OCPO: OBSERVED CONTINUATION POLICY OPTIMIZATION FOR SOCIAL AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Social intelligence simulation challenges language agents to navigate open-ended, multi-turn interactions while balancing social goals and interpersonal relationships. However, training signals are typically available only at the dialogue level, making credit assignment to individual utterances difficult and resulting in sparse, delayed supervision. Existing methods often obtain finer-grained feedback through additional dialogue rollouts, external evaluation, or separately trained reward models, making supervision costly to acquire. To address this, we propose Observed Continuation Policy Optimization (OCPO), a framework that turns immediate partner responses recorded in high-quality dialogue trajectories into local policy supervision. OCPO scores policy-sampled utterances by their compatibility with the same observed partner continuation. It uses sensitivity-based token weighting to emphasize tokens sensitive to candidate utterances and mean-only advantage centering to preserve reward-difference magnitudes in sequence-level updates. By reusing recorded continuations, OCPO acquires rewards without additional full-dialogue rollouts, a separately trained reward model, or external evaluator calls during policy optimization. Experiments on the benchmark social intelligence environment demonstrate that OCPO consistently outperforms large reasoning models and existing optimization methods. Notably, OCPO cuts GPU-hours by 65% compared with previous social-policy optimization pipelines, offering an effective and cost-efficient paradigm for building social agents.

1 INTRODUCTION

Social intelligence, the capacity to navigate evolving interpersonal interactions, infer others' intentions, and pursue social goals, has emerged as a core capability for language agents operating in open-ended environments (Zhou et al., 2024). Large language models (LLMs) enable the study of these capabilities through social simulation, where agents engage in open-ended, multi-turn interactions (Park et al., 2023). Benchmarks such as SOTOPIA place agents in scenarios with distinct profiles, private goals, and social relationships, evaluating their behavior across multiple dimensions (Zhou et al., 2024). In these environments, effective interaction depends on learning which utterances help an agent navigate the evolving social context.

While dialogue-level evaluation measures the success of a complete interaction, a social policy must choose an utterance at every turn. This distinction matters for learning because a final outcome reflects a sequence of decisions and partner responses, making it difficult to identify the contribution of an individual utterance. Moreover, open-ended social interactions admit multiple reasonable responses, making it difficult to specify a directly verifiable reward for each decision (Zhou et al., 2024; Kong et al., 2025). Providing informative local supervision is therefore a central challenge in learning socially effective behavior.

To obtain such supervision, existing methods learn from expert and agent-generated dialogues (Wang et al., 2024), construct segment-level preferences through additional continuations and evaluation (Kong et al., 2025), or train reward models from utterance-level attribution of dialogue outcomes (Yu et al., 2025; Feng et al., 2026). These approaches improve the supervision available for social policy learning, but generating alternative interactions, evaluating them, and preparing reward models introduce additional costs. Meanwhile, recorded dialogues already contain the partner responses that follow each utterance. Prior work on implicit feedback and follow-up likelihood

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

suggests that these responses provide useful information for dialogue learning and evaluation (Pang et al., 2024; De Bruyn et al., 2022; Zhang et al., 2025a). This motivates learning from the continuations already present in social interaction records.

To bridge this gap, we propose Observed Continuation Policy Optimization (OCPO), a framework that converts immediate partner responses recorded in high-quality expert dialogues into local supervision for social policy learning. Our key idea is to use an observed continuation as a shared reference for comparing policy-sampled utterances at the same dialogue state. To make this comparison informative, we introduce the Sensitivity-Weighted Likelihood Reward (SWLR), which aggregates candidate-conditioned token log-likelihoods using sensitivity-based weights. Token sensitivity is measured by the absolute change in log-likelihood when the candidate utterance is included in the context. Because an observed continuation may distinguish some candidate groups only weakly, we further use mean-only advantage centering to preserve the magnitude of within-group reward differences. This avoids amplifying weak distinctions through standard deviation normalization and provides relative advantages for sequence-level policy optimization. By reusing recorded continuations, OCPO obtains these rewards without additional full-dialogue rollouts, external evaluator calls, or a separately trained reward model during policy optimization.

Our contributions are:

- We introduce OCPO, a social policy learning framework that builds turn-level local supervision from observed partner continuations in offline trajectories, eliminating extra dialogue rollouts, separate reward models, and external evaluator calls to reduce the high cost of fine-grained supervision.
- We develop two dedicated algorithmic components: sensitivity-based token weighting that emphasizes tokens sensitive to candidate utterances, and mean-only advantage centering that avoids over-amplification of noise from weak supervision.
- We evaluate our method on the Sotopia benchmark environment, where we obtain the best backbone-matched Overall score in seven out of eight settings and reduce A100 GPU-hours by 64.8%-73.6% against prevailing social RL pipelines.

2 METHODOLOGY

To enable social language agents to learn adaptive dialogue policies from observed partner responses, we introduce the Observed Continuation Policy Optimization (OCPO) framework, as illustrated in Figure 1. The framework follows a two-stage training procedure: behavioral cloning first initializes the policy, after which observed partner continuations provide local supervision for policy refinement.

2.1 TASK FORMULATION

Following SOTOPIA (Zhou et al., 2024), we consider a social interaction environment with two LLM agents. Each agent $i \in \{1, 2\}$ is characterized by a profile b_i and a private social goal g_i. A scenario S specifies the social setting and interaction constraints, and $\mathcal{H}_{t-1}$ denotes the dialogue history before turn t.

The interaction proceeds over a sequence of dialogue turns. At turn t, the active agent forms its current context from the scenario, its profile and goal, and the preceding dialogue history:

$$s_t = \text{Context}\left(S, \mathcal{H}_{t-1}, b_i, g_i\right). \tag{1}$$

Given s_t, the agent samples an utterance a_t from the social policy:

$$a_t \sim \pi_\theta(\cdot \mid s_t), \tag{2}$$

where π_θ denotes the policy being optimized. If the interaction continues, the other agent produces the next utterance y_{t+1}, which is added to the dialogue history together with the current utterance. The interaction proceeds until termination, producing a complete dialogue history $\mathcal{H}_T$.

At the end of the interaction, an evaluator assesses each agent over M social dimensions:

$$\mathbf{d}_i = \text{Eval}\left(S, \mathcal{H}_T, b_i, g_i\right), \qquad \mathbf{d}_i = \left(d_{i,1}, \ldots, d_{i,M}\right). \tag{3}$$

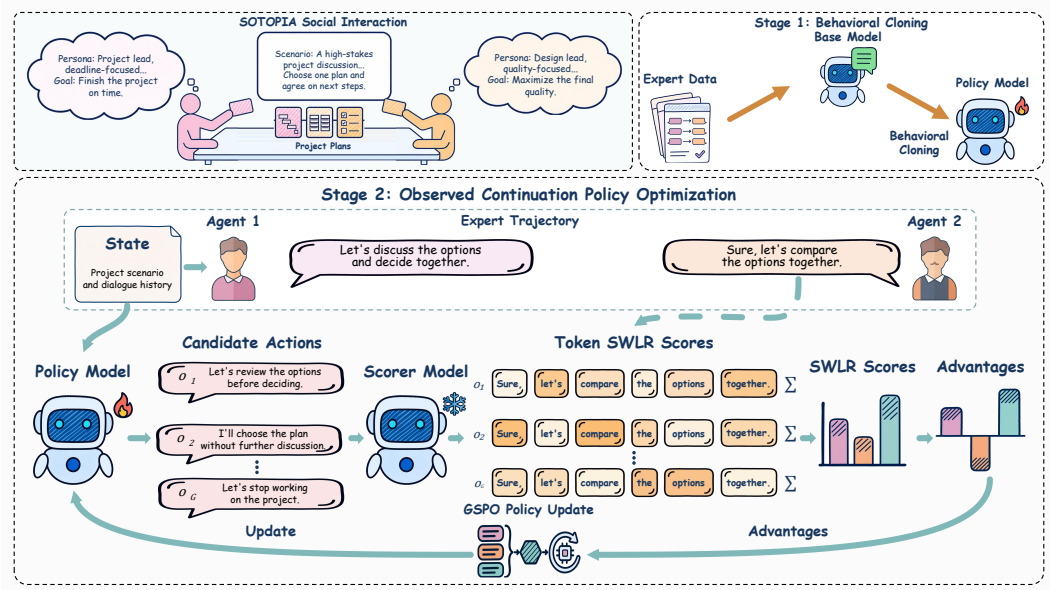


Figure 1: Overview of our OCPO framework. Stage 1 initializes the policy through behavioral cloning on expert dialogues. Stage 2 computes SWLR from observed continuations using sensitivity-based token weighting and uses mean-centered advantages for GSPO updates.

These dimension scores characterize the final social outcome of the complete interaction. However, the quality of an individual utterance is often revealed through later dialogue, leaving turn-level decisions with delayed and indirect supervision. This poses a central challenge for social policy learning: how to provide informative local supervision for individual utterances before the complete interaction outcome is available. To address this challenge, we propose OCPO, which turns observed next utterances into immediate local supervision for turn-level policy optimization.

