NCCA: NESTED CREDIT WITH CONDITIONAL AMPLIFICATION FOR LLM-BASED CONVERSATIONAL RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Conversational recommender systems often use past dialogues to recover user preferences missing from a current request. We study conversational recommendation as a two-stage pipeline using large language models (LLMs) that first distills the dialogue history into a request-specific *user state* in natural language and then reranks item candidates from that state. Because downstream ranking quality can be computed automatically from observed feedback, the pipeline can be trained end-to-end with reinforcement learning with verifiable rewards (RLVR). A straightforward implementation assigns the same outcome advantage to both stages. In conversational recommendation, however, this signal mixes two sources of variation: state quality and stochastic ranking execution, with the latter especially strong when only a few candidate positions yield high reward. This coupling makes the feedback for each stage noisy: state updates inherit variation from stochastic ranking execution, while reranking updates vary with state quality. To solve this problem, we propose **NCCA** (**N**ested **C**redit with **C**onditional **A**mplification), a credit-assignment strategy for joint two-stage training. In NCCA, *Nested Credit* samples multiple states per request and multiple rankings per state, averages sibling rewards for state credit, and performs within-state relative credit for reranking. Thus, it suppresses execution noise upstream and isolates state-conditioned ranking variation downstream. Furthermore, *Conditional Amplification* upweights positive within-state advantages while retaining negative corrective updates. This strengthens the learning signal from the relatively few successful rankings without discarding corrections from lower-reward rankings. Our experiments show that NCCA improves NDCG@10 by 13.2% and 15.1% over the shared-advantage RLVR baseline on TG-ReDial and LLM-REDIAL, and enables a 4B model to match strong general-purpose LLMs.

1 INTRODUCTION

Conversational recommender systems (CRSs) elicit user needs through multi-turn natural-language dialogue and recommend items accordingly. Large language models (LLMs) enhance CRSs with language understanding, item knowledge, and preference reasoning, and can already serve as effective zero-shot conversational recommenders (He et al., 2023; Kim & Cho, 2026). Recent LLM-based CRSs increasingly incorporate information beyond the current dialogue, including cross-session memory, user profiles, and preference-oriented experience distilled from historical conversations (Xi et al., 2024; Huang et al., 2025; Zhu et al., 2025b). Such history can complement the current request with preferences that are not explicitly restated.

The challenge is to extract evidence relevant to the current request from history that may be long, redundant, and only partially relevant (Lin et al., 2024; Xi et al., 2024). LLM-based recommendation commonly handles history by directly conditioning on it, retrieving relevant records (Lin et al., 2024; Xi et al., 2024), or compressing it into a natural-language user profile (Zhou et al., 2024; Gao et al., 2025; Wu et al., 2025; Wang et al., 2025). Direct conditioning requires the model to filter the full record, while retrieval passes selected historical evidence downstream and leaves preference synthesis to decision time. The profile route instead summarizes history before recommendation. We study this route and term the request-specific profile a *user state*: the LLM maps dialogue history and the

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

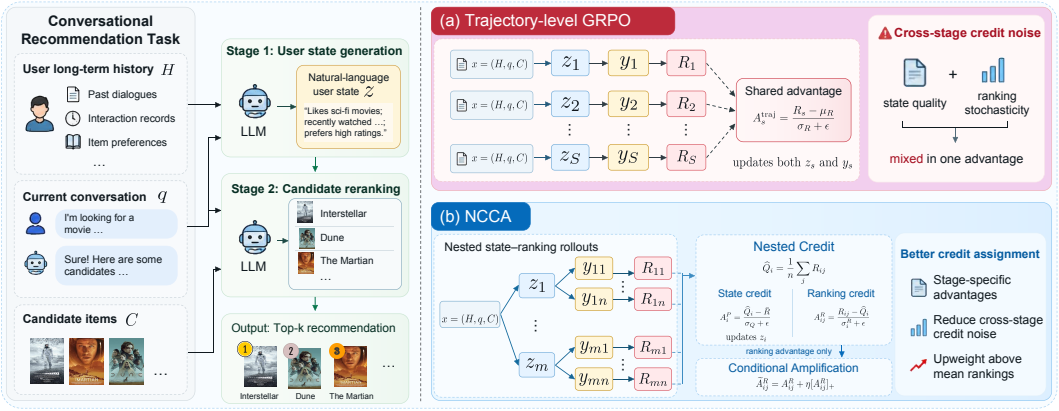


Figure 1: **Credit assignment in the two-stage pipeline.** (a) Our shared-advantage baseline (Traj-GRPO) samples independent state–ranking trajectories and applies one outcome advantage to both stages. (b) NCCA samples n reranking rollouts per state. *Nested Credit* credits each state by the mean reward of its rollouts and compares rankings only within the same state; *Conditional Amplification* gives extra weight to rankings above their state's mean reward.

current request into a state and then reranks candidate items from that state (Figure 1). This yields two coupled abilities—state inference and state-conditioned ranking—that must be optimized from the same downstream feedback.

Downstream feedback has been used to optimize language-based user representations: RLPF and LettinGo train summaries or profiles from prediction feedback, while LangPTune optimizes profiles for recommendation (Wu et al., 2025; Wang et al., 2025; Gao et al., 2025). RL with verifiable rewards (RLVR) instead optimizes policies from automatically checkable outcomes (Lambert et al., 2024); here, the ranking reward is computed from the candidate order and observed target. A straightforward GRPO-based (Shao et al., 2024) extension samples complete state–ranking rollouts, standardizes their terminal rewards, and assigns the same advantage to both generated stages, as in outcome-trained multi-step LLM policies (Yu et al., 2025; Jin et al., 2025). We call this shared-advantage baseline *Traj-GRPO*.

Outcome-level credit across multiple decisions—assigning a terminal reward back to the decisions that produced it—is a general RL problem (Kazemnejad et al., 2025; Jin et al., 2025). Our setting gives it a recommendation-specific structure: under a fixed user state, stochastic rerankings can still produce substantially different rewards, with relatively few rankings placing the relevant item near the top. Sharing one outcome advantage therefore mixes state quality with ranking execution: state updates inherit ranking noise, while reranking updates vary with state quality. We formalize this coupling through a cross-stage variance decomposition. Moreover, the right-skewed within-state reward means that positive advantage is concentrated in relatively few successful rankings, whereas negative advantage is spread across many similar low-reward rankings.

To address the above issues, we propose **NCCA** (**N**ested **C**redit with **C**onditional **A**mplification), a credit-assignment strategy for joint two-stage training. *Nested Credit* averages sibling ranking rewards to estimate state utility and centers reranking credit within each state, reducing execution noise upstream and state-quality variation downstream. *Conditional Amplification* then increases the weight of positive within-state advantages while retaining negative correction, strengthening the learning signal from successful rankings. We summarize our contributions as follows:

- We formulate credit assignment for two-stage history-conditioned conversational recommendation and derive a cross-stage variance decomposition of shared outcome credit.
- We propose NCCA, which combines nested state–ranking sampling for stage-specific credit with conditional amplification of successful within-state rankings, together with analyses of credit reliability and the induced policy gradient.
- On TG-ReDial and LLM-REDIAL, NCCA improves NDCG@10 over Traj-GRPO by 13.2% and 15.1%; diagnostics, ablations, and history interventions further support the proposed mechanism.

2 METHOD

2.1 PROBLEM FORMULATION

We cast memory-augmented conversational recommendation as two-stage candidate reranking. Each instance is $x = (H, q, C)$, where H is the user's long-term history (past dialogue sessions and interaction records), q the current request, and $C = \{c_1, \ldots, c_M\}$ a fixed candidate set. The positive-feedback set $G \subseteq C$ is used only to compute the reward. The model first writes a user state from the history and the request,

$$z \sim \pi_\theta^P(z \mid H, q), \tag{1}$$

and then, conditioned on this state, reranks the candidates:

$$y \sim \pi_\theta^R(y \mid z, q, C), \qquad y \in \mathrm{Perm}(C). \tag{2}$$

The state generator does not see C, so z has to describe the user rather than pre-select candidates, and the reranker sees H only through z. Both stages are produced by the same LLM with shared parameters θ, giving the joint policy $p_\theta(z, y \mid x) = \pi_\theta^P(z \mid H, q)\, \pi_\theta^R(y \mid z, q, C)$. With the terminal ranking reward $R = r(y, G)$, NDCG@10 in our experiments, we maximize

$$J(\theta) = \mathbb{E}_{(x,G)\sim\mathcal{D}}\, \mathbb{E}_{z\sim\pi_\theta^P}\, \mathbb{E}_{y\sim\pi_\theta^R(\cdot|z,q,C)}[r(y, G)]. \tag{3}$$

2.2 CROSS-STAGE VARIANCE DECOMPOSITION

RLVR provides a verifiable terminal reward but does not by itself specify how that reward should be assigned across intermediate decisions (Lambert et al., 2024; Kazemnejad et al., 2025). We instantiate a simple critic-free baseline with GRPO (Shao et al., 2024): for the same instance, we sample S complete trajectories $\tau_s = (z_s, y_s)$, where each rollout generates its own state and downstream ranking, standardize the S terminal rewards, and apply the resulting scalar advantage to both stages. We refer to this shared-advantage baseline as *Traj-GRPO*; similar outcome-level credit is used in multi-step LLM policies (Yu et al., 2025; Jin et al., 2025). The rewards are standardized as

$$A_s^{\mathrm{traj}} = \frac{R_s - \mu_R}{\sigma_R + \epsilon}, \qquad \mu_R = \frac{1}{S}\sum_{k=1}^{S} R_k. \tag{4}$$

Let $s_P = \nabla_\theta \log \pi_\theta^P(z \mid H, q)$ and $s_R = \nabla_\theta \log \pi_\theta^R(y \mid z, q, C)$. Ignoring clipping and importance ratios, the same scalar updates all generated tokens of both stages, $\nabla_\theta J_{\mathrm{traj}} \propto \mathbb{E}\big[A^{\mathrm{traj}}(s_P + s_R)\big]$. Rollouts are drawn from $\pi_{\theta_{\mathrm{old}}}$; we analyze the on-policy case $\theta = \theta_{\mathrm{old}}$ and drop the subscript until Section 2.3.

