COMPASS: COMPLETION-BASED PROCESS ADVANTAGE SUPERVISION FOR SEQUENTIAL RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Delayed rewards make credit assignment difficult in sequential decision making. A denser signal can be obtained by trying different actions from the same prefix, completing each branch, and comparing outcomes. These returns belong to the completion policy, however, and their absolute values can become stale as the learner improves. Comparisons between actions can be more stable than the corresponding absolute returns. COMPASS exploits this distinction by learning action advantages from offline completions and using their centered predictions to guide value learning. Centering removes the shared return level while the usual TD objective continues to learn from environmental rewards. The completion model is trained once, frozen, and reused throughout RL. On RL4RS, COMPASS improves DQN's mean held-out return by 20.7% and 43.1% on two sequential recommendation tasks. The same frozen model also improves IQN on all eight evaluated seeds. We further extend the idea to slate recommendation, improving DDPG by 12.5% on KuaiSim. Ablations show that centering, continued supervision during RL, and when to turn that supervision off all matter. Overall, our results suggest a simple principle: old completion returns may stop being good value targets long before they stop being useful comparisons.

1 INTRODUCTION

A recommender often makes several choices before learning whether a page or session succeeded, making early decisions hard to evaluate from logged rewards. In a simulator, we can instead stop at the same prefix, replace an action, and roll the session forward. Repeating this gives evidence about alternatives the online agent may never try at exactly the same state.

Once these rollouts have been collected, we can fit a model that predicts alternative outcomes and query it cheaply in later training runs. Completion data thus turn expensive rollouts into reusable action-level feedback.

There is an immediate problem, though. A completion return asks what happens if the fixed policy μ takes over after an action. The learner's value function evaluates its own continuation, which changes throughout training. More completions cannot remove this mismatch: even a perfectly estimated return under μ is still a return under μ.

Suppose two actions have completion returns 1 and 0. A better continuation may later raise them to $M + 1$ and M. The old returns are now poor estimates of either value, but their difference is still exactly right. If the extra gain is available only after one of the actions, however, the ordering can reverse. What matters is therefore not simply whether the completion policy is weak, but how policy improvement changes the alternatives relative to one another.

COMPASS learns these contrasts between actions. Offline, we collect intervention and reference completions and fit a completion-advantage model. During RL, the model is frozen and queried on replay states. We center its scores and the agent's Q-values over legal actions before matching them. Figure 1 summarizes the resulting offline-to-online pipeline.

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.

Our analysis characterizes how changing the continuation policy alters an action comparison and how exact or imperfect centered guidance affects value learning. This makes the reuse question precise without fixing the learner to the completion policy's value scale.