2.2 Behavioral Cloning Initialization

Before policy optimization, we initialize the policy through behavioral cloning on expert data. Given

$$\mathcal{D}_{\mathrm{BC}} = \left\{(s_n, a_n^{\mathrm{ref}})\right\}_{n=1}^{N}, \tag{4}$$

where a_n^{ref} denotes the reference utterance from the expert trajectory. We minimize the behavioral cloning objective (Bain & Sammut, 1999; Ross & Bagnell, 2010)

$$\mathcal{L}_{\mathrm{BC}}(\theta) = -\mathbb{E}_{(s,a^{\mathrm{ref}})\sim\mathcal{D}_{\mathrm{BC}}}\left[\frac{1}{|a^{\mathrm{ref}}|}\sum_{m=1}^{|a^{\mathrm{ref}}|}\log \pi_\theta\left(a_m^{\mathrm{ref}} \mid s, a_{<m}^{\mathrm{ref}}\right)\right]. \tag{5}$$

Minimizing this objective yields the initial policy π_{θ_0} used by OCPO. Appendix B details data construction, and Appendix D reports the training settings.

Behavioral cloning provides a useful initialization, but further policy improvement requires supervision that can distinguish among utterances sampled from the current policy. Existing optimization methods often obtain such supervision through continued dialogue interaction or external evaluation. Although these signals can guide policy improvement, the supervision assigned to a particular utterance is affected by later dialogue turns and usually requires additional generation or evaluation.

2.3 Observed Continuation Policy Optimization

OCPO constructs local supervision from the observed next utterance. Given a dialogue state s, the current policy samples a group of utterances,

$$a_i \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid s), \qquad i = 1, \ldots, G, \tag{6}$$

where G denotes the number of sampled utterances and $\pi_{\theta_{\text{old}}}$ denotes the policy before the current update. All sampled utterances are scored against the same observed continuation using SWLR, computed from token log-likelihoods using sensitivity-based weights (Section 2.3.1). We then mean-center SWLR scores to obtain group-relative advantages (Section 2.3.2) for the sequence-level policy update (Section 2.3.3). Candidate validation and sampling details are provided in Appendix C.

2.3.1 Sensitivity-Weighted Likelihood Reward

Prior work has shown that follow-up likelihood can provide useful evaluative information for dialogue and language model alignment (De Bruyn et al., 2022; Zhang et al., 2025a; Pang et al., 2024). In social policy optimization, however, the tokens in the next utterance do not contribute equally to the learning signal. Some are already highly predictable from the preceding dialogue context. Their log-likelihoods therefore change little when the current utterance is included. Other tokens show larger log-likelihood changes under the current utterance. We quantify this sensitivity and use it to weight token contributions to SWLR.

Token Sensitivity. Let $p_{\theta_{\text{old}}}$ denote the policy before the current update, and let $\mathcal{C}$ index the content tokens of the observed continuation y. Template and formatting tokens are excluded from $\mathcal{C}$ (Appendix C.3). For each sampled utterance a_i, we compute the candidate-conditioned token log-likelihood

$$\ell_{i,m}^{\text{utt}} = \log p_{\theta_{\text{old}}}\left(y_m \mid s, a_i, y_{<m}\right), \tag{7}$$

and the corresponding log-likelihood using only the preceding dialogue context

$$\ell_{m}^{\text{ctx}} = \log p_{\theta_{\text{old}}}\left(y_m \mid s, y_{<m}\right). \tag{8}$$

We define token sensitivity as the absolute change in log-likelihood when the sampled utterance is included in the conditioning context:

$$\Delta_{i,m} = \left|\ell_{i,m}^{\text{utt}} - \ell_{m}^{\text{ctx}}\right|. \tag{9}$$

Sensitivity-Based Token Weighting. We convert token sensitivities into normalized weights using a softmax:

$$\alpha_{i,m} = \frac{\exp(\Delta_{i,m}/\tau)}{\sum_{k\in\mathcal{C}} \exp(\Delta_{i,k}/\tau)}, \qquad m \in \mathcal{C}, \tag{10}$$

where τ controls the concentration of the weighting distribution. This weighting emphasizes continuation tokens whose log-likelihoods change more when the sampled utterance is included.

Sensitivity-Weighted Likelihood Reward. We then aggregate the candidate-conditioned token log-likelihoods using these weights:

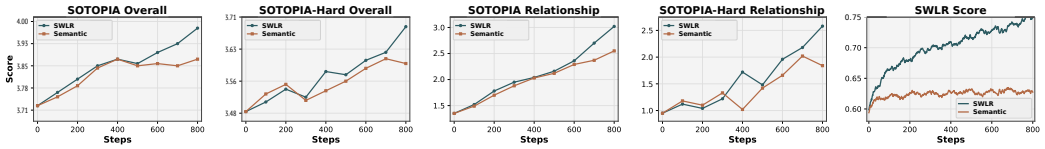
$$r_i^{\text{SWLR}} = \sum_{m\in\mathcal{C}} \alpha_{i,m}\ell_{i,m}^{\text{utt}}. \tag{11}$$

Token sensitivities $\Delta_{i,m}$ determine the weights $\alpha_{i,m}$, which are used to aggregate the candidate-conditioned log-likelihoods $\ell_{i,m}^{\text{utt}}$ into SWLR.

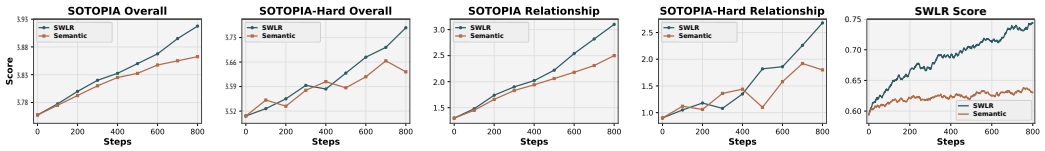
2.3.2 Mean-Only Advantage Estimation

SWLR assigns a reward to each sampled utterance within a group. These rewards must then be converted into relative advantages for policy optimization. A common choice in group-based optimization is to subtract the group mean and divide by the within-group standard deviation, placing different groups on a similar advantage scale.

For OCPO, this normalization requires particular care. The supervision does not come from explicit preference labels. It is also not a score produced by a reward model on a predefined scale. Instead, all sampled utterances in a group are evaluated using the same next utterance, and their relative preference is determined by the resulting SWLR scores. The size of the SWLR differences therefore indicates how clearly this next utterance distinguishes the sampled utterances. When the rewards differ substantially, the next utterance provides a clear preference among them. When the rewards are close, it provides only limited preference information for the current decision.



(a) Results on Qwen2.5-7B-Instruct



(b) Results on Llama3.1-8B-Instruct

Figure 2: Comparison of SWLR and Semantic supervision under GPT-4o-as-Partner evaluation across different base models in terms of training dynamics, including task performance and SWLR Score. (a) Results on Qwen2.5-7B-Instruct. (b) Results on Llama3.1-8B-Instruct.

Standard deviation normalization removes this distinction in signal strength. A group with only small reward differences also has a small standard deviation. Dividing by it amplifies these weak differences. As a result, limited preference information can produce policy updates comparable in magnitude to much clearer supervision. This effect is consistent with prior analyses showing that standard deviation normalization can overemphasize groups with low reward variance (Liu et al., 2025b).

We therefore use mean-only centering and define the group advantage as

$$\widehat{A}_i = r_i^{\mathrm{SWLR}} - \frac{1}{G}\sum_{j=1}^{G} r_j^{\mathrm{SWLR}}, \qquad i = 1, \ldots, G. \tag{12}$$

Subtracting the group mean preserves the relative preference within each group: sampled utterances with higher SWLR rewards receive positive advantages, while those with lower rewards receive negative advantages. Without standard deviation normalization, the magnitude of the advantages also retains the strength of the preference information provided by the next utterance.

2.3.3 Sequence-Level Policy Optimization

SWLR assigns a scalar reward to each sampled utterance, and mean-only centering converts these rewards into utterance-level advantages. The policy update should follow the same level of supervision. We therefore use Group Sequence Policy Optimization (GSPO) (Zheng et al., 2025), which performs optimization at the sequence level and assigns one importance ratio to each sampled utterance. Let $\mathcal{I}_i$ denote the token positions of utterance a_i. Its sequence-level importance ratio is

$$\rho_i(\theta) = \exp\left(\frac{1}{|\mathcal{I}_i|}\sum_{m\in\mathcal{I}_i}\log\frac{\pi_\theta(a_{i,m} \mid s, a_{i,<m})}{\pi_{\theta_{\mathrm{old}}}(a_{i,m} \mid s, a_{i,<m})}\right). \tag{13}$$

Using the advantage $\widehat{A}_i$ from Section 2.3.2, we optimize the sampled utterances with the clipped GSPO objective:

$$\mathcal{L}_{\mathrm{GSPO}}(\theta) = -\frac{1}{G}\sum_{i=1}^{G}\min\left(\rho_i(\theta)\widehat{A}_i, \mathrm{clip}\left(\rho_i(\theta), 1-\epsilon, 1+\epsilon\right)\widehat{A}_i\right), \tag{14}$$

where ϵ controls the clipping range. This keeps the policy update aligned with the utterance-level SWLR supervision instead of assigning separate optimization ratios to individual generation tokens.

OCPO optimizes relative preferences among utterances sampled from the current policy. These relative updates can gradually move the policy away from the supervised behavior learned during behavioral cloning. We therefore retain the Stage 1 policy as a fixed reference,

$$\pi_{\mathrm{ref}} = \pi_{\theta_0}, \tag{15}$$

Table 1: Main results on SOTOPIA and SOTOPIA-Hard under Self-Play and GPT-4o-as-Partner evaluation. We report Overall and highlight the best result for each backbone and setting in bold. Results for our models are averaged over four runs. Evaluation and aggregation details are provided in Appendix E.