Each stage needs a different part of the reward. Define the downstream utility of a state,

$$Q_\theta^P(x, z) = \mathbb{E}_{y\sim\pi_\theta^R(\cdot|z,q,C)}[R], \tag{5}$$

which depends on x because x supplies q and C to the reranker. The score-function identity (Appendix A.1) splits the policy gradient into

$$g_P = \mathbb{E}\big[Q_\theta^P(x, z)\, s_P\big], \qquad g_R = \mathbb{E}\big[\big(R - \mathrm{sg}[Q_\theta^P(x, z)]\big)\, s_R\big], \tag{6}$$

where $\mathrm{sg}[\cdot]$ is stop-gradient. For the baseline $b(x) = \mathbb{E}[Q_\theta^P(x, z) \mid x]$, the centered reward decomposes as

$$R - b = \underbrace{\big(Q_\theta^P(x, z) - b\big)}_{\text{state component}} + \underbrace{\big(R - Q_\theta^P(x, z)\big)}_{\text{execution component}}. \tag{7}$$

The state component measures how much more downstream utility a state supports than an average state; the execution component measures how far a sampled ranking lands from what its state supports on average. By Eq. 6, g_P needs only the first and g_R only the second. The law of total variance gives their sizes:

$$\mathrm{Var}(R \mid x) = \underbrace{\mathrm{Var}_z\big(Q_\theta^P(x, z)\big)}_{\sigma_B^2(x)} + \underbrace{\mathbb{E}_z\big[\sigma_W^2(x, z)\big]}_{\sigma_W^2(x)}, \qquad \rho(x) = \frac{\sigma_B^2(x)}{\sigma_B^2(x) + \sigma_W^2(x)}, \tag{8}$$

with $\sigma_W^2(x,z) = \mathrm{Var}(R \mid x, z)$. The intra-state correlation ρ is the share of reward variance explained by the state. With few relevant items per candidate set we expect it to be small, and Section 3.4 measures $\rho \approx 0.3$.

The shared trajectory advantage feeds both components to both stages. The following proposition quantifies this mixture at the state–ranking boundary.

Proposition 1 (Cross-stage variance decomposition)**.** *For the shared-outcome estimator under the two-stage factorization above, $\mathbb{E}[(R-b)\, s_P] = g_P$ and $\mathbb{E}[(R-b)\, s_R] = g_R$, while each stage carries the other stage's component as additional variance:*

$$\mathrm{tr}\,\mathrm{Cov}\big((R-b)\, s_P\big) = \mathrm{tr}\,\mathrm{Cov}\big((Q_\theta^P - b)\, s_P\big) + \mathbb{E}\big[\sigma_W^2(x,z)\, \|s_P\|^2\big], \tag{9}$$

$$\mathrm{tr}\,\mathrm{Cov}\big((R-b)\, s_R\big) = \mathrm{tr}\,\mathrm{Cov}\big((R - Q_\theta^P)\, s_R\big) + \mathbb{E}\big[(Q_\theta^P - b)^2\, \|s_R\|^2\big], \tag{10}$$

where Eq. 10 uses the simplifying assumption that $R - Q_\theta^P$ is uncorrelated with $\|s_R\|^2$ given (x,z).

For this two-stage recommender, the corresponding unstandardized shared-outcome estimator has the correct expectation but carries cross-stage variance: the upstream gradient includes execution variance σ_W^2, while the downstream gradient includes the spread of state utilities. When ρ is small, ranking execution dominates the reward variation available to the state generator. Population normalization would leave the state component with standard deviation $\sqrt{\rho}$; empirical group standardization likewise rescales the mixture without separating its two components. Proofs are in Appendix A.2.

2.3 Nested Credit

Proposition 1 suggests giving each stage only its own component. Neither is observable from one trajectory because Q^P is unknown, so we estimate it by nested sampling. For each instance we draw m states and n reranking rollouts per state:

$$z_i \sim \pi_{\theta_{\mathrm{old}}}^P(\cdot \mid H, q), \quad y_{ij} \sim \pi_{\theta_{\mathrm{old}}}^R(\cdot \mid z_i, q, C), \quad R_{ij} = r(y_{ij}, G), \quad i \in [m],\ j \in [n]. \tag{11}$$

The rollout mean $\widehat{Q}_i = \frac{1}{n}\sum_j R_{ij}$ is a Monte Carlo estimate of $Q_{\theta_{\mathrm{old}}}^P(x, z_i)$. With $\bar{R} = \frac{1}{m}\sum_i \widehat{Q}_i$, the sample analogue of Eq. 7 is

$$R_{ij} - \bar{R} = \big(\widehat{Q}_i - \bar{R}\big) + \big(R_{ij} - \widehat{Q}_i\big), \tag{12}$$

and the two terms are orthogonal within an instance because $\sum_j (R_{ij} - \widehat{Q}_i) = 0$. The first term becomes the state advantage, standardized across states, and the second the reranking advantage, standardized among rollouts of the same state:

$$A_i^P = \frac{\widehat{Q}_i - \bar{R}}{\sigma_Q + \epsilon}, \qquad A_{ij}^R = \frac{R_{ij} - \widehat{Q}_i}{\sigma_i^R + \epsilon}, \tag{13}$$

where σ_Q and σ_i^R are the empirical standard deviations of $\{\widehat{Q}_k\}_{k=1}^m$ and $\{R_{i\ell}\}_{\ell=1}^n$.

Proposition 2 (Credit reliability under nested sampling)**.** *Fix x, and let rollouts be conditionally independent given (x,z) with constant execution variance σ_W^2. Before standardization,*

(i) the execution noise left in the upstream estimator has variance $\mathbb{E}[\sigma_W^2 \|s_P\|^2]/n$;

(ii) the state credit satisfies

$$\mathrm{corr}^2\left(\widehat{Q}_i,\, Q_{\theta_{\mathrm{old}}}^P(x, z_i)\right) = \frac{n\rho}{1 + (n-1)\rho}; \tag{14}$$

(iii) the within-state term $R_{ij} - \widehat{Q}_i$ contains no state component, and equals $\frac{n-1}{n}$ times the leave-one-out deviation $R_{ij} - \widehat{Q}_i^{(-j)}$, whose baseline is independent of y_{ij}.

Part (ii) is the Spearman–Brown relation for the mean of n parallel measurements. At $n = 1$ it reduces to ρ, the reliability of a single trajectory reward; at the measured $\rho \approx 0.3$, $n = 4$ roughly doubles it. Part (iii) shows that the downstream contamination in Eq. 10 is removed exactly from the numerator of A_{ij}^R (proofs in Appendix A.3). Dividing by σ_i^R scales all rollouts of a state by one positive factor, which keeps their signs and order; since the factor is computed from the same rewards, unbiasedness holds only up to this per-state reweighting, as for the group standard deviation of GRPO (Liu et al., 2025). Nested Credit requires $n \geq 2$; the split of a budget $B = mn$ is analyzed in Appendix B.

2.4 CONDITIONAL AMPLIFICATION

Nested Credit fixes what each stage is credited with, but not how the reranker weighs better and worse rankings from the same state. In sparse-positive candidate reranking, the within-state reward can be right-skewed, as we observe empirically (Section 3.4): a few rankings place the target near the top, while many lower-reward rankings differ little from each other. Because a standardized within-state advantage sums to zero, its positive and negative sides receive equal aggregate advantage mass. In this setting, the positive mass is carried by relatively few successful rankings, whereas the negative mass is spread across many weak distinctions; a push-down further redistributes removed probability over the other permutations in proportion to the current policy (Lemma 1, Appendix A.5).

We therefore weight the two sides of the reranking advantage separately,

$$\Psi_{\alpha_+,\alpha_-}(A) = \alpha_+[A]_+ - \alpha_-[A]_-, \qquad [A]_+ = \max(A, 0), \;\; [A]_- = \max(-A, 0). \tag{15}$$

A common rescaling of α_+ and α_- is equivalent to rescaling the stage weight λ_R in Eq. 19, so the ratio α_+/α_- is the only new degree of freedom. Fixing $\alpha_- = 1$ and $\alpha_+ = 1 + \eta$ with $\eta \geq 0$ gives

$$\widetilde{A}^R_{ij} = \Psi_{1+\eta,1}\big(A^R_{ij}\big) = A^R_{ij} + \eta\,[A^R_{ij}]_+, \tag{16}$$

which amplifies above-baseline rankings by $1 + \eta$ and keeps the corrective gradient of below-baseline rankings at unit weight. The resulting change of the expected update has a simple form.

