]

    Justification: This research remains centered at the algorithmic level; consequently, we do not foresee any immediate or significant societal implications.

    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: The paper poses no such risks.

    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: We cite the original paper that produced the code and environment.

    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: [N/A]

    Justification: This paper does not introduce or release any new assets, such as datasets or models.

    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: The paper does not involve crowdsourcing nor research with human subjects.

    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: This paper does not involve any human subjects, biological samples, or clinical trials.

    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: We have explicitly detailed the specific LLM versions used as priors within our experimental setup, while the corresponding prompts are fully disclosed in the Appendix for reproducibility.

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.