Models	*Self-Play*		*GPT-4o-as-Partner*	
	SOTOPIA OVERALL↑	**SOTOPIA-Hard** OVERALL↑	**SOTOPIA** OVERALL↑	**SOTOPIA-Hard** OVERALL↑
Proprietary LLMs				
GPT-4o	3.76	3.46	3.76	3.46
Claude-3.5-Sonnet	3.71	3.09	3.77	3.30
DeepSeek-V3	3.62	3.09	3.72	3.31
Large Reasoning Models				
OpenAI-o1	3.58	2.71	3.69	3.20
OpenAI-o3-mini	3.30	2.36	3.61	2.98
Gemini-2.5-Pro	3.43	2.55	3.59	3.09
DeepSeek-R1	3.40	2.73	3.49	2.95
QwQ-32B	3.30	2.41	3.47	2.91
Qwen2.5-7B-Instruct	3.55	3.01	3.13	2.90
w/ PPDPP (Deng et al., 2024)	3.65	3.31	3.71	3.35
w/ EPO (Liu et al., 2025a)	3.51	3.12	3.86	3.51
w/ DAT (Li et al., 2024)	3.59	3.10	3.70	3.36
w/ DSI (Zhang et al., 2025b)	3.75	3.51	3.70	3.42
w/ SOTOPIA-RL (Yu et al., 2025)	3.55	3.80	3.90	3.29
w/ SAVOIR (Feng et al., 2026)	3.85	**3.97**	3.94	3.51
w/ ALSO (Li et al., 2026)	3.75	3.20	3.80	3.14
w/ OCPO (Ours)	**3.87**	3.57	**3.98**	**3.69**
Llama3.1-8B-Instruct	3.23	2.36	3.62	3.05
w/ PPDPP (Deng et al., 2024)	3.40	2.57	3.66	3.10
w/ EPO (Liu et al., 2025a)	3.78	3.27	3.83	3.53
w/ DAT (Li et al., 2024)	3.44	2.85	3.65	3.03
w/ DSI (Zhang et al., 2025b)	3.64	3.31	3.67	3.41
w/ ALSO (Li et al., 2026)	3.68	3.16	3.73	3.13
w/ OCPO (Ours)	**3.88**	**3.70**	**3.92**	**3.76**

and apply KL regularization to constrain overall policy drift. We also retain the reference utterance through an auxiliary supervised loss $\mathcal{L}_{\mathrm{SFT}}$. The KL term regularizes the policy distribution, while the supervised term directly reinforces the reference behavior at each training state.

The complete OCPO objective is

$$\mathcal{L}_{\mathrm{OCPO}}(\theta) = \mathcal{L}_{\mathrm{GSPO}}(\theta) + \lambda_{\mathrm{SFT}}\mathcal{L}_{\mathrm{SFT}} + \lambda_{\mathrm{KL}}\mathcal{L}_{\mathrm{KL}}. \tag{16}$$

The reference utterance contributes only through the supervised loss and does not participate in SWLR scoring or relative advantage estimation. Detailed KL estimation, masking, and optimization settings are provided in Appendices C and D.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETTINGS

Models. We instantiate OCPO with Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct, allowing us to evaluate its effectiveness across different model backbones under a consistent training and evaluation setup. Using two distinct model families provides a broader assessment of the generality and stability of OCPO across backbone choices.

Training. We construct the behavioral cloning and OCPO training data from high-quality SOTOPIA-Ω trajectories. During OCPO, the current policy samples eight utterances for each training state, and both models are optimized for 800 policy updates. Detailed data construction, sampling, and optimization settings are provided in Appendices B, C, and D.

Baselines. We compare OCPO with three groups of baselines. Proprietary LLMs include GPT-4o, Claude-3.5-Sonnet, and DeepSeek-V3 (Hurst et al., 2024; Anthropic, 2024; Liu et al., 2024). Large reasoning models include OpenAI-o1, OpenAI-o3-mini, Gemini-2.5-Pro, DeepSeek-R1, and QwQ-32B (Jaech et al., 2024; OpenAI, 2025; Comanici et al., 2025; Guo et al., 2025; Qwen Team, 2024). Specialized social intelligence methods include PPDPP, EPO, DAT, DSI, SOTOPIA-RL, SAVOIR, and ALSO (Deng et al., 2024; Liu et al., 2025a; Li et al., 2024; Zhang et al., 2025b; Yu et al., 2025; Feng et al., 2026; Li et al., 2026). For detailed baseline implementations and result sources, see Appendix F.

Benchmarks. We evaluate on SOTOPIA and SOTOPIA-Hard (Zhou et al., 2024). SOTOPIA covers diverse social scenarios, while SOTOPIA-Hard contains more challenging interactions that require stronger social coordination and reasoning.

Evaluation. Following the official SOTOPIA protocol, GPT-4o evaluates each completed interaction along seven social dimensions. We use their mean score, Overall, as the primary metric. Goal and Relationship are reported only in diagnostic analyses. We evaluate under two interaction settings. In Self-Play, both roles are played by the evaluated policy. In GPT-4o-as-Partner, one role is fixed to GPT-4o while the other is played by the evaluated policy. Agent and evaluator temperatures are set to 0.7 and 0, respectively. Results for our models are averaged over four runs, with evaluation and aggregation details provided in Appendix E.

3.2 Experimental Results and Analysis

RQ1: Is OCPO effective for social policy learning? As shown in Table 1, OCPO achieves the highest Overall scores in seven of eight evaluated settings among methods using the same backbone. On Llama3.1-8B-Instruct, OCPO outperforms all corresponding baselines across both benchmarks and interaction settings. In particular, it improves Overall on SOTOPIA-Hard by 8.7% (3.46 $\rightarrow$ 3.76) over GPT-4o under GPT-4o-as-Partner evaluation. On Qwen2.5-7B-Instruct, OCPO leads in three of four settings, demonstrating its effectiveness across different model families. Compared with proprietary LLMs and large reasoning models, both OCPO models achieve higher scores across all four evaluation settings. Against specialized social intelligence methods, OCPO shows a clear advantage on SOTOPIA-Hard with GPT-4o-as-Partner. It exceeds the strongest corresponding baselines by 0.18 points on Qwen2.5-7B-Instruct and 0.23 points on Llama3.1-8B-Instruct. Overall, these findings highlight the effectiveness of OCPO for social policy learning, particularly in challenging interactions with a fixed partner.

RQ2: How computationally efficient is OCPO? Table 2 compares the training efficiency of OCPO with SOTOPIA-RL and SAVOIR. For Qwen2.5-7B-Instruct, OCPO uses 56.33 A100 GPU-hours for behavioral cloning and policy optimization. This represents reductions of 73.6% and 64.8% relative to the reported budgets of SOTOPIA-RL and SAVOIR, respectively. Both baselines require additional LLM calls to generate training data for a separate 7B reward model. By deriving rewards from observed partner continuations, OCPO avoids both reward-model data generation and separate reward-model training, simplifying the overall training pipeline.

Table 2: Training efficiency comparison with existing methods. ✓/✗ indicates whether additional LLM calls are required to generate training data for a separately trained reward model (RM).

Method	Extra LLM Calls for RM Data	Reward Model Params.	A100 GPU-h
SOTOPIA-RL	✓	7B	213
SAVOIR	✓	7B	160
OCPO (Ours)	✗	0	**56.33**

RQ3: Does SWLR capture information beyond semantic similarity? To test whether the gains of OCPO mainly come from encouraging the policy to imitate expert utterances more closely, we construct a matched Semantic variant that replaces SWLR with semantic similarity to the expert utterance at the same dialogue state, while keeping the remaining training procedure unchanged. In open-ended social interaction, multiple different utterances may be reasonable, making supervision tied to a single expert utterance restrictive. SWLR does not require a sampled utterance to resemble the expert utterance; instead, it evaluates each sampled utterance using the likelihood of the subsequent utterance, allowing different expressions to receive useful supervision. Figure 2 shows that

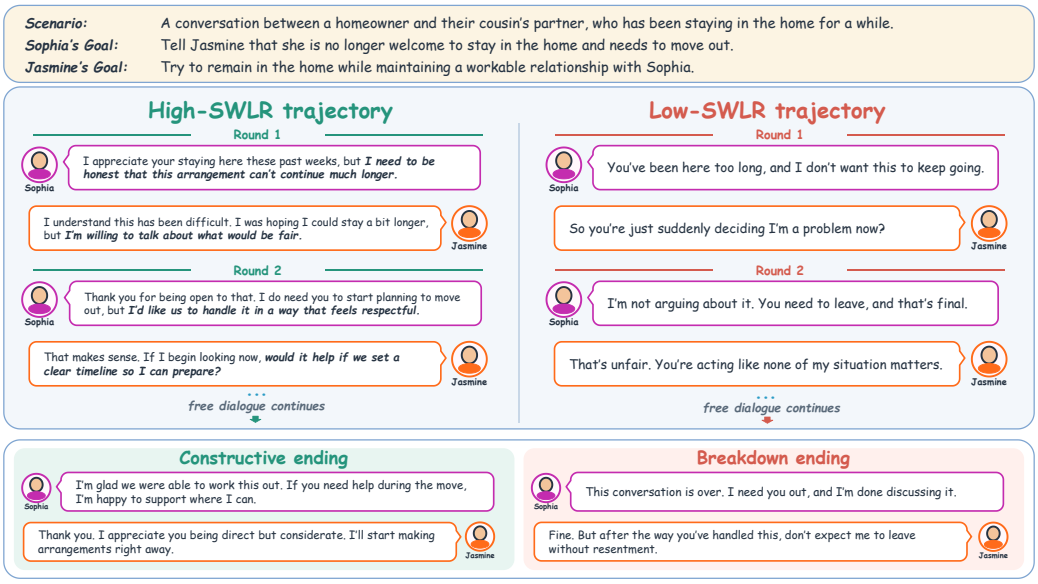


Figure 3: Qualitative comparison under cumulative SWLR intervention. Repeated high-SWLR selections preserve a constructive interaction trajectory, while repeated low-SWLR selections lead to breakdown under the same social scenario.