Proposition 3 (What amplification optimizes)**.** *Fix (x, z), let $e = R - Q^P_\theta(x, z)$ with $\Pr(e = 0) = 0$, $p_+ = \Pr(e > 0 \mid x, z)$, and $M(x, z) = \mathbb{E}[|e| \mid x, z]$, the within-state mean absolute deviation of the reward. Before standardization, with ∇^R differentiating through π^R_θ only,*

$$\mathbb{E}\big[(e + \eta[e]_+)\, s_R \mid x, z\big] = (1 + \eta p_+)\, \nabla^R Q^P_\theta(x, z) + \tfrac{\eta}{2}\, \nabla^R M(x, z). \tag{17}$$

The first term preserves the exact reranking gradient up to a positive scale in $[1, 1 + \eta]$; the only additional bias is the second term, which increases the within-state mean absolute reward deviation (proof in Appendix A.4). Thus amplification strengthens the policy-gradient signal from above-baseline rankings while adding controlled pressure to maintain reward separation within the state-conditioned group. A moderate η can preserve usable group-relative signal when many rewards cluster together, while a large η can overemphasize dispersion relative to mean ranking quality (Section 3.6).

The amplification is *conditional* because it acts after within-state centering: extra weight is allocated according to ranking improvement relative to sibling rollouts, not according to the absolute trajectory reward. In our right-skewed reranking setting, this shifts additional gradient mass toward the scarce high-reward rankings while retaining the corrective gradient on lower-reward rankings. If the same asymmetric weight is applied to the shared trajectory advantage, e in Eq. 17 is replaced by $R - b$ and the amplified term again mixes state-level and ranking-level variation; our frozen-policy diagnostic measures this mixture directly (Section 3.4). Thus the benefit of asymmetric weighting (Oh et al., 2018; Arnal et al., 2025) is substantially stronger after Nested Credit isolates within-state ranking credit (Section 3.5). We do not amplify A^P_i, since $\widehat{Q}_i$ is an average and far less skewed.

2.5 TRAINING OBJECTIVE

The stage-specific advantages enter the clipped GRPO surrogate. For a sampled sequence τ with advantage A,

$$\mathcal{L}_{\text{clip}}(\tau, A) = \frac{1}{|\tau|}\sum_{t=1}^{|\tau|} \min\big(w_t(\theta)A,\, \text{clip}(w_t(\theta), 1 - \epsilon_c, 1 + \epsilon_c)A\big), \qquad w_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}, \tag{18}$$

and the NCCA objective is

$$J_{\text{NCCA}} = \lambda_P \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}_{\text{clip}}(z_i, A^P_i) + \lambda_R \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} \mathcal{L}_{\text{clip}}(y_{ij}, \widetilde{A}^R_{ij}), \tag{19}$$

with $\lambda_P = 1/(1 + n)$ and $\lambda_R = n/(1 + n)$, so that each state and ranking sequence carries equal weight. Setting $\lambda_R = 0$ removes the explicit reranking loss; because the two stages share parameters, it does not freeze the reranker. NCCA addresses the setting in which both stage-specific losses are optimized from the same reward. The KL term to the reference policy is omitted for readability. NCCA adds two hyperparameters, the allocation (m, n) and the amplification weight η.

3 EXPERIMENTS

We ask five questions. **RQ1**: Does NCCA improve over SFT and our shared-advantage baseline, Traj-GRPO? **RQ2**: Does the gain come from what the state says or only from the extra generation step? **RQ3**: Is the cross-stage variance large in practice, and does nested estimation improve state credit as predicted? **RQ4**: How do the two mechanisms contribute? **RQ5**: How sensitive is NCCA to η and to the split of mn?

3.1 EXPERIMENTAL SETUP

Datasets. TG-ReDial is a Chinese movie CRS benchmark with topic-guided multi-turn dialogues (Zhou et al., 2020). LLM-REDIAL is an English multi-domain benchmark (Books, Movies, Electronics, Sports) whose dialogues are grounded in Amazon user behavior (Liang et al., 2024). Following the sequential CRS setting of MemoCRS (Xi et al., 2024), the last eligible session of each user is the target; H holds the earlier sessions and interaction records, and the last user utterance before the target recommendation is the request q. Each instance has one accepted target and 19 in-domain negatives, including history-sensitive hard negatives, and all methods rerank the same frozen candidate sets (Appendix C.1). We report Recall@K (R@K, the hit rate with one relevant item) and NDCG@K (N@K) for $K \in \{3, 5, 10\}$.

Baselines. *Sequential recommenders*: GRU4Rec (Hidasi et al., 2016), SASRec (Kang & McAuley, 2018), and BERT4Rec (Sun et al., 2019), trained on item-ID sequences from the same histories and scoring the same candidates (Appendix C.2). *General-purpose LLMs*: DeepSeek-V4-Flash and GPT-5.6 Luna, given the same prompts, histories, and candidates as our model. *Trained LLMs*, all on Qwen3-4B-Thinking-2507 (Qwen Team, 2025): the backbone without task training (Base); an SFT model trained on GPT-5.4-distilled demonstrations; GRPO without a user state, which reranks directly from (H, q, C); and trajectory-level GRPO (Traj-GRPO) with 16 independent trajectories, i.e., $(m, n) = (16, 1)$. All RL methods start from Base and share data, reward, optimization steps, and the budget of 16 terminal reranking evaluations per instance; NCCA generates 4 rather than 16 states per instance, so its generation cost is lower.

Implementation and evaluation protocol. RL uses eight GPUs, global batch size 128, and learning rate 1×10^{-6}; NCCA uses $(m, n) = (4, 4)$ and $\eta = 0.3$ unless stated otherwise. Both stages sample with temperature 0.7 and top-p 0.95, with up to 8,192 prompt and response tokens. Each trained LLM uses two training seeds. At test time, every LLM is decoded at temperature 0.3, averaging 8 stochastic decodes per test instance; sequential recommenders use 8 training seeds. We report mean and standard deviation under this evaluation protocol (Appendix C.3).

3.2 OVERALL PERFORMANCE (RQ1)

Table 1 shows three findings. **(i) Under shared trajectory credit, the user state brings no gain.** Traj-GRPO on the two-stage pipeline and GRPO without a user state, which reranks directly from the raw history, reach the same N@10 within a tenth of a standard deviation (0.4015 vs. 0.4002 and 0.3984 vs. 0.3991). This pair ablates the state interface: with one advantage per trajectory, the interface goes unused, and neither method consistently beats SFT. **(ii) NCCA is the strongest trained method.** It is best among all trained methods on all twelve metrics, and improves over Traj-GRPO by 6.2% R@10 and 13.2% N@10 on TG-ReDial and by 10.2% and 15.1% on LLM-REDIAL; the N@10 gaps are 3.7 and 4.7 pooled standard deviations. **(iii) A 4B model trained with NCCA is competitive with strong general-purpose LLMs.** NCCA has the highest mean on 8 of the 12 reported metrics, including N@10 on both datasets. Validation curves (Appendix C.8) show that the gap to Traj-GRPO is not a checkpoint-selection artifact: Traj-GRPO's best checkpoint stays below NCCA.

3.3 DOES THE STATE CARRY HISTORY? (RQ2)

On TG-ReDial, we hold the reranker, candidate sets, and decoding fixed and change only what the state is generated from: the true history (*Full*), the request alone (*No-history*), another user's history (*Mismatched*), or no state at all, reranking from (q, C) (*No-state*). Mismatched keeps the length,

Table 1: **Recommendation performance**, mean ± standard deviation (protocol in Section 3.1). Best per column and dataset in **bold**, second underlined. Trained LLMs use Qwen3-4B-Thinking.