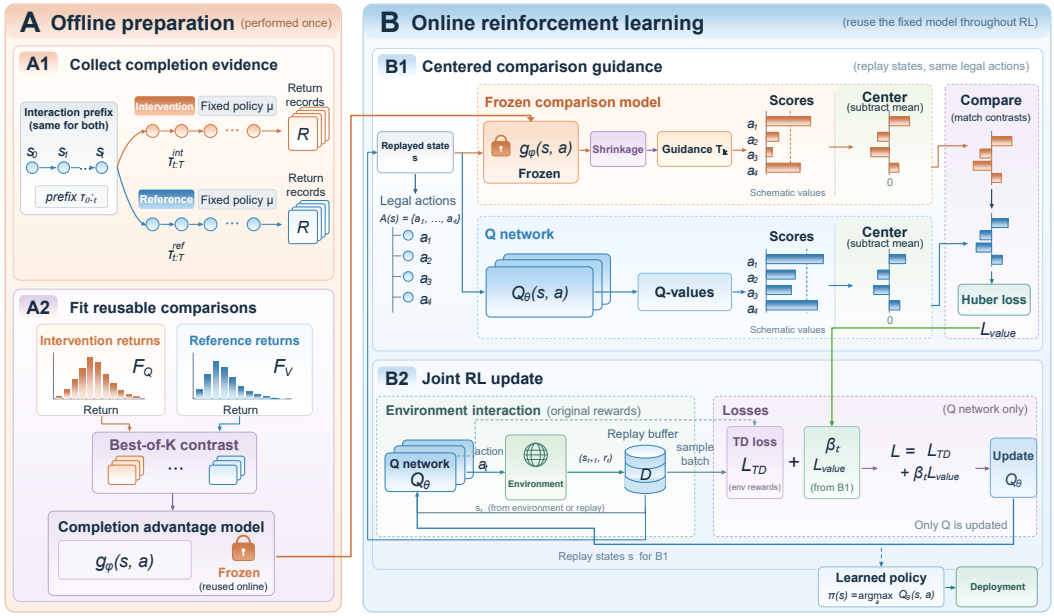


Figure 1: COMPASS value interface, shown with optional Best-of-K and shrinkage. Offline completions provide targets for a completion-advantage model (A1–A2). Online, its frozen scores and Q-values are centered over legal actions (B1); auxiliary and TD losses jointly update the Q-network (B2). The completion model is reused across runs. Bars and distributions are schematic.

2 Related Work

Credit assignment and process supervision. RUDDER addresses delayed rewards by redistributing return so that future rewards become easier to predict (Arjona-Medina et al., 2019), while hindsight and counterfactual methods use future information to obtain more informative credit estimates or lower-variance policy gradients (Harutyunyan et al., 2019; Mesnard et al., 2021; Meulemans et al., 2023). Process Advantage Verifier supervision is the closest conceptual precursor (Setlur et al., 2025), which also evaluates intermediate decisions using completions from a separate policy and shows that even a weaker prover can be useful. COMPASS instead uses completion-derived contrasts to supervise relative action values. We analyze how action gaps change across continuation policies.

Action gaps and value guidance. Prior work studies action gaps and optimality-preserving gap enlargement (Farahmand, 2011; Bellemare et al., 2016). Prediction-assisted Q-learning uses predictions of optimal Q-values (Golowich & Moitra, 2022). Our scores come from a fixed completion policy that need not remain competitive with the learner.

Demonstrations and lookahead. DQfD adds demonstration-action supervision to TD learning (Hester et al., 2018), while Distral transfers behavior through a distilled policy shared across tasks (Teh et al., 2017). MCTS and MuZero evaluate alternatives through online lookahead (Browne et al., 2012; Schrittwieser et al., 2020). COMPASS completes branches offline and reuses their comparisons through a fixed predictor without online counterfactual rollout.

3 Problem Setup

We consider a finite-horizon Markov decision process with horizon H, environmental reward r_t, discount $0 < \gamma \leq 1$, and legal action set $\mathcal{A}(s)$.

The state contains the interaction history and time index. For a policy π, $J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{H-1} \gamma^t r_t\right]$, and $Q^\pi(s,a)$ denotes the expected remaining return after taking action a at state s and following π thereafter.

We reserve μ for the fixed completion policy used to construct offline evidence, and π for a policy learned online. Since the state includes the interaction history and time, resuming from a shared interaction prefix is equivalent to resuming from the same state s.

For each such state, we consider two completion branches under μ. An intervention branch first takes a selected legal action $a \in \mathcal{A}(s)$ and then follows μ, while a reference branch follows μ directly from s. Let $F_Q^\mu(\cdot \mid s,a)$ and $F_V^\mu(\cdot \mid s)$ denote the corresponding remaining-return distributions. Let $Q^\mu(s,a)$ denote the mean return of the intervention branch and $V^\mu(s)$ the mean return of the reference branch. We define the completion advantage as $A^\mu(s,a) = Q^\mu(s,a) - V^\mu(s)$. Because the reference term is shared across actions at the same state, $A^\mu(s,a) - A^\mu(s,b) = Q^\mu(s,a) - Q^\mu(s,b)$ for any $a, b \in \mathcal{A}(s)$. Thus, subtracting the shared reference preserves the pairwise action differences under μ.

4 METHOD

Figure 1 separates offline construction from online learning.

4.1 CONSTRUCTING COMPLETION SUPERVISION

Estimating completion returns. We fit the intervention and reference distributions from Section 3 with conditional quantile regression on RL4RS, and empirical repeats on KuaiSim (Appendices B.2 and C.3).

Once the distributions have been estimated, different supervision statistics can be computed without collecting another trajectory.