SWLR achieves stronger task performance and higher SWLR Scores than Semantic, indicating that SWLR captures useful information beyond semantic similarity.

RQ4: How does OCPO's local supervision relate to final social outcomes? OCPO provides turn-level supervision, while social performance is evaluated after the complete interaction. We therefore examine whether its local preferences translate into better final outcomes using Overall (the mean across seven SOTOPIA dimensions), Goal, and Relationship. Figures 3 and 4 provide qualitative and quantitative analyses.

Figure 3 shows that repeated high-SWLR selections keep the interaction open to negotiation and reciprocal engagement, whereas low-SWLR selections elicit increasingly defensive responses and push the dialogue toward breakdown. This suggests that OCPO's local supervision can accumulate across turns and shape subsequent interaction dynamics over the remaining dialogue.

Figure 4 quantitatively tests this effect using paired high- and low-SWLR branches with three consecutive interventions starting at early, middle, or late stages. High-SWLR branches consistently achieve better Overall, Goal, and Relationship outcomes, indicating that OCPO supports both task progress and interpersonal outcomes throughout the interaction and across later dialogue turns.

The advantage is largest for early interventions and diminishes at later stages, consistent with earlier choices influencing more subsequent turns. Together, these results link OCPO's local supervision to more favorable interaction development and final social outcomes.

RQ5: Which design choices are important for OCPO?

To isolate the contributions of OCPO's optimization and reward-design choices, we compare the full method with variants that replace GSPO with GRPO (Shao et al., 2024), use uniform token weights, or replace mean-only centering with standard-deviation normalization. The results in Table 3 identify advantage scaling as the most influential design choice: standard-deviation normalization reduces Overall from 3.98 to 3.63 on SOTOPIA and from 3.69 to 3.41 on SOTOPIA-Hard. Removing sensitivity-based token weighting causes the next largest degradation, lowering the scores to 3.67 and 3.48, respectively, which supports weighting continuation tokens according to their sensitivity to the candidate utterance. Replacing GSPO with GRPO also consistently reduces performance, producing scores of 3.80 and 3.51. These results support the complementary contributions of sensitivity-based weighting, mean-only advantage centering, and sequence-level optimization.

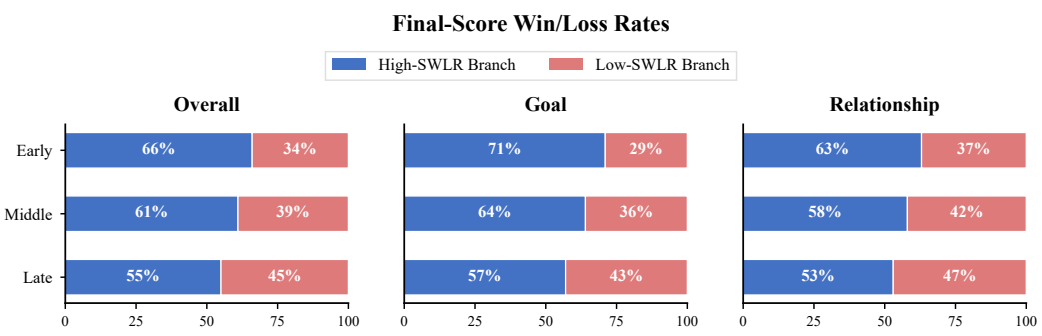


Figure 4: Final-score win/loss rates for high- and low-SWLR branches across Overall, Goal, and Relationship after three-step interventions starting at early, middle, and late stages.

4 Related Work

Large Reasoning Models. Large reasoning models approach social intelligence by strengthening the internal reasoning capabilities of language models (Jaech et al., 2024; OpenAI, 2025; Comanici et al., 2025; Guo et al., 2025; Qwen Team, 2024). Social interaction requires agents to interpret evolving dialogue contexts, infer other participants' intentions or private information, and balance task objectives with interpersonal considerations (Sap et al., 2019; Zadeh et al., 2019; Chen et al., 2024; Zhou et al., 2024). Stronger reasoning can support these decisions, but does not directly provide informative supervision for individual social actions (Zhou et al., 2024; Wang et al., 2024; Kong et al., 2025; Yu et al., 2025; Wang et al., 2026). OCPO complements this approach with local supervision derived from observed next utterances.

Table 3: OCPO ablation results.

Qwen2.5-7B-Instruct	SOTOPIA	SOTOPIA-Hard
	Overall↑	Overall↑
OCPO (full)	**3.98**	**3.69**
w/ GRPO	3.80	3.51
w/o Token Weighting	3.67	3.48
w/o Mean-only Advantage	3.63	3.41

Social Agent Learning. Static benchmarks provide limited coverage of dynamic social interaction (Sap et al., 2019; Zadeh et al., 2019; Chen et al., 2024), while SOTOPIA evaluates open-ended multi-turn interactions with social goals and evolving contexts (Zhou et al., 2024). Methods in these settings generally follow two paradigms. (1) End-to-end social policy learning directly optimizes agent behavior from interaction supervision (Wang et al., 2024; Kong et al., 2025; Yu et al., 2025; Feng et al., 2026), but often relies on sparse or costly learning signals for individual actions. (2) Planning- and strategy-based methods provide explicit guidance for social decision making (Deng et al., 2024; Li et al., 2024; Liu et al., 2025a; Zhang et al., 2025b; Wang et al., 2026; Li et al., 2026), but introduce additional inference or evaluation overhead. These limitations motivate OCPO, which leverages observed next-utterance likelihood to provide local supervision, enabling more efficient and informative turn-level social policy optimization.

5 Conclusion

This paper introduces Observed Continuation Policy Optimization (OCPO), a framework that transforms observed partner responses in high-quality dialogue trajectories into turn-level supervision for social policy learning. To assess sampled utterances against a shared continuation, we develop the Sensitivity-Weighted Likelihood Reward (SWLR), which emphasizes response tokens whose likelihood is sensitive to the candidate utterance. We further use mean-only advantage centering and sequence-level policy optimization to preserve the strength of relative preferences during policy updates. By reusing recorded continuations, OCPO avoids additional dialogue rollouts, separately trained reward models, and external evaluator calls during optimization. Experiments on SOTOPIA and SOTOPIA-Hard show that OCPO achieves the best backbone-matched Overall score in seven of eight settings while reducing A100 GPU-hours by 64.8%–73.6% compared with SOTOPIA-RL and SAVOIR. Together, these results show that OCPO enables the efficient development of stronger social agents, achieving competitive social performance with substantially lower training compute.

6 AI USE DISCLOSURE

Generative AI tools were used solely to aid and polish the writing of this paper, including language editing, rephrasing, and improving the clarity and organization of the manuscript. All scientific content, research decisions, methods, experiments, analysis, and conclusions were developed and verified by the authors. The authors reviewed all AI-assisted text and take full responsibility for the content of the paper. Language models used as agents, partners, scorers, or evaluators are components of the experimental methodology and are described separately in the corresponding sections and appendices.

REFERENCES

Anthropic. Claude 3.5 sonnet model card addendum. *Claude 3.5 Sonnet Model Card*, 2024. URL https://www.anthropic.com/claude/sonnet.

Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pp. 103–129. Oxford University Press, 1999. doi: 10.1093/oso/9780198538677.003.0006. URL https://www.cse.unsw.edu.au/~claude/papers/MI15.pdf.

Hongzhan Chen, Hehong Chen, Ming Yan, Wenshen Xu, Gao Xing, Weizhou Shen, Xiaojun Quan, Chenliang Li, Ji Zhang, and Fei Huang. SocialBench: Sociality evaluation of role-playing conversational agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2108–2126. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.125. URL https://aclanthology.org/2024.findings-acl.125/.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. URL https://arxiv.org/abs/2507.06261.

Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann, and Walter Daelemans. Open-domain dialog evaluation using follow-ups likelihood. In *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 496–504. International Committee on Computational Linguistics, 2022. URL https://aclanthology.org/2022.coling-1.40/.

Yang Deng, Wenxuan Zhang, Wai Lam, See-Kiong Ng, and Tat-Seng Chua. Plug-and-play policy planner for large language model powered dialogue agents. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=MCNqgUFTHI.