Method	R@3	R@5	R@10	N@3	N@5	N@10
TG-ReDial						
GRU4Rec	0.3376 ±.0209	0.5010 ±.0223	0.6671 ±.0133	0.2435 ±.0166	0.3104 ±.0144	0.3652 ±.0099
SASRec	0.3023 ±.0270	0.4658 ±.0202	0.6473 ±.0176	0.2302 ±.0219	0.2969 ±.0177	0.3562 ±.0128
BERT4Rec	0.3566 ±.0220	0.4735 ±.0287	0.6899 ±.0228	0.2712 ±.0166	0.3195 ±.0162	0.3920 ±.0143
DeepSeek-V4-Flash	0.3642 ±.0122	0.5258 ±.0164	0.7780 ±.0068	0.2844 ±.0112	0.3504 ±.0121	0.4319 ±.0074
GPT-5.6 Luna	0.3546 ±.0234	**0.5452** ±.0292	**0.8168** ±.0110	0.2750 ±.0202	0.3511 ±.0219	0.4366 ±.0157
Base (Qwen3-4B-Thinking)	0.2791 ±.0257	0.4291 ±.0337	0.7314 ±.0288	0.2132 ±.0216	0.2743 ±.0188	0.3718 ±.0148
+SFT	0.3069 ±.0392	0.4891 ±.0387	0.7826 ±.0166	0.2247 ±.0277	0.2996 ±.0263	0.3944 ±.0148
+GRPO (no state)	0.3161 ±.0172	0.4843 ±.0283	0.7469 ±.0289	0.2470 ±.0159	0.3152 ±.0166	0.4002 ±.0134
+Traj-GRPO	0.3138 ±.0267	0.4894 ±.0318	0.7546 ±.0309	0.2443 ±.0201	0.3161 ±.0191	0.4015 ±.0155
+NCCA (ours)	**0.3950** ±.0270	0.5437 ±.0212	0.8016 ±.0059	**0.3098** ±.0216	**0.3706** ±.0178	**0.4543** ±.0130
LLM-REDIAL						
GRU4Rec	0.2857 ±.0189	0.4039 ±.0198	0.6946 ±.0305	0.2045 ±.0261	0.2535 ±.0243	0.3461 ±.0172
SASRec	0.2365 ±.0288	0.3399 ±.0166	0.6379 ±.0120	0.1691 ±.0253	0.2111 ±.0179	0.3057 ±.0132
BERT4Rec	0.2512 ±.0210	0.3842 ±.0205	0.6453 ±.0276	0.1736 ±.0178	0.2286 ±.0158	0.3117 ±.0166
DeepSeek-V4-Flash	**0.4350** ±.0161	0.5594 ±.0130	0.7897 ±.0166	0.3021 ±.0137	0.3527 ±.0121	0.4236 ±.0113
GPT-5.6 Luna	0.4055 ±.0111	0.5568 ±.0132	0.8018 ±.0149	**0.3102** ±.0080	0.3722 ±.0088	0.4515 ±.0083
Base (Qwen3-4B-Thinking)	0.2976 ±.0200	0.4482 ±.0243	0.7165 ±.0224	0.2142 ±.0162	0.2759 ±.0138	0.3624 ±.0173
+SFT	0.3490 ±.0158	0.4763 ±.0171	0.7441 ±.0144	0.2686 ±.0124	0.3207 ±.0135	0.4067 ±.0125
+GRPO (no state)	0.3418 ±.0237	0.4866 ±.0213	0.7313 ±.0201	0.2608 ±.0181	0.3201 ±.0162	0.3991 ±.0131
+Traj-GRPO	0.3438 ±.0151	0.4911 ±.0152	0.7507 ±.0200	0.2539 ±.0133	0.3144 ±.0124	0.3984 ±.0098
+NCCA (ours)	0.4032 ±.0282	**0.5657** ±.0146	**0.8274** ±.0202	0.3072 ±.0186	**0.3738** ±.0135	**0.4585** ±.0151

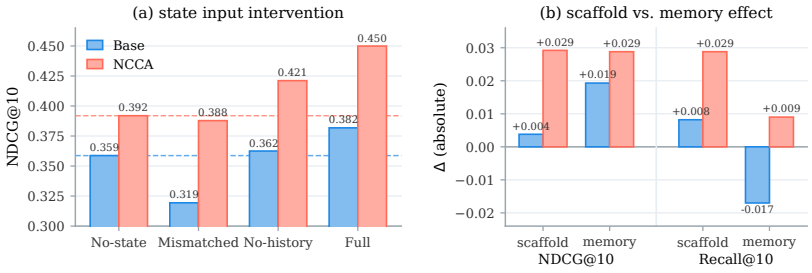


Figure 2: **Intervention on the state input (TG-ReDial).** (a) NDCG@10 per arm; dashed lines: No-state level. (b) Gain of Full over No-state split into a *scaffold* (No-history − No-state) and a *memory* effect (Full − No-history). Full results in Appendix C.7.

format, and extra computation of a state and removes only its relevance to the user. All arms use one sampled decoding pass rather than the 8-sample average, so absolute values differ slightly from Table 1.

The state transmits user-specific content. Generating the state from another user's history lowers N@10 by 13.8% and R@10 by 13.0% for NCCA, and by 16.3% and 12.4% for Base. For both models the mismatched state also falls below the No-state level (Figure 2a): the reranker uses what the state says.

Training strengthens both routes through the interface. Before training, the scaffold effect is negligible (+0.004 N@10) and adding history even lowers R@10 (−0.017; Figure 2b). After NCCA training, the scaffold effect grows to +0.029 N@10, and the memory effect, the reliance on history, grows from +0.019 to +0.029 N@10 and turns positive on R@10 (+0.009).

3.4 Cross-Stage Variance and Credit Reliability (RQ3)

We test the mechanism on frozen policies. For each instance we sample 16 states and 32 reranking rollouts per state at the training temperature, and simulate every credit scheme offline on this 16×32

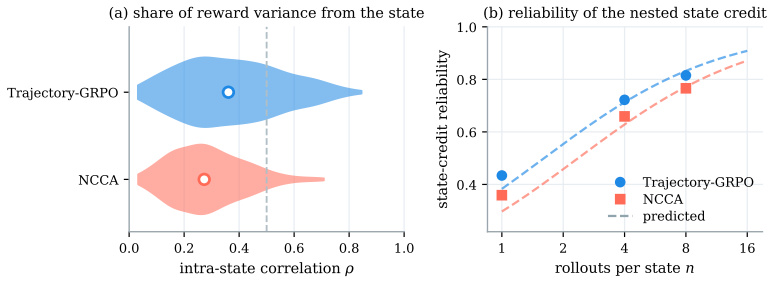


Figure 3: **Credit on frozen policies.** (a) Per-instance ρ (Eq. 8); circles: medians. (b) Reliability $\mathrm{corr}^2(\widehat{Q}^{(n)}, Q)$ of the state credit, measured (points) and predicted by Proposition 2(ii) at the pool's mean ρ (dashed).

Table 2: **Ablation.** Without NC: $(m, n) = (16, 1)$; with NC: $(4, 4)$; Amp.: $\eta = 0.3$. Relative change over the first row in parentheses.

		TG-ReDial		LLM-REDIAL	
NC	**Amp.**	**R@10**	**N@10**	**R@10**	**N@10**
✗	✗	0.7546	0.4015	0.7507	0.3984
✗	✓	0.7794 (+3.3%)	0.4178 (+4.1%)	0.7616 (+1.5%)	0.4019 (+0.9%)
✓	✗	0.7753 (+2.7%)	0.4302 (+7.1%)	0.7480 (−0.4%)	0.3997 (+0.3%)
✓	✓	**0.8016** (+6.2%)	**0.4543** (+13.2%)	**0.8274** (+10.2%)	**0.4585** (+15.1%)

reward pool: Traj-GRPO credit uses one rollout per state, whereas Nested Credit averages n. Pools come from the Traj-GRPO and NCCA checkpoints.

Execution noise dominates the reward. A one-way random-effects fit over states gives a median ρ of 0.36 on the Traj-GRPO pool and 0.27 on the NCCA pool (means 0.38 and 0.30; Figure 3a). Under a fixed state, the sampled ranking accounts for roughly two thirds of the reward variance, and its reward is right-skewed (within-state skewness 0.94 and 0.50), as assumed in Section 2.4. Under Traj-GRPO credit, 19.7% and 17.3% of rankings below their own state's oracle utility still receive a positive advantage, and about 21.5% of those above it a negative one: roughly one advantage in five has its sign set by reranking stochasticity.

Nested Credit recovers the state signal. We split the 32 rollouts of each state into a 16-draw oracle half defining Q^{oracle} and a disjoint half from which $\widehat{Q}^{(n)}$ is drawn (1,000 random splits), and report the instance-centered corr^2. Reliability rises from 0.43 at $n = 1$ to 0.72 at $n = 4$ and 0.82 at $n = 8$ on the Traj-GRPO pool, and from 0.36 to 0.66 and 0.77 on the NCCA pool (Figure 3b), within 0.03 of Proposition 2 for $n \geq 4$ (4–6 points above it at $n = 1$, as expected when ρ varies across instances). Four rollouts cut the credit MSE by about 70% (Appendix C.5).

3.5 Ablation: Nested Credit and Conditional Amplification (RQ4)

Removing both mechanisms recovers Traj-GRPO (Table 2). Either alone helps on TG-ReDial (N@10 +7.1% and +4.1%) but stays within 1% on LLM-REDIAL; together they give the largest gains on every metric. The effect of amplification depends on the credit it amplifies: on LLM-REDIAL, adding it to the shared trajectory credit changes N@10 by +0.9%, and adding it to Nested Credit by +14.7%. The frozen pools show why (Appendix C.6): 42–49% of the positive mass of the flat trajectory advantage comes from its state part, and 28–31% of positive advantages start from a below-mean state, so amplification of the shared trajectory advantage strengthens a mixture of state-level and ranking- level variation. Under Nested Credit, the amplified term is purely within-state by construction, so the extra positive weight acts on state-conditioned reranking credit rather than on this cross-stage mixture.

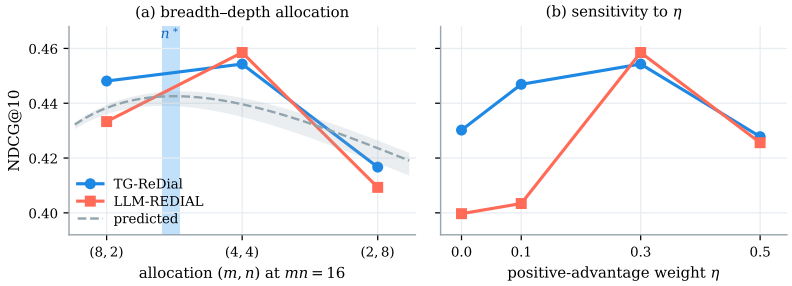


Figure 4: **Sensitivity.** (a) NDCG@10 at $mn = 16$, $\eta = 0.3$; dashed: Eq. 30 (rescaled); band: $n^* = 1 + \rho^{-1/2}$. (b) NDCG@10 versus η at $(4, 4)$ (Appendix C.4).