Xiachong Feng, Yi Jiang, Xiaocheng Feng, Deyi Yin, Libo Qin, Yangfan Ye, Lei Huang, Weitao Ma, Yuxuan Gu, Chonghan Qin, Bing Qin, and Lingpeng Kong. SAVOIR: Learning social savoir-faire via shapley-based reward attribution, 2026. URL https://arxiv.org/abs/2604.18982.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, A. J. Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. URL https://arxiv.org/abs/2410.21276.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024. URL https://arxiv.org/abs/2412.16720.

Aobo Kong, Wentao Ma, Shiwan Zhao, Yongbin Li, Yuchuan Wu, Ke Wang, Xiaoqian Liu, Qicheng Li, Yong Qin, and Fei Huang. SDPO: Segment-level direct preference optimization for social agents, 2025. URL https://arxiv.org/abs/2501.01821.

Kenneth Li, Yiming Wang, Fernanda Viégas, and Martin Wattenberg. Dialogue action tokens: Steering language models in goal-directed dialogue with a multi-turn planner. *arXiv preprint arXiv:2406.11978*, 2024. URL https://arxiv.org/abs/2406.11978.

Xiang Li, Liping Yi, Mingze Kong, Min Zhang, Zhongxiang Dai, and Qinghua Hu. ALSO: Adversarial online strategy optimization for social agents, 2026. URL https://arxiv.org/abs/2605.15768.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024. URL https://arxiv.org/abs/2412.19437.

Xiaoqian Liu, Ke Wang, Yongbin Li, Yuchuan Wu, Wentao Ma, Aobo Kong, Fei Huang, Jianbin Jiao, and Junge Zhang. EPO: Explicit policy optimization for strategic reasoning in LLMs via reinforcement learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 15371–15396, 2025a. URL https://aclanthology.org/2025.acl-long.747/.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *Conference on Language Modeling (COLM)*, 2025b. URL https://arxiv.org/abs/2503.20783.

OpenAI. Openai o3-mini system card. *OpenAI o3-mini System Card*, 2025. URL https://openai.com/index/o3-mini-system-card/.

Richard Yuanzhe Pang, Stephen Roller, Kyunghyun Cho, He He, and Jason Weston. Leveraging implicit feedback from deployment data in dialogue. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 60–75. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.eacl-short.8. URL https://aclanthology.org/2024.eacl-short.8/.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown. *Qwen Blog*, 2024. URL https://qwenlm.github.io/blog/qwq-32b-preview/.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pp. 661–668. PMLR, 2010. URL https://proceedings.mlr.press/v9/ross10a.html.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 4463–4473. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

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

Minzheng Wang, Yongbin Li, Haobo Wang, Xinghua Zhang, Nan Xu, Bingli Wu, Fei Huang, Haiyang Yu, and Wenji Mao. Adaptive social learning via mode policy optimization for language agents. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GG7YQnsdhp.

Ruiyi Wang, Haofei Yu, Wenxin Zhang, Zhengyang Qi, Maarten Sap, Yonatan Bisk, Graham Neubig, and Hao Zhu. SOTOPIA-π: Interactive learning of socially intelligent language agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12912–12940. Association for Computational Linguistics,

2024. doi: 10.18653/v1/2024.acl-long.698. URL https://aclanthology.org/2024.acl-long.698/.

Haofei Yu, Zhengyang Qi, Yining Zhao, Kolby Nottingham, Keyang Xuan, Bodhisattwa Prasad Majumder, Hao Zhu, Paul Pu Liang, and Jiaxuan You. Sotopia-RL: Reward design for social intelligence, 2025. URL https://arxiv.org/abs/2508.03905.

Amir Zadeh, Michael Chan, Paul Pu Liang, Edmund Tong, and Louis-Philippe Morency. Social-IQ: A question answering benchmark for artificial social intelligence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8807–8817, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Zadeh_Social-IQ_A_Question_Answering_Benchmark_for_Artificial_Social_Intelligence_CVPR_2019_paper.html.

Chen Zhang, Dading Chong, Feng Jiang, Chengguang Tang, Anningzhe Gao, Guohua Tang, and Haizhou Li. Aligning language models using follow-up likelihood as reward signal. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25832–25841, 2025a. doi: 10.1609/aaai.v39i24.34776. URL https://ojs.aaai.org/index.php/AAAI/article/view/34776.

Wenyuan Zhang, Tianyun Liu, Mengxiao Song, Xiaodong Li, and Tingwen Liu. SOTOPIA-Ω: Dynamic strategy injection learning and social instruction following evaluation for social agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24669–24697. Association for Computational Linguistics, 2025b. doi: 10.18653/v1/2025.acl-long.1203. URL https://aclanthology.org/2025.acl-long.1203/.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Zhengyang Qi, Haofei Yu, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. SOTOPIA: Interactive evaluation for social intelligence in language agents. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mM7VurbA4r.

APPENDIX OVERVIEW

The appendix documents broader impacts and limitations (Appendix A), data construction and provenance (Appendix B), the complete OCPO procedure and SWLR implementation (Appendix C), training configuration and compute (Appendix D), evaluation (Appendix E), baseline and ablation implementations (Appendix F), signal and intervention analyses (Appendix G), and prompts and data formats (Appendix H).

A BROADER IMPACTS AND LIMITATIONS

OCPO provides a compute-efficient route for adapting social agents from multi-turn interaction data, with potential applications in negotiation, coordination, tutoring, and assistance. Such systems can nevertheless influence users, reproduce cultural stereotypes, or optimize observable reactions that diverge from user welfare. Practical deployments should preserve user control, audit behavior across demographic and cultural settings, and require domain-specific validation before high-stakes use.

OCPO evaluates multiple sampled utterances using the likelihood of the same next utterance. This local comparison assumes that the next utterance remains informative when the current utterance is varied. A sampled utterance could plausibly elicit a different yet socially valid next utterance and therefore receive a lower SWLR score under the fixed scoring target. SWLR should therefore be interpreted as local likelihood-based supervision rather than a direct estimate of long-horizon causal value. In addition, when the scorer and policy share a model family, model-specific language predictability can partly influence the learning signal.

The Stage 2 corpus retains only training instances with an immediate nonempty next `speak` utterance, and online sampling excludes `leave` actions. Terminal-action behavior is therefore learned mainly through behavioral cloning and the auxiliary supervised loss. SOTOPIA and GPT-4o evaluation cover only a subset of social settings and cultural norms. Finally, the main compute comparison reports directly measured A100 GPU-hours rather than training FLOPs.

B DATA CONSTRUCTION AND STATISTICS

B.1 DATASET PROVENANCE

OCPO uses the public `WYRipple/sotopia-omega` release, distributed under the CC-BY-NC-SA-4.0 license. The released SOTOPIA-Ω corpus was generated with a Qwen2.5-72B-Instruct expert under dynamic native, simple, and negotiation strategy injection (Zhang et al., 2025b). Upstream generation uses a maximum of 8,192 tokens, temperature 1.0, and top-p 0.95. The release applies final-GOAL filtering thresholds of 8.0 for negotiation-injected dialogues and 8.5 otherwise, with up to four generation attempts per profile. OCPO operates on the released action records and subsequently performs schema validation, exact deduplication, episode-level splitting, and next-utterance availability filtering.

B.2 ACTION VALIDATION AND DEDUPLICATION

The released data contain 25,922 raw action records. Table 4 gives the complete filtering chain. The value 25,867 denotes schema-valid records before exact deduplication; all downstream BC statistics use 25,863 unique expert actions.

Table 4: Action validation and deduplication statistics.

Stage	Count
Raw action records	25,922
Non-verbal records removed	26
Unsupported `action` records removed	21
Null actions removed	8
Valid records before deduplication	25,867
Exact duplicates removed	4
Unique expert actions	25,863
`speak` / `leave`	22,541 / 3,322

B.3 Behavioral-Cloning Split and Coverage

We construct the behavioral-cloning training and development split with seed 20260603, targeting approximately 500 development actions. The resulting split and corpus statistics are summarized in Table 5.

Table 5: Behavioral-cloning split and corpus coverage.

Quantity	Count
Training actions / episodes	25,354 / 1,639
Development actions / episodes	509 / 33
Distinct scenarios	270
Distinct scenario–role pairs	1,034
Early / middle / late action buckets	8,661 / 8,520 / 8,682

We use the development set for model selection and intermediate evaluation throughout training.

B.4 Stage-2 Training-Instance Construction

Stage 2 retains a reference action only when it is followed by an immediate, nonempty partner `speak` utterance. We remove `leave` actions, instances with an invalid immediate partner `speak` utterance, and instances without a valid next turn. Table 6 summarizes the resulting training-instance pool.

Table 6: Stage 2 next-utterance training-instance construction.

Quantity	Count
Unique expert actions entering filter	25,863
Expert `leave` actions removed	3,322
Invalid immediate partner `speak` removed	1,656
Missing next turn removed	18
Eligible OCPO training instances	20,867
Unique rendered social states	20,238
Early / middle / late instances	8,658 / 8,517 / 3,692
Train / development instances	19,797 / 1,070
Train / development episodes	1,584 / 84
Fixed development probe pool	64 states

We use the resulting training and development instances for Stage 2 optimization and intermediate evaluation. For Figure 2, we additionally fix a pool of 64 development states to provide a consistent probe set across training checkpoints.

Each Stage 2 record stores the policy-visible state, reference action, immediate next utterance, episode identity, turn identity, and a next-utterance scoring template. Construction checks chronological order and partner identity. The policy prompt ends before the next utterance; no later utter-

ance or terminal reward is exposed during sampled-utterance generation. The next utterance is used only as a teacher-forced scoring target.

C OCPO Algorithm and SWLR Implementation

C.1 End-to-End Procedure

Algorithm 1 BC-to-OCPO Training

Require: Stage 1 BC training set $\mathcal{D}_{\mathrm{BC}}$; Stage 2 training instances $\mathcal{D}_{\mathrm{OCPO}}$; number of policy updates $T = 800$; target group count $B = 16$; candidate group size $G = 8$
Ensure: Optimized policy π_θ

Behavioral cloning: train an action-only LoRA policy for one epoch on $\mathcal{D}_{\mathrm{BC}}$
Initialize trainable policy π_θ and frozen reference policy π_{ref} from the Stage 1 adapter
for $t = 1$ **to** T **do**
    Set $\pi_{\theta_{\mathrm{old}}} \leftarrow \pi_\theta$
    Initialize sampled-utterance group set $\mathcal{G} \leftarrow \emptyset$
    while $|\mathcal{G}| < B$ and fewer than 64 states have been tried **do**
        Sample a Stage 2 instance (s, a^{ref}, y) from $\mathcal{D}_{\mathrm{OCPO}}$
        Sample from $\pi_{\theta_{\mathrm{old}}}(\cdot \mid s)$ until G valid distinct `speak` utterances are accepted or 48 generation attempts are reached
        if G valid utterances are obtained **then**
            Add the sampled-utterance group to $\mathcal{G}$
        end if
    end while
    if $|\mathcal{G}| < 12$ **then**
        continue
    end if
    for all sampled utterances a_i in $\mathcal{G}$ **do**
        Construct utterance-conditioned and context-only scoring inputs
        Teacher-force the same observed next utterance y
        Compute token sensitivities $\Delta_{i,m}$ from absolute log-likelihood changes
        Compute sensitivity-based token weights $\alpha_{i,m}$ and SWLR scores r_i^{SWLR}
    end for
    Remove groups with numerically identical SWLR rewards
    Mean-center SWLR rewards within each remaining group to obtain advantages $\hat{A}_i$
    Recompute sampled-utterance and reference-utterance log-probabilities
    Compute sequence-level GSPO ratios, frozen-reference KL regularization, and auxiliary SFT loss
    Update θ with one AdamW step
end for
return π_θ

No reward model or external API is called during Stage 2 training.

C.2 Sampled-Utterance Generation and Validation

Generation uses structured JSON decoding. A sampled utterance is accepted only when it parses to the canonical action schema, has `action_type=speak`, contains a nonempty natural-language argument, has valid generated token IDs and token round-trip, yields a nonempty argument mask, and is distinct from previously accepted utterances for the same state. Outputs containing exposed reasoning markers, malformed fields, `leave` actions, exact duplicates, or length violations are rejected.

Table 7 reports the final Llama3.1-8B-Instruct run. Diagnostic counters operate at different stages of the generation pipeline and are reported independently rather than treated as a mutually exclusive decomposition.

Table 7: Online sampling diagnostics for the final Llama3.1-8B-Instruct OCPO run.

Runtime quantity	**Measured value**
Online updates completed	800 / 800
State draws	12,864
Complete sampled-utterance groups	12,800
Accepted sampled utterances	102,400
Raw generations	115,226
Parse failures	136
Generation-filter failures	12,521
Exact duplicates rejected	202
`leave` outputs rejected	12,172
Other invalid outputs	147
Accepted-utterance truncations	10

For Qwen2.5-7B-Instruct, the matched final run completes 800 updates with 12,800 complete groups and 102,400 accepted sampled utterances from 112,968 raw generations. It records 119 parse failures, 10,425 generation-filter failures, 188 exact duplicates, 10,124 rejected `leave` outputs, 113 other invalid outputs, and 7 accepted-utterance truncations.

Complete groups contain exactly eight accepted sampled utterances. If fewer than 12 complete groups remain after generation filtering or equal-reward removal, OCPO skips the optimizer update.

C.3 Likelihood Scoring Contexts and Token Masks

For sampled utterance a_i, the utterance-conditioned scoring input replaces only the reference utterance argument with the sampled utterance argument. The context-only scoring input preserves the same state, speaker position, dialogue history, chat template, and JSON serialization while replacing the current utterance argument with an empty string. Teacher forcing uses the same next-utterance target and keeps its prefix identical across both scoring inputs.

The next-utterance content mask selects tokenizer offsets entirely contained inside the JSON `argument` string and containing at least one alphanumeric character. JSON shell tokens, `action_type`, recipient fields, punctuation-only pieces, padding, and terminal tokens receive zero weight. GSPO ratios, the auxiliary SFT term, and KL regularization use the analogous argument-only mask for the sampled or reference utterance.

C.4 SWLR and Mean-Only Advantage

Let $y = (y_1, \ldots, y_M)$ denote the next utterance and let $\mathcal{C}$ denote its content-token positions. For sampled utterance a_i, the utterance-conditioned and context-only token log-likelihoods are

$$\ell_{i,m}^{\mathrm{utt}} = \log p_{\theta_{\mathrm{old}}}\left(y_m \mid s, a_i, y_{<m}\right), \tag{17}$$

$$\ell_{m}^{\mathrm{ctx}} = \log p_{\theta_{\mathrm{old}}}\left(y_m \mid s, y_{<m}\right). \tag{18}$$

Token sensitivity is the absolute change between these log-likelihoods:

$$\Delta_{i,m} = \left|\ell_{i,m}^{\mathrm{utt}} - \ell_{m}^{\mathrm{ctx}}\right|,$$

and the sensitivity-based token weight is

$$\alpha_{i,m} = \frac{\exp(\Delta_{i,m}/\tau)}{\sum_{k\in\mathcal{C}} \exp(\Delta_{i,k}/\tau)}, \qquad \tau = 1.0.$$

Token sensitivities $\Delta_{i,m}$ determine the weights $\alpha_{i,m}$ used to aggregate the candidate-conditioned log-likelihoods $\ell_{i,m}^{\mathrm{utt}}$ into SWLR, matching Section 2.3.1. For a group of $G = 8$ sampled utterances, OCPO uses mean-only centering,

$$\widehat{A}_i = r_i^{\mathrm{SWLR}} - \frac{1}{G}\sum_{j=1}^{G} r_j^{\mathrm{SWLR}}.$$

Groups with reward spread at or below 10^{-8} are removed. No within-group standard-deviation normalization is applied, preserving the strength of the relative SWLR signal in the update magnitude.

C.5 Sequence-Level Policy Update

For the sampled-utterance argument-token set $\mathcal{I}_i$, OCPO uses the sequence-level ratio

$$\rho_i(\theta) = \exp\left[\frac{1}{|\mathcal{I}_i|}\sum_{m\in\mathcal{I}_i}\log\frac{\pi_\theta(a_{i,m}\mid s, a_{i,<m})}{\pi_{\theta_{\text{old}}}(a_{i,m}\mid s, a_{i,<m})}\right].$$

The sequence log-ratio is clipped to a numerically stable range before exponentiation, and the GSPO policy ratio is clipped to $[0.8, 1.2]$. The reference utterance never enters the sampled group and contributes only through an argument-token SFT loss. The frozen Stage 1 adapter defines π_{ref} for KL regularization. For sampled-utterance token m, define the reference log-ratio

$$\delta^{\text{KL}}_{i,m} = \log\pi_\theta(a_{i,m}\mid s, a_{i,<m}) - \log\pi_{\text{ref}}(a_{i,m}\mid s, a_{i,<m}),$$

with the estimator

$$\widehat{D}_{\text{KL},i,m} = \exp(-\delta^{\text{KL}}_{i,m}) + \delta^{\text{KL}}_{i,m} - 1.$$

KL is averaged within sampled-utterance argument tokens and then across sampled utterances. The complete Stage 2 objective is

$$\mathcal{L}_{\text{OCPO}} = \mathcal{L}_{\text{GSPO}} + 0.05\,\mathcal{L}_{\text{SFT}} + 0.001\,\mathcal{L}_{\text{KL}}.$$

D Training Configuration and Compute

D.1 Stage-1 Behavioral Cloning

The Qwen backbone is `Qwen/Qwen2.5-7B-Instruct` at revision `main` with the official Qwen2.5 tokenizer and ChatML chat template. The Llama backbone is `meta-llama/Meta-Llama-3.1-8B-Instruct` at revision `main` with its official tokenizer and chat template. Table 8 reports the shared Stage 1 configuration. The Qwen and Llama BC configurations use the same data split and method-side hyperparameters.

Table 8: Stage 1 behavioral-cloning configuration.

Field	**Qwen2.5-7B-Instruct**	**Llama3.1-8B-Instruct**
Train / dev actions	25,354 / 509	25,354 / 509
Per-device batch / grad. accumulation	1 / 16	1 / 16
Effective batch size / epochs	16 / 1	16 / 1
Learning rate / weight decay	2×10^{-5} / 0	2×10^{-5} / 0
Scheduler / warmup ratio	cosine / 0.03	cosine / 0.03
Maximum sequence length	4,096	4,096
Prompt head retained	1,024	1,024
LoRA rank / alpha / dropout	16 / 32 / 0.05	16 / 32 / 0.05
LoRA target modules	q/k/v/o, gate/up/down	q/k/v/o, gate/up/down
Gradient checkpointing	enabled	enabled
Precision	bfloat16	bfloat16
Optimizer	AdamW	AdamW
AdamW betas / epsilon	0.9, 0.999 / 10^{-8}	0.9, 0.999 / 10^{-8}
Seed	20260603	20260603

For Llama3.1-8B-Instruct, Stage 1 contains 1,585 optimizer steps, 41,943,040 trainable parameters, achieves a 64/64 structured-generation parse probe. Qwen2.5-7B-Instruct likewise contains 1,585 optimizer steps, uses 40,370,176 trainable LoRA parameters, achieves a 64/64 parse probe.