3.6 Sensitivity and Breadth–Depth Allocation (RQ5)

Amplification strength. Moderate amplification improves N@10 over $\eta = 0$, and $\eta = 0.3$ gives the highest N@10 on both datasets (Figure 4b). At $\eta = 0.5$, N@10 drops while R@10 on TG-ReDial still rises, as expected once the dispersion term of Proposition 3 competes with ranking.

Breadth versus depth. With $mn = 16$, the balanced $(4, 4)$ gives the best N@10 on both datasets (Figure 4a). Going deep with $(2, 8)$ costs 8.3% and 10.7%: with two states, the standardized state advantage is ± 1. Going wide with $(8, 2)$ is close on TG-ReDial but loses 5.5% on LLM-REDIAL, since two rollouts give a noisy state estimate and a binary within-state comparison. This shape is in line with Appendix B, which places the best depth between $n = 2$ and $n = 4$ and predicts a clear loss at $n = 8$; we use $(4, 4)$ as the default.

4 Related Work

LLM-based conversational recommendation with memory. LLMs serve as conversational recommenders (He et al., 2023; Kim & Cho, 2026). Cross-session CRS methods retrieve or refine historical information (Xi et al., 2024), while general LLM memory methods maintain information across long-term interactions (Zhong et al., 2024). Language-based methods compress history into natural-language profiles (Zhou et al., 2024; Zhang et al., 2026). RLPF, LettinGo, and LangPTune optimize such representations with downstream feedback (Wu et al., 2025; Wang et al., 2025; Gao et al., 2025). RL has also been used to optimize recommendation outputs (Tan et al., 2025; Zhu et al., 2026) and retrieval (Yue et al., 2026). NCCA instead assigns one terminal ranking reward across the profile and reranking stages.

Credit assignment and sample weighting in RLVR. RLVR trains LLM policies from automatically verifiable task rewards (Lambert et al., 2024). Delayed outcome credit across multiple generated decisions is a general problem; multi-step LLM policies often share one trajectory advantage across segments (Yu et al., 2025; Jin et al., 2025). Finer credit can use a learned critic or Monte Carlo values branched at intermediate points, as in VinePPO and TreeRL (Kazemnejad et al., 2025; Hou et al., 2025); NCCA places such estimation at the state–ranking boundary. GiGPO groups recurring environment states (Feng et al., 2025), whereas free-text user states do not recur verbatim, so we create sibling rollouts conditionally from each sampled state. Prior work also studies asymmetric weighting of positive and negative samples (Oh et al., 2018; Zhu et al., 2025a; Arnal et al., 2025); we apply it after within-state credit and characterize the induced objective.

5 Conclusion

When an LLM recommender writes a user state and ranks from it, one ranking reward must train two decisions. NCCA uses nested reranking rollouts for stage-specific credit and amplifies informative positive within-state advantages, making this memory-to-decision interface trainable.

AI USE STATEMENT

Generative AI tools were used to assist with translation, language editing, and polishing. All AI-assisted text was reviewed and verified by the authors, who take full responsibility for the final content of this work.

REFERENCES

Charles Arnal, Gaëtan Narozniak, Vivien Cabannes, Yunhao Tang, Julia Kempe, and Remi Munos. Asymmetric REINFORCE for off-policy reinforcement learning: Balancing positive and negative rewards. In *Advances in Neural Information Processing Systems*, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, 2025.

Zhaolin Gao, Joyce Zhou, Yijia Dai, and Thorsten Joachims. Langptune: Optimizing language-based user profiles for recommendation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 707–717. ACM, 2025. doi: 10.1145/3746252.3761369.

Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 720–730, 2023. doi: 10.1145/3583780.3614949.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. TreeRL: LLM reinforcement learning with on-policy tree search. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025.

Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. Recommender AI agent: Integrating large language models for interactive recommendations. *ACM Transactions on Information Systems*, 43(4):96:1–96:33, 2025. doi: 10.1145/3731446.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining*, pp. 197–206. IEEE, 2018. doi: 10.1109/ICDM.2018.00035.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 29557–29590. PMLR, 2025.

Yeong-Hyeon Kim and Yoon-Sik Cho. Conversational recommender systems with large language models: A review of current developments. *Expert Systems with Applications*, 324:132508, 2026. doi: 10.1016/j.eswa.2026.132508.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Tingting Liang, Chenxin Jin, Lingzhi Wang, Wenqi Fan, Congying Xia, Kai Chen, and Yuyu Yin. LLM-REDIAL: A large-scale dataset for conversational recommender systems created from user behaviors with LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 8926–8939. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.529.

Jianghao Lin, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. ReLLa: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation. In *Proceedings of the ACM Web Conference 2024*, pp. 3497–3508. ACM, 2024. doi: 10.1145/3589334.3645467.

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. *arXiv preprint arXiv:2503.20783*, 2025.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3878–3887. PMLR, 2018.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1441–1450. ACM, 2019. doi: 10.1145/3357384.3357895.

Junfei Tan, Yuxin Chen, An Zhang, Junguang Jiang, Bin Liu, Ziru Xu, Han Zhu, Jian Xu, Bo Zheng, and Xiang Wang. Reinforced preference optimization for recommendation. *arXiv preprint arXiv:2510.12211*, 2025.

Lu Wang, Di Zhang, Fangkai Yang, Pu Zhao, Jianfeng Liu, Yuefeng Zhan, Hao Sun, Qingwei Lin, Weiwei Deng, Dongmei Zhang, Feng Sun, and Qi Zhang. Lettingo: Explore user profile generation for recommendation system. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 2985–2995. ACM, 2025. doi: 10.1145/3711896.3737024.

Jiaxing Wu, Lin Ning, Luyang Liu, Harrison Lee, Neo Wu, Chao Wang, Sushant Prakash, Shawn O'Banion, Bradley Green, and Jun Xie. Rlpf: Reinforcement learning from prediction feedback for user summarization with llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25488–25496, 2025. doi: 10.1609/aaai.v39i24.34738.

Yunjia Xi, Weiwen Liu, Jianghao Lin, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. Memocrs: Memory-enhanced sequential conversational recommender systems with large language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 2585–2595. ACM, 2024. doi: 10.1145/3627673.3679599.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. MemAgent: Reshaping long-context LLM with multi-conv RL-based memory agent. *arXiv preprint arXiv:2507.02259*, 2025.

Zhenrui Yue, Honglei Zhuang, Zhen Qin, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. Retrieval augmented conversational recommendation with reinforcement learning. *arXiv preprint arXiv:2604.04457*, 2026.

Yi Zhang, Yiwen Zhang, Yu Wang, Tong Chen, and Hongzhi Yin. Proex: A unified framework leveraging large language model with profile extrapolation for recommendation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1940–1951. ACM, 2026. doi: 10.1145/3770854.3780284.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024. doi: 10.1609/AAAI.V38I17.29946.

Joyce Zhou, Yijia Dai, and Thorsten Joachims. Language-based user profiles for recommendation. *arXiv preprint arXiv:2402.15623*, 2024.

Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. Towards topic-guided conversational recommender system. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 4128–4139. International Committee on Computational Linguistics, 2020. doi: 10.18653/v1/2020.coling-main.365.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in LLM reasoning. In *Advances in Neural Information Processing Systems*, 2025a.

Yaochen Zhu, Harald Steck, Dawen Liang, Yinhan He, Nathan Kallus, and Jundong Li. LLM-based conversational recommendation agents with collaborative verbalized experience. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 2207–2220. Association for Computational Linguistics, 2025b. doi: 10.18653/v1/2025.findings-emnlp.119.

Yaochen Zhu, Harald Steck, Dawen Liang, Yinhan He, Vito Ostuni, Jundong Li, and Nathan Kallus. Rank-grpo: Training llm-based conversational recommender systems with reinforcement learning. In *International Conference on Learning Representations*, 2026.

A PROOFS

Throughout, expectations are over $(x, G) \sim \mathcal{D}$, $z \sim \pi_\theta^P(\cdot \mid H, q)$, and $y \sim \pi_\theta^R(\cdot \mid z, q, C)$ unless stated otherwise. We write $Q = Q_\theta^P(x, z)$, $b = b(x) = \mathbb{E}[Q \mid x]$, and $e = R - Q$ for the execution component. By definition of Q,

$$\mathbb{E}[e \mid x, z] = 0, \qquad \mathbb{E}[e^2 \mid x, z] = \sigma_W^2(x, z). \tag{20}$$

We also use the score identities $\mathbb{E}[s_P \mid x] = 0$ and $\mathbb{E}[s_R \mid x, z] = 0$.

A.1 NESTED SCORE-FUNCTION DECOMPOSITION

Since $\log p_\theta(z, y \mid x) = \log \pi_\theta^P(z \mid H, q) + \log \pi_\theta^R(y \mid z, q, C)$, the log-derivative identity gives

$$\nabla_\theta J = \mathbb{E}\big[R\, s_P\big] + \mathbb{E}\big[R\, s_R\big]. \tag{21}$$

For the first term, s_P depends only on (x, z), so $\mathbb{E}[R\, s_P] = \mathbb{E}\big[\mathbb{E}[R \mid x, z]\, s_P\big] = \mathbb{E}[Q\, s_P] = g_P$. For the second term, Q depends only on (x, z), so $\mathbb{E}[Q\, s_R] = \mathbb{E}\big[Q\, \mathbb{E}[s_R \mid x, z]\big] = 0$ and $\mathbb{E}[R\, s_R] = \mathbb{E}[(R - Q)\, s_R] = g_R$, which is Eq. 6. No assumption of separate parameters is needed: with shared parameters, both terms contribute to the same vector.