D.2 Stage-2 OCPO Configuration

The final Qwen and Llama runs use the same OCPO method-side hyperparameters in Table 9; only the backbone-specific online seed differs.

Table 9: Stage 2 OCPO configuration.

Field	Qwen final run	Llama final run
Online updates	800	800
Target / minimum complete groups	16 / 12	16 / 12
Sampled utterances per state	8	8
Max. state draws / update	64	64
Max. generation attempts / state	48	48
Micro-batch size	8	8
Max. prompt / response tokens	3,840 / 256	3,840 / 256
Generation temperature / top-p	0.8 / 0.95	0.8 / 0.95
SWLR weighting temperature	1.0	1.0
Equal-reward tolerance	10^{-8}	10^{-8}
Learning rate / weight decay	10^{-6} / 0	10^{-6} / 0
Scheduler / warmup ratio	cosine / 0.10	cosine / 0.10
GSPO clip / max grad. norm	0.2 / 1.0	0.2 / 1.0
SFT / KL coefficient	0.05 / 0.001	0.05 / 0.001
Online seed	20260810	20260809
Precision	bfloat16	bfloat16

The Qwen run completes all 800 updates, forms 12,800 complete groups, and accepts 102,400 sampled utterances. It reaches 99.72% parse success and records zero train-time reward-model or external-API calls. The corresponding Llama run also completes all 800 updates with 12,800 complete groups and 102,400 accepted sampled utterances.

D.3 Runtime Environment

The Qwen2.5-7B-Instruct is trained end-to-end on one NVIDIA A100-80GB GPU.

D.4 Measured Computational Budget

Table 10: Measured wall-clock training time for the Qwen2.5-7B-Instruct run.

Stage	Wall-clock time	Hours
Action BC	4:29:58	4.50
Online OCPO	51:50:00	51.83
Total	**56:19:58**	**56.33**

The compute comparison in Table 2 uses directly reported A100 GPU-hours. The SOTOPIA-RL and SAVOIR costs are calculated from the GPU counts and wall-clock training times reported in their papers, yielding 213 and 160 A100 GPU-hours, respectively. The Qwen2.5-7B-Instruct run requires 56.33 A100 GPU-hours for the complete training pipeline.

E Benchmarks and Evaluation Protocol

E.1 Metrics and Interaction Settings

SOTOPIA evaluates seven dimensions with the official `EpisodeLLMEvaluator`: Believability (0–10), Relationship (−5–5), Knowledge (0–10), Secret (−10–0), Social Rules (−10–0), Financial and Material Benefits (−5–5), and Goal (0–10) (Zhou et al., 2024). Overall is the arithmetic mean of the seven returned scores. Main tables report Overall, while Goal and Relationship are retained for diagnostic analyses.

For every completed episode, the evaluator history contains the scenario, the two participants' visible profiles and private social goals, the complete dialogue, and both `speak` and `leave` actions. Messages containing `did nothing` are filtered before evaluation. GPT-4o receives the official `EpisodeLLMEvaluator` prompt and the Pydantic

`EvaluationForAgents[SotopiaDimensions]` schema as the structured-output constraint. The schema requires a separate `reasoning` string and numeric `score` for each of the seven dimensions under exactly two agent keys.

Self-Play pairs two instances of the evaluated policy. GPT-4o-as-Partner fixes GPT-4o as the other participant. Agent decoding temperature is 0.7 and evaluator temperature is 0. Invalid structured actions and parse failures are counted as failures without replacement utterances or reward clamping.

SOTOPIA-Hard contains 14 scenarios with five role pairs per scenario. Evaluating both agent positions gives 140 episodes per complete manifest. SOTOPIA uses 90 environments with two pairings per environment. All OCPO evaluation runs use the same fixed scenario-role manifest within each benchmark and interaction protocol.

E.2 Evaluation Runs and Aggregation

The four complete OCPO and ALSO evaluation runs use seeds 0, 1, 2, and 3 for each backbone, benchmark, and interaction protocol. Each run evaluates the full corresponding scenario-role manifest, and the reported OCPO and ALSO results are the arithmetic mean across the four runs. Externally published baselines are transcribed from their cited sources.

F Baseline and Ablation Implementations

F.1 Main-Table Sources

The proprietary models, large reasoning models, PPDPP, EPO, DAT, DSI, SOTOPIA-RL, and SAVOIR rows in Table 1 are published results transcribed from their cited sources. The ALSO rows are produced by our runs under the Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct settings. The OCPO rows are produced by the present training and evaluation pipeline. This distinction prevents published baseline aggregates from being presented as local reruns.

PPDPP uses a policy planner over predefined strategies (Deng et al., 2024); EPO generates open-ended strategies (Liu et al., 2025a); DAT predicts continuous control vectors (Li et al., 2024); DSI trains through dynamic strategy injection (Zhang et al., 2025b); SOTOPIA-RL performs utterance-level reward attribution from dialogue-level social feedback (Yu et al., 2025); SAVOIR uses Shapley-based reward attribution (Feng et al., 2026); and ALSO performs adversarial online strategy optimization (Li et al., 2026).

F.2 Matched Ablation Controls

All OCPO variants in Table 3 share the Qwen2.5-7B-Instruct BC initialization, Stage 2 training instances, sampled-utterance group size, generation parameters, online budget, and evaluation manifest. The w/ GRPO variant replaces the sequence-level ratio with token-level clipped ratios. The w/o Token Weighting variant assigns uniform next-utterance token weights. The w/o Mean-only Advantage variant divides the mean-centered group advantage by the within-group reward standard deviation. No other component changes within a paired comparison.

G Additional Signal and Intervention Analysis

G.1 SWLR and Semantic Similarity

SWLR and Semantic differ only in the supervision signal. Semantic uses Qwen3-Embedding-8B sentence representations with mean pooling, L2 normalization, and cosine similarity between each sampled utterance and the expert utterance at the same dialogue state. Both methods share the same backbone, BC initialization, sampled-utterance budget, optimizer, and training length. The SWLR Score is evaluated on a fixed 64-state development probe.

Table 11 records the endpoint values reported in Figure 2.

Table 11: Endpoint values for the SWLR and Semantic comparison.

Backbone	Metric	SWLR	Semantic
Qwen2.5-7B	SOTOPIA Overall	3.9783	3.8767
	SOTOPIA-Hard Overall	3.6870	3.5988
	SOTOPIA Relationship	3.0200	2.5500
	SOTOPIA-Hard Relationship	2.5800	1.8400
	SWLR Score	0.7548	0.6350
Llama3.1-8B	SOTOPIA Overall	3.9175	3.8625
	SOTOPIA-Hard Overall	3.7580	3.6320
	SOTOPIA Relationship	3.1000	2.5000
	SOTOPIA-Hard Relationship	2.6800	1.8000
	SWLR Score	0.7441	0.6383

SWLR and Semantic use identical run seeds within each backbone: 20260810 for Qwen2.5-7B-Instruct and 20260809 for Llama3.1-8B-Instruct. The two variants therefore share initialization, sampled-utterance budget, optimization schedule, and evaluation manifests, isolating the supervision signal as the controlled difference.

G.2 Temporal Intervention Analysis

Figure 4 tests whether repeated local SWLR preferences can accumulate through partner-conditioned dialogue states and produce episode-level outcome differences. Early, middle, and late correspond to the first, middle, and final thirds of normalized dialogue progress. Each stage contains 200 sampled starting states, with at most one starting state drawn from an episode for a given stage.

For every starting state, we construct paired high- and low-SWLR branches and apply SWLR-guided intervention at three consecutive focal-agent decision points. At each intervention point, the evaluated policy samples eight utterances. The high branch executes the sampled utterance with the largest SWLR score, while the low branch executes the one with the smallest SWLR score. The selected utterance is written to the actual branch trajectory, after which the original partner agent produces its next response naturally.

The first intervention uses the next utterance available at the original starting state. After that intervention, the two branches no longer follow the original trajectory. For the second and third intervention points, we therefore construct a branch-local continuation probe separately for each branch. Given the current branch state, DeepSeek-V4-Flash first produces a reference utterance for the focal role and then produces the corresponding partner response. This temporary reference turn is used only to obtain a plausible branch-local next utterance and is discarded immediately; neither the DeepSeek reference utterance nor its response is inserted into the evaluated trajectory.

The current evaluated policy then samples eight utterances from the unchanged branch state. All eight sampled utterances within that branch are scored against the same DeepSeek-generated next-utterance probe, preserving the controlled SWLR comparison in which the state and scoring target are fixed while only the sampled utterance varies. DeepSeek is used only for constructing this off-trajectory probe: it does not generate the evaluated sampled-utterance set, compute SWLR, produce the executed partner response, participate in the actual rollout, or perform terminal evaluation.