A.2 PROOF OF PROPOSITION 1

Unbiasedness. Since b depends only on x, $\mathbb{E}[b\, s_P] = \mathbb{E}[b\, \mathbb{E}[s_P \mid x]] = 0$, so $\mathbb{E}[(R - b)s_P] = \mathbb{E}[R\, s_P] = g_P$. Likewise $\mathbb{E}[(Q - b)s_R] = \mathbb{E}\big[(Q - b)\, \mathbb{E}[s_R \mid x, z]\big] = 0$, so $\mathbb{E}[(R - b)s_R] = \mathbb{E}[(R - Q)s_R] = g_R$.

Upstream variance (Eq. 9). Write $(R - b)s_P = a + c$ with $a = (Q - b)s_P$ and $c = e\, s_P$. Then $\operatorname{tr}\operatorname{Cov}(a+c) = \operatorname{tr}\operatorname{Cov}(a) + \operatorname{tr}\operatorname{Cov}(c) + 2\big(\mathbb{E}[a^\top c] - \mathbb{E}[a]^\top \mathbb{E}[c]\big)$. Because s_P and Q are functions of (x, z), Eq. 20 gives $\mathbb{E}[c] = \mathbb{E}\big[s_P\, \mathbb{E}[e \mid x, z]\big] = 0$ and $\mathbb{E}[a^\top c] = \mathbb{E}\big[(Q - b)\|s_P\|^2\, \mathbb{E}[e \mid x, z]\big] = 0$, so the cross term vanishes. Finally $\operatorname{tr}\operatorname{Cov}(c) = \mathbb{E}[e^2 \|s_P\|^2] = \mathbb{E}\big[\sigma_W^2(x, z)\|s_P\|^2\big]$. No further assumption is needed.

Downstream variance (Eq. 10). Write $(R - b)s_R = a' + c'$ with $a' = e\, s_R$ and $c' = (Q - b)s_R$. We showed $\mathbb{E}[c'] = 0$, so $\operatorname{tr}\operatorname{Cov}(c') = \mathbb{E}\big[(Q - b)^2 \|s_R\|^2\big]$. The cross term is $2\,\mathbb{E}\big[(Q - b)\, e\, \|s_R\|^2\big]$. Here s_R depends on y, and so does e; under the stated assumption that e and $\|s_R\|^2$ are uncorrelated given (x, z), $\mathbb{E}[e\|s_R\|^2 \mid x, z] = \mathbb{E}[e \mid x, z]\, \mathbb{E}[\|s_R\|^2 \mid x, z] = 0$, and the cross term vanishes. □

A.3 Proof of Proposition 2

Fix x. For state z_i, write $Q_i = Q_\theta^P(x, z_i)$ and $R_{ij} = Q_i + e_{ij}$. Under the assumptions, the e_{ij} are conditionally independent given (x, z_i) with mean zero and variance σ_W^2, and $\widehat{Q}_i = Q_i + \bar{e}_i$ with $\bar{e}_i = \frac{1}{n}\sum_j e_{ij}$.

(i) Residual execution noise. Replacing R by $\widehat{Q}_i$ in the upstream estimator and repeating the argument of Appendix A.2 with $\bar{e}_i$ in place of e leaves an execution term $\mathbb{E}\big[\mathrm{Var}(\bar{e}_i \mid x, z_i)\|s_P\|^2\big]$. Conditional independence gives $\mathrm{Var}(\bar{e}_i \mid x, z_i) = \frac{1}{n^2}\sum_j \sigma_W^2 = \sigma_W^2/n$, hence the residual term $\mathbb{E}[\sigma_W^2\|s_P\|^2]/n$.

(ii) Reliability. Over the draw of z_i, $\mathrm{Var}(Q_i) = \sigma_B^2$. Since $\mathbb{E}[\bar{e}_i \mid z_i] = 0$, $\bar{e}_i$ is uncorrelated with Q_i, so $\mathrm{Cov}(\widehat{Q}_i, Q_i) = \sigma_B^2$ and $\mathrm{Var}(\widehat{Q}_i) = \sigma_B^2 + \sigma_W^2/n$. Therefore

$$\mathrm{corr}^2(\widehat{Q}_i, Q_i) = \frac{\sigma_B^4}{\sigma_B^2(\sigma_B^2 + \sigma_W^2/n)} = \frac{\sigma_B^2}{\sigma_B^2 + \sigma_W^2/n} = \frac{n\rho}{1 + (n-1)\rho}, \tag{22}$$

where the last step divides numerator and denominator by $(\sigma_B^2+\sigma_W^2)/n$. This is the Spearman–Brown formula for the reliability of a mean of n parallel measurements.

(iii) No state component; relation to leave-one-out. $R_{ij} - \widehat{Q}_i = (Q_i + e_{ij}) - (Q_i + \bar{e}_i) = e_{ij} - \bar{e}_i$, in which Q_i cancels. Let $\widehat{Q}_i^{(-j)} = \frac{1}{n-1}\sum_{\ell \neq j} R_{i\ell}$. Since $\widehat{Q}_i = \frac{1}{n}R_{ij} + \frac{n-1}{n}\widehat{Q}_i^{(-j)}$,

$$R_{ij} - \widehat{Q}_i = \frac{n-1}{n}\big(R_{ij} - \widehat{Q}_i^{(-j)}\big). \tag{23}$$

Given (x, z_i), $\widehat{Q}_i^{(-j)}$ is a function of sibling rewards only and is independent of y_{ij}, so $\mathbb{E}\big[\widehat{Q}_i^{(-j)}\, s_R(y_{ij}) \mid x, z_i\big] = \mathbb{E}[\widehat{Q}_i^{(-j)} \mid x, z_i]\ \mathbb{E}[s_R \mid x, z_i] = 0$. The self-included deviation is thus an unbiased estimator of g_R up to the positive constant $(n-1)/n$. The same argument applies to the state advantage with the factor $(m-1)/m$.

Effect of standardization. Eq. 13 divides the within-state deviations of state i by $\sigma_i^R + \epsilon > 0$, one factor shared by its n rollouts. Signs and order within the state are unchanged, so A_{ij}^R still contains no state component; because σ_i^R depends on the same rewards, the scaled estimator is unbiased only up to this per-state reweighting (Liu et al., 2025). A state whose n rollouts tie yields $A_{ij}^R = 0$ and contributes no reranking gradient, exactly as a constant-reward group in GRPO. □

A.4 Proof of Proposition 3

Fix (x, z) and write $Q = Q_\theta^P(x, z)$, $e = R - Q$, and $\nabla = \nabla^R$; all expectations are conditional on (x, z). By the score-function identity, $\mathbb{E}[e\, s_R] = \mathbb{E}[R\, s_R] - Q\,\mathbb{E}[s_R] = \nabla Q$. Since $\mathbb{E}[e] = 0$, the positive and negative parts of e have equal means, so $M = \mathbb{E}|e| = 2\,\mathbb{E}[e]_+$. Differentiating $F(\theta) = \mathbb{E}_{y\sim\pi_\theta^R}\big[(R - Q_\theta)_+\big]$, in which θ enters through both the sampling distribution and Q_θ, and using $\Pr(e = 0) = 0$,

$$\nabla F = \mathbb{E}\big[[e]_+\, s_R\big] - p_+\, \nabla Q. \tag{24}$$

With $F = M/2$ this gives $\mathbb{E}\big[[e]_+ s_R\big] = \frac{1}{2}\nabla M + p_+ \nabla Q$, and therefore

$$\mathbb{E}\big[(e + \eta[e]_+)\, s_R\big] = \nabla Q + \eta\big(\tfrac{1}{2}\nabla M + p_+\nabla Q\big) = (1 + \eta p_+)\nabla Q + \tfrac{\eta}{2}\nabla M. \tag{25}$$

Replacing e by $R - b$ with $b = b(x)$, as trajectory-level amplification does, the positive part becomes $[(Q - b) + e]_+$: its mean and the set of amplified rankings now depend on the state component $Q - b$, and the extra term no longer reduces to a function of the within-state reward distribution. □

A.5 Mass transfer under positive and negative updates

Lemma 1 (Mass transfer)**.** *Consider a softmax policy $\pi(y \mid z) \propto \exp f(y)$ over $\mathrm{Perm}(C)$ and a single-sample policy-gradient step with advantage A on ranking y, $\Delta f = \alpha A\, \nabla_f \log \pi(y \mid z)$. For any set S of rankings,*

$$\Delta\pi(S) = \alpha A\, \pi(y \mid z)\big(\mathbb{1}[y \in S] - \pi(S)\big) + O\Big(\alpha \textstyle\sum_{y'} \pi(y' \mid z)^2\Big) + O(\alpha^2). \tag{26}$$

Proof. For a softmax, $\nabla_f \log \pi(y) = \mathbf{e}_y - \pi$, so $\Delta f(y') = \alpha A(\mathbb{1}[y' = y] - \pi(y'))$. To first order, $\Delta\pi(y') = \pi(y')\big(\Delta f(y') - \sum_{y''} \pi(y'')\Delta f(y'')\big)$, and $\sum_{y''} \pi(y'')\Delta f(y'') = \alpha A\big(\pi(y) - \sum_{y''} \pi(y'')^2\big)$. Summing over $y' \in S$,

$$\Delta\pi(S) = \alpha A\Big[\pi(y)\mathbb{1}[y \in S] - \textstyle\sum_{y' \in S} \pi(y')^2 - \pi(S)\pi(y) + \pi(S) \sum_{y''} \pi(y'')^2\Big], \tag{27}$$

and the two squared-mass terms are bounded by $\sum_{y'} \pi(y')^2$. □

The permutation space of a 20-item candidate set is large and the reranker's mass is spread over many orderings, so the squared-mass terms are negligible. A positive update on y moves mass onto y itself, whereas a negative update on y spreads the removed mass over all other rankings in proportion to π, including other poor ones. We use the lemma as intuition for Conditional Amplification in Section 2.4.