After the selected second- or third-step utterance is executed, the original partner agent again responds naturally, yielding the next real branch state. After the third SWLR-guided intervention and its natural partner response, all intervention stops and both branches continue with the original interaction agents until termination. The high and low branches use the same policy checkpoints, partner agents, decoding parameters, intervention schedule, and terminal evaluator. The intervention run uses seed 20260824. DeepSeek probe construction uses deterministic decoding with temperature 0, while the original interaction agents retain their normal decoding configuration.

Final Overall, Goal, and Relationship scores are produced with the same official GPT-4o SOTOPIA `EpisodeLLMEvaluator` used in the main experiments. A high-SWLR win occurs when the high branch receives a larger final metric score than the paired low branch, and conversely for a

low-SWLR win. Exact ties contribute 0.5 to each branch when producing the plotted rates. Table 12 reports the paired outcome counts and high-SWLR win rates.

Table 12: Temporal-intervention statistics. Each stage contains 200 paired starting states. Each branch receives three consecutive SWLR-guided utterance interventions, followed by natural dialogue completion after the third intervention.

Metric	Stage	High wins	Low wins	Ties	High rate
Overall	Early	123	59	18	66%
	Middle	115	71	14	61%
	Late	104	86	10	55%
Goal	Early	132	48	20	71%
	Middle	120	65	15	64%
	Late	108	82	10	57%
Relationship	Early	118	66	16	63%
	Middle	109	78	13	58%
	Late	100	90	10	53%

These counts reproduce the 66/34, 61/39, and 55/45 Overall rates; 71/29, 64/36, and 57/43 Goal rates; and 63/37, 58/42, and 53/47 Relationship rates reported in Figure 4. The larger early-stage gaps likely arise because the same three-step intervention, when introduced earlier in the dialogue, can influence more subsequent turns and therefore has more opportunities to affect the final outcome.

H Prompts and Data Formats

H.1 Policy-Visible State

The state renderer supplies the role-specific visible context when present. Otherwise it renders the scenario, participants, acting agent, partner, acting-agent profile, partner profile, the acting agent's private social goal, visible dialogue history, current turn, and available action types. Internal records additionally preserve episode and environment identifiers, actor identities, terminal state, and maximum-turn metadata for reconstruction and validation.

H.2 Canonical Action Schema

Every policy action contains exactly three fields: `action_type`, `argument`, and `to`. A `speak` action requires a nonempty natural-language argument; a `leave` action requires an empty argument.

```
{
  "action_type": "speak",
  "argument": "[utterance]",
  "to": ["[partner]"]
}
{
  "action_type": "leave",
  "argument": "",
  "to": []
}
```

The policy instruction appended to the rendered state is:

```
Choose the next action from the available actions.
Return exactly one JSON object and no surrounding text.
Use speak only with a non-empty natural-language argument.
Use leave only with an empty argument.
```

```
Do not include hidden reasoning.
Required JSON schema: object with fields action_type,
argument, and to; action_type is speak or leave, argument
is a string, and to is an array of strings. No additional
fields are allowed.
```

H.3 BC Input–Target Format

A BC record pairs the complete policy-visible prompt with one canonical expert action target. Prompt tokens receive zero language-modeling weight; the complete canonical action JSON is supervised. A representative format is:

```
INPUT
Scenario: [scenario]
Participants: [agent 1], [agent 2]
You are: [acting agent]
Partner: [partner]
Your profile: [acting profile]
Partner profile: [partner profile]
Your social goal: [private goal]
Dialogue history before this action: [visible history]
You are at Turn #[turn id].
Available action types: [speak, leave]

TARGET
{
  "action_type": "speak",
  "argument": "[expert utterance]",
  "to": ["[partner]"]
}
```

H.4 Stage-2 OCPO Training Record

Each Stage 2 record stores at least the training-instance identifier, episode identifier, policy prompt, prompt digest, reference action, immediate next utterance, source turn identity, and next-utterance scoring template. The next utterance is never included in the sampled-utterance generation prompt.

H.5 Utterance-Conditioned, Context-Only, and Target Inputs

For sampled utterance a_i, OCPO forms the following paired scoring inputs:

```
UTTERANCE-CONDITIONED INPUT
[same policy-visible state and scoring template, with
the reference argument replaced by the sampled utterance
argument]

CONTEXT-ONLY INPUT
[same state and scoring template, with the current utterance
argument span replaced by an empty string]

TEACHER-FORCED NEXT-UTTERANCE TARGET
{
  "action_type": "speak",
  "argument": "[next utterance]",
  "to": ["[recipient]"]
}
```

Both scoring inputs preserve the speaker position, preceding dialogue history, chat template, serialization, and next-utterance target. Only the current utterance argument differs.

H.6 TOKEN-MASK EXAMPLE

For a target of the form

```
{
  "action_type":  "speak",
  "argument":  "That timeline works for me.",
  "to":  ["Sophia"]
},
```

the next-utterance mask keeps tokenizer pieces corresponding to the natural-language content inside `argument`. JSON punctuation, field names, recipient tokens, padding, terminal tokens, and punctuation-only pieces receive zero weight.

H.7 GENERATION AND FAILURE RULES

Structured generation rejects malformed JSON objects, field-set violations, wrong field types, empty `speak` arguments, `leave` actions during OCPO sampled-utterance collection, exact duplicates, exposed-reasoning markers, token round-trip failures, empty argument masks, and prompt/response length violations. Generation attempts continue until eight distinct valid utterances are collected or the per-state attempt limit is reached. Incomplete states are discarded; optimizer steps with fewer than 12 complete groups are skipped.

H.8 BRANCH-LOCAL NEXT-UTTERANCE PROBE

The temporal-intervention analysis requires SWLR scoring after a branch has departed from the original trajectory. At these off-trajectory decision points, no original next utterance is available. We therefore use DeepSeek-V4-Flash only to construct a temporary branch-local probe turn.

Given the current branch state s, DeepSeek first acts in the focal role and generates a reference utterance $\tilde{a}$. Conditioned on the same state and $\tilde{a}$, DeepSeek then acts in the partner role and generates a next utterance $\tilde{y}$. The temporary pair

$$s \to \tilde{a} \to \tilde{y}$$

is never appended to the evaluated branch. We return to the unchanged state s, sample eight utterances $a_1, \ldots, a_8$ from the evaluated policy, and compute

$$r_i^{\mathrm{SWLR}} = \mathrm{SWLR}(s, a_i, \tilde{y}).$$

Thus, within a branch-local group, s and $\tilde{y}$ are fixed and only the sampled utterance varies. The high branch executes $\arg\max_i r_i^{\mathrm{SWLR}}$, whereas the low branch executes $\arg\min_i r_i^{\mathrm{SWLR}}$.

DeepSeek probe construction uses temperature 0 to reduce probe variance. The probe model is not used as the actual interaction partner: after the selected utterance is executed, the original partner agent generates the response that enters the real trajectory. DeepSeek also does not generate the evaluated sampled-utterance set, compute the SWLR likelihoods, or perform terminal evaluation.

H.9 EVALUATION INTERFACE

Main-task evaluation uses the official SOTOPIA `EpisodeLLMEvaluator`. For a two-agent episode, the text sent to GPT-4o follows the official template:

```
{history}

Based on previous interactions, evaluate how well
participants achieve their goals.
There are exactly 2 agents.  Under the 'evaluations' field,
use exactly these keys:  ["agent_1", "agent_2"] (no other
keys).
Please follow the format:
{format_instructions}
```

Here, {history} contains the scenario, the two participants' visible profiles and private social goals, the full dialogue history, and all speak and leave actions. Messages containing did nothing are removed before constructing the evaluator history.

The {format_instructions} field is generated from the Pydantic EvaluationForAgents[SotopiaDimensions] schema. The same schema is also passed to GPT-4o as the API structured-output response_format. For each of agent_1 and agent_2, the evaluator returns a reasoning string and a numeric score for each dimension:

```
{
  "evaluations": {
    "agent_1": {
      "believability": {"reasoning": "...", "score": 0},
      "relationship": {"reasoning": "...", "score": 0},
      "knowledge": {"reasoning": "...", "score": 0},
      "secret": {"reasoning": "...", "score": 0},
      "social_rules": {"reasoning": "...", "score": 0},
      "financial_and_material_benefits": {"reasoning":
"...", "score": 0},
      "goal": {"reasoning": "...", "score": 0}
    },
    "agent_2": {"...": "same seven dimensions"}
  }
}
```

The valid score ranges are 0–10 for Believability, Knowledge, and Goal; -5–5 for Relationship and Financial and Material Benefits; and -10–0 for Secret and Social Rules. Overall is the arithmetic mean of the seven dimension scores.

For the temporal-intervention experiment, each high/low branch receives three consecutive SWLR-guided focal-agent interventions. The first intervention uses the next utterance available at the original starting state. At the second and third decision points, DeepSeek-V4-Flash constructs a temporary branch-local reference turn solely to provide a fixed next utterance for SWLR scoring; this probe turn is discarded and never enters the evaluated dialogue. After each selected utterance is executed, the original partner agent supplies the response that advances the real branch. Following the third intervention, both branches continue naturally to termination with no further SWLR-based intervention. The high- and low-SWLR branches use the same original interaction agents, policy checkpoints, decoding parameters, intervention schedule, and terminal EpisodeLLMEvaluator.