B Breadth–Depth Allocation

NCCA draws m states and n rollouts per state, so an instance consumes $B = mn$ reranking evaluations. We ask how a fixed B should be split.

Upstream information. By Proposition 2(ii), each $\widehat{Q}_i$ recovers $Q^P_\theta(x, z_i)$ with reliability $n\rho/[1 + (n-1)\rho]$, and an instance contributes m states:

$$I_P(m, n) = m \cdot \frac{n\rho}{1 + (n-1)\rho} \overset{m=B/n}{=} \frac{B\rho}{1 + (n-1)\rho}, \tag{28}$$

which decreases in n. Taking $n = 1$ maximizes I_P, the classical result that one observation per cluster is optimal for estimating a mean of cluster means at a fixed total sample size.

Downstream information. The reranking advantage of Eq. 13 uses within-state deviations. Since $\sum_j (R_{ij} - \widehat{Q}_i) = 0$, each state supplies $n - 1$ independent comparisons:

$$I_R(m, n) = m(n-1) \overset{m=B/n}{=} B\Big(1 - \frac{1}{n}\Big), \tag{29}$$

which increases in n and vanishes at $n = 1$.

Optimal depth. The two terms move in opposite directions and share θ, so we score an allocation by the geometric mean of the normalized terms,

$$U(n) = \sqrt{\frac{I_P(n)}{B\rho} \cdot \frac{I_R(n)}{B}} = \sqrt{\frac{1}{1 + (n-1)\rho}\Big(1 - \frac{1}{n}\Big)}. \tag{30}$$

With $u = n - 1$, maximizing U^2 is maximizing $f(u) = u/[1 + u(1 + \rho) + u^2\rho]$, whose derivative has numerator $1 - u^2\rho$. The unique interior maximizer is $u = \rho^{-1/2}$:

$$n^* = 1 + \rho^{-1/2}, \tag{31}$$

independently of B. The optimal depth grows as ρ falls: when most reward variation comes from execution, more rollouts are needed before a state can be judged. The geometric mean is one reasonable aggregation among several; we use it because it is scale-free and has a closed-form optimum.

Comparison with the measured optimum. The measured $\rho \in [0.27, 0.38]$ gives $n^* \in [2.6, 2.9]$. Since n must divide B, the relevant question is how U ranks $n \in \{2, 4, 8\}$. Setting $U(2) = U(4)$ gives $\frac{1}{2}(1 + 3\rho) = \frac{3}{4}(1 + \rho)$, a crossover at $\rho = \frac{1}{3}$, inside the measured range; $n = 2$ and $n = 4$ differ by at most about 2.6% over $[0.27, 0.38]$, $n = 8$ loses about 14–19%, and $n = 1$ gives no reranking gradient. This ordering is in line with the training results of Section 3.6. Between $n = 2$ and $n = 4$, the latter avoids reducing each within-state comparison to a binary ± 1 advantage after standardization and bases each state estimate on more than two rollouts. Because $\lambda_P = 1/(1 + n)$ in Eq. 19, the total weight on the state stage also changes with n; the allocation results therefore compare complete training configurations rather than isolating the estimator.

Unequal stage costs. If generation cost is budgeted instead, with c_P per state and c_R per rollout, the budget is $C = m(c_P + nc_R)$, and minimizing the upstream estimator variance $(\sigma_B^2 + \sigma_W^2/n)/m$ alone gives the classical two-stage sampling optimum

$$n^*_{\text{cost}} = \sqrt{\frac{1-\rho}{\rho} \cdot \frac{c_P}{c_R}}. \tag{32}$$

The two expressions agree for c_P/c_R between roughly 3 and 5 at the measured ρ. This criterion ignores the downstream stage and serves only as a consistency check.

B.1 When should nested credit help?

The analysis ties the benefit of NCCA to one measurable quantity, the intra-state correlation ρ. Nesting helps most when many outputs are plausible under the same intermediate state, which is typical of ranking with few relevant items. It should help less when the intermediate state nearly determines the final output ($\rho \to 1$), and the optimal depth $n^* = 1 + \rho^{-1/2}$ shrinks accordingly. Measuring ρ on a frozen policy before training is cheap and gives a principled starting point for (m, n).

B.2 Allocations evaluated as estimators

The frozen pool of Section 3.4 lets us compare allocations as estimators, on identical rewards and without training. For each (m, n) with $mn = 16$ we draw m states and n rollouts and score the state credits against the held-out oracle (Table 3). Upstream credit quality improves monotonically with depth: $(2, 8)$ halves the credit MSE of $(4, 4)$. That this allocation is still the worst in training reflects two effects an upstream metric cannot see. The reranker's comparison count $m(n-1)$ rises only from 12 to 14 while the number of states halves, and at $m = 2$ the standardized state advantage is ± 1 by construction. The predicted $U(n)$, which accounts for both stages, tracks the training outcome instead. The $(16, 1)$ row is reported only to show how much upstream quality the boundary case would trade for a reranker without gradient.

Table 3: **Allocations evaluated as estimators on the frozen pool.** $U(n)$ is Eq. 30 at the pool's $\bar{\rho}$, normalized to its maximum. The instance-centered corr^2 is undefined for $m < 3$.

(m, n)	**Credit MSE** ↓	**Pair acc.** ↑	corr^2 ↑	**Within-state pairs**	$U(n)$ **predicted**
Traj-GRPO pool ($\bar{\rho} = 0.38$)					
$(16, 1)$	0.0155	75.1%	0.413	0	0.00
$(8, 2)$	0.0083	77.6%	0.549	8	**1.00**
$(4, 4)$	0.0047	80.7%	0.673	12	0.98
$(2, 8)$	**0.0028**	**84.0%**	—	14	0.81
NCCA pool ($\bar{\rho} = 0.30$)					
$(16, 1)$	0.0183	72.1%	0.339	0	0.00
$(8, 2)$	0.0097	74.6%	0.477	8	0.99
$(4, 4)$	0.0053	77.6%	0.624	12	**1.00**
$(2, 8)$	**0.0031**	**80.9%**	—	14	0.85

C Additional Experimental Details

C.1 Data construction

An instance is kept only if it has a valid history and request, a catalog-mapped target with explicit acceptance feedback, no target-title leakage in the model input, and a complete candidate set. The history contains earlier dialogue sessions and past interaction records of the user. Each retained instance has one accepted target and 19 unique in-domain negatives, drawn from rejected items, metadata-matched items, and history-sensitive negatives that are plausible given the current dialogue but not given the user's history; remaining slots are filled from a filtered, metadata-compatible pool.

C.2 SEQUENTIAL RECOMMENDATION BASELINES

We implement GRU4Rec, SASRec, and BERT4Rec with RecBole. These models do not read natural-language dialogue, so each record is converted into an item-interaction sequence. For TG-ReDial, each training dialogue is treated as a pseudo-user, and the chronologically ordered movies of its pre-conversation viewing history form the interaction sequence; the target recommendation is not appended, which prevents direct target leakage. For LLM-REDIAL, each original user forms one sequence from the ordered product IDs of the interaction history, and users that appear in the evaluation set are excluded from training. The baselines therefore use item IDs and interaction order only, without dialogue utterances, item text, or user states.

All models are trained with RecBole's next-item prediction objective and leave-one-out splitting (the last interaction for internal testing, the second-to-last for validation), full-softmax cross-entropy without negative sampling, a maximum sequence length of 50, learning rate 10^{-3}, at most 100 epochs, and early stopping with patience 10. Results are averaged over eight independently trained seeds, matching the eight test-time decoding runs used for the LLM-based methods. At evaluation time, a model receives the target user's observed item history and scores the full item vocabulary; we keep the scores of the same 20 candidates that the LLM-based methods rerank and rank them accordingly. A candidate unseen in training receives the median score of the in-vocabulary candidates of the same instance, so out-of-vocabulary items are neither favored nor penalized.

C.3 HYPERPARAMETERS AND PROMPTS

Table 4: **Training configuration.**

Backbone	Qwen3-4B-Thinking-2507, both stages share parameters
Hardware / global batch	8 GPUs / 128 instances; 1 epoch
Reranking evaluations per instance	16 (Traj-GRPO: $(16, 1)$; NCCA: $(4, 4)$)
Amplification η	0.3
Training sampling (both stages)	temperature 0.7, top-p 0.95
Max prompt / response length	8,192 / 8,192 tokens
Terminal reward $r(y, G)$	NDCG@10 of the reranked list
Optimizer	AdamW, learning rate 1×10^{-6}
Training seeds	2 per trained LLM
Test-time decoding	temperature 0.3, 8 stochastic decodes per instance, averaged

Prompts. Both stages are generated by the same model with the templates below. Placeholders are shown in {braces}; the dialogue context is the current session up to the request. The templates are shown for movies (TG-ReDial); for LLM-REDIAL the item type follows the domain. The Stage 1 output between <profile> tags is the user state z, and it replaces {user state} in the Stage 2 template.

Stage 1: user-state generation

System. You are the user-understanding module of a recommender system. Your task is not to recommend movies, but to distill the user's historical information into a preference profile for the downstream ranking module.

User.
[Dialogue Context] {turns of the current session}
[User Request] {current user request}
[User Self-Description] {self-description sentence}
[Watching History] (in chronological order, earliest to latest) {watched titles}
[Recommendation Records in Historical Sessions]
– Recommended "{title}" to the user; the user accepted. Feedback at the time: "{feedback}"
– Recommended "{title}" to the user; the user was not interested. Feedback at the time: "{feedback}"
{...}

Please read and deeply analyze the historical information above, and fully uncover this user's interests, taste characteristics, and emotional needs. Do not stop at surface-level summarization; instead, grounded in the specific viewing records, user feedback, and dialogue content, extract preference patterns and aesthetic tendencies in depth, paying particular attention to the details most relevant to the current request.

Write your analysis between <profile> and </profile>. The profile should be grounded in actual evidence from the historical information, should be analytically deep, and should capture as much as possible what the user truly cares about.

The downstream ranking module can only see the user's request and the profile you write; it cannot see any of the original historical information. Therefore, the profile must state clearly and completely the preferences relevant to the current request, so that the downstream module can make accurate recommendation judgments on that basis.

Do not give any specific movie recommendations or conclusions in the profile; describe only the preferences themselves. Output nothing other than <profile></profile> and its content.

Stage 2: candidate ranking

[Dialogue Context] {turns of the current session}
[User Request] {current user request}
[User Preference Profile] (generated by the upstream module from this user's long-term history)
{user state}
[Candidate Movies] ({N} in total, randomly ordered)
{candidate titles, one per line}

Based on the user's request and the preference profile above, rank all candidate movies by recommendation priority for this user. For preferences and aversions explicitly stated in the profile, apply them when ranking; for aspects the profile does not cover, refer back to the user's request itself.

You must write the final ranking between <answer> and </answer> (the tags must not be omitted), with each candidate on its own line, in the format `Title####Ranking`. The title must match the wording in the candidate list character by character. `Ranking` is an integer from 1 to {N}, where 1 is the highest recommendation priority and {N} is the lowest. Every integer from 1 to {N} must be used exactly once, so all candidates receive unique ranks. The movies in the answer must be exactly consistent with the candidate list in both number and content: every movie in the candidate list must appear, no movie outside the candidate list may appear, and no movie may be merged or omitted. Between <answer> and </answer> only the ranking lines are allowed; no other content may appear.

The permutation y is obtained by ordering candidates by ascending assigned rank.

C.4 Sensitivity and allocation results

Table 5: **Sensitivity and breadth–depth allocation.** (a) varies η at $(m, n) = (4, 4)$; (b) varies (m, n) with $mn = 16$ and $\eta = 0.3$. Blue rows are the defaults.

	TG-ReDial		LLM-REDIAL	
Setting	**R@10**	**N@10**	**R@10**	**N@10**
(a) Amplification weight η				
$\eta = 0$	0.7753	0.4302	0.7480	0.3997
$\eta = 0.1$	0.7903	0.4469	0.7611	0.4034
$\eta = 0.3$	0.8016	**0.4543**	**0.8274**	**0.4585**
$\eta = 0.5$	**0.8101**	0.4278	0.7914	0.4256
(b) Allocation (m, n) with $mn = 16$				
$(8, 2)$	**0.8073**	0.4481	0.7906	0.4333
$(4, 4)$	0.8016	**0.4543**	**0.8274**	**0.4585**
$(2, 8)$	0.7549	0.4167	0.7683	0.4093

C.5 State-credit quality on frozen policies

Table 6 reports the credit errors summarized in Section 3.4. The ordering is the same on both policies, so it reflects the estimator rather than a particular training run.

C.6 What trajectory-level amplification amplifies

On the frozen pools of Section 3.4, including one built from Base, we standardize one rollout per state across the 16 states of an instance, as trajectory-level credit does, and split each flat advantage into a state part $(Q^{\text{oracle}}(z_s) - \mu)/\sigma$ and an execution part $(R_s - Q^{\text{oracle}}(z_s))/\sigma$, where μ and σ

Table 6: **State-credit quality on frozen policies.** $n{=}1$ is the trajectory-level credit of Traj-GRPO; $n{>}1$ is Nested Credit on the same reward pool. MSE: credit MSE; A-MSE: advantage MSE; Sign: sign accuracy (%); Good$^-$: fraction (%) of states above the oracle mean that receive a negative advantage; Bad$^+$: the converse; Cos: cosine between estimated and oracle state advantages.

	Traj-GRPO policy						NCCA policy					
n	MSE↓	A-MSE↓	Sign↑	Good$^-$↓	Bad$^+$↓	Cos↑	MSE↓	A-MSE↓	Sign↑	Good$^-$↓	Bad$^+$↓	Cos↑
1	0.0134	0.822	73.0	31.0	25.8	0.589	0.0146	0.938	70.3	37.3	25.3	0.511
4	0.0040	0.423	80.6	21.3	18.5	0.788	0.0043	0.559	78.0	26.0	19.2	0.713
8	**0.0024**	**0.294**	**83.9**	**17.9**	**14.9**	**0.853**	**0.0026**	**0.419**	**81.1**	**21.5**	**16.9**	**0.790**

are the mean and standard deviation of the 16 rewards. The state share of a set of trajectories is $\sum |\text{state part}| / \sum \big(|\text{state part}| + |\text{execution part}|\big)$, with both sums taken over that same set. Table 7 reports it separately for positive and negative advantages, together with the fraction of positive advantages whose state lies below the instance mean.

Table 7: **Composition of the flat trajectory advantage on frozen pools.** Under Nested Credit the state share of A^R_{ij} is zero by construction.

Pool	**State share of** $A > 0$	**State share of** $A < 0$	$A > 0$ **from a below-mean state**
Base	48.1%	50.4%	30.1%
Traj-GRPO	49.2%	48.8%	27.7%
NCCA	42.1%	44.8%	30.7%

Close to half of the positive advantage mass, the part that amplification enlarges, comes from the state, and more than a quarter of positive advantages start from a below-mean state. The shares are in line with the measured ρ, slightly above the Gaussian reference: if the two parts were Gaussian with variances in the ratio $\rho : (1-\rho)$, the expected state share would be $\sqrt{\rho}/(\sqrt{\rho} + \sqrt{1-\rho})$, i.e., 0.44 at $\bar{\rho} = 0.38$ and 0.40 at $\bar{\rho} = 0.30$, against the measured 0.49 and 0.42 for positive advantages. The lower share on the NCCA pool matches its lower ρ.

C.7 Full results of the history intervention

Table 8: **History intervention on TG-ReDial, all arms.** Same decoding pass as Figure 2. Relative changes from Full of the same model in parentheses.

	NCCA		Base	
Arm	**N@10**	**R@10**	**N@10**	**R@10**
Full	**0.4499**	**0.8169**	**0.3818**	0.7384
No-history	0.4211 (−6.4%)	0.8079 (−1.1%)	0.3625 (−5.1%)	**0.7554** (+2.3%)
No-state	0.3919 (−12.9%)	0.7791 (−4.6%)	0.3587 (−6.1%)	0.7472 (+1.2%)
Mismatched	0.3878 (−13.8%)	0.7105 (−13.0%)	0.3194 (−16.3%)	0.6468 (−12.4%)

C.8 TRAINING DYNAMICS

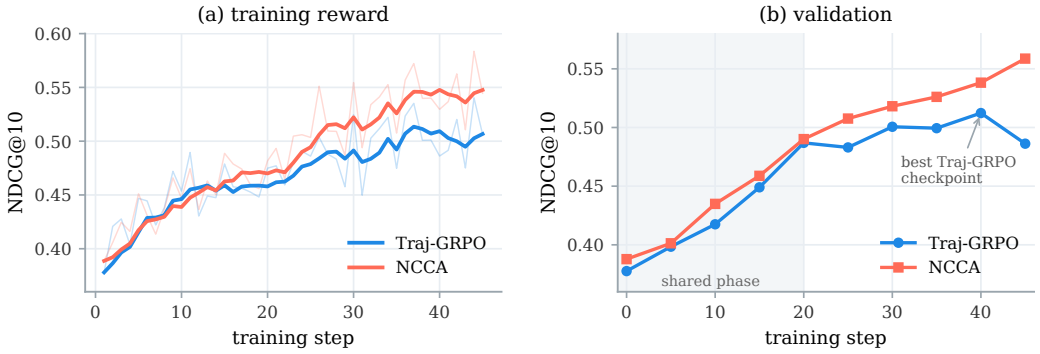


Figure 5: **Training dynamics.** Both runs start from Base with the same data, reward, and schedule; only the credit assignment differs. (a) Training reward (thin: per step; thick: smoothed). (b) Validation NDCG@10. The two runs are indistinguishable for about 20 steps; afterwards NCCA keeps improving while Traj-GRPO stops improving and ends below its own best checkpoint.

Early progress comes from signal the two objectives share. The difference appears once further gains require telling good states apart from the rankings they happen to produce, which is where trajectory-level credit is noisiest (Section 3.4). Because Traj-GRPO's best validation checkpoint is still below NCCA's, the gap in Table 1 is not a checkpoint-selection artifact.