Choosing a completion statistic. Alongside the ordinary mean, we use Best-of-K, $B_K(F) = \mathbb{E}\left[\max_{1\le j\le K} G_j\right]$, $G_1, \ldots, G_K \overset{\text{i.i.d.}}{\sim} F$. At $K = 1$, B_K is simply the mean. Larger K places more weight on successful continuations that occur less frequently under μ. Applying the same statistic to the intervention and reference distributions gives $A_K(s,a) = B_K\left(F_Q^\mu(\cdot \mid s,a)\right) - B_K(F_V^\mu(\cdot \mid s))$. We compute B_K by reweighting fitted quantiles or empirical samples; Appendix A.5 gives the corresponding formulas.

Fitting the score model. The distribution models construct targets. A separate state–action model g_ϕ predicts $A_K(s,a)$ at new replay states encountered during RL.

Optional dispersion-aware shrinkage. The predicted advantage can be used directly. On RL4RS, we optionally suppress contrasts small relative to return variability. The score model also predicts $\sigma_K^A = \sqrt{(\sigma_K^Q)^2 + (\sigma_K^V)^2}$, and we form

$$T_K(s,a) = \mathrm{sg}\left[\mathrm{sign}(\widehat{A}_K(s,a))\left(|\widehat{A}_K(s,a)| - \kappa\widehat{\sigma}_K(s,a)\right)_+\right].$$

The dispersion describes variability in the completion outcomes, not uncertainty of g_ϕ. Setting $\kappa = 0$ removes this optional shrinkage and recovers the predicted completion advantage.

After fitting, g_ϕ is frozen.

4.2 CENTERED SUPERVISION FOR VALUE LEARNING

For any score vector $x(s,\cdot)$, define $C_s x(a) = x(s,a) - \frac{1}{|\mathcal{A}(s)|}\sum_{b\in\mathcal{A}(s)} x(s,b)$. Because $C_s x(a) - C_s x(b) = x(s,a) - x(s,b)$, centering preserves every action gap while removing a statewise common offset.

For replay states $\mathcal{B}$, the auxiliary objective is

$$L_{\text{value}}(\theta) = \mathbb{E}_{s\sim\mathcal{B}}\left[\frac{1}{|\mathcal{A}(s)|}\sum_{a\in\mathcal{A}(s)} h_\delta\left(C_s Q_\theta(a) - C_s T_K(a)\right)\right], \tag{1}$$

where h_δ is the Huber loss and sg stops gradients. The objective is invariant to state-dependent shifts in either score vector.

The value network is trained with $L_t = L_{\text{TD}} + \beta_t L_{\text{value}},\ \beta_t \geq 0$. T_K is detached and never enters the TD target. Both centering and loss averaging use legal actions before numerical masking. The schedule for β_t is specified with each experiment.

4.3 Value and preference interfaces

We use action-level scores for discrete value learners and separately fitted slate-level scores for structured slate policies. **Value interface.** DQN uses the centered objective directly. For IQN, we average its quantile predictions to obtain mean action values and apply the same loss, while leaving its distributional TD objective unchanged.

Preference interface. For slates, a separately fitted model supplies $T_K^{\text{slate}}(s, L)$ for each complete list L. For L_i, L_j, $\Delta T_{ij} = T_K^{\text{slate}}(s, L_i) - T_K^{\text{slate}}(s, L_j)$. The sign of ΔT_{ij} determines the preferred slate, while its magnitude weights the pair. We train the actor to assign higher differentiable slate likelihood to the preferred list. Unlike the value interface, this objective transfers ordering rather than a numerical action-value gap. Appendix C.1 gives the full pairwise softplus loss and six-item decoder.

5 Theoretical Analysis

We study value-update stability, comparison transfer across policies, and the resulting policy-return error. The value-update analysis applies to the numerical value interface of Section 4.2; the slate preference interface has a different gradient. We use independently represented state–action values, population expectations, and detached TD targets in a finite-state, finite-action, finite-horizon MDP with bounded rewards and zero terminal value. Appendix A contains proofs and discrete-update variants.

5.1 What centering changes in the value update

Write $(BQ)_h(s,a) = r_h(s,a) + \gamma P_{h,a} V_{Q,h+1}$, where $P_{h,a} f = \mathbb{E}[f(s_{h+1}) \mid s, a]$ and $V_{Q,h} = \max_a Q_h$. Let C denote legal-action centering and $\psi_\delta = h'_\delta$. After normalizing time to the TD update rate,

$$\dot{Q} = BQ - Q - \beta(t) C \psi_\delta(C(Q - T_K)). \tag{2}$$

Theorem 1 (Centered guidance and value convergence). For bounded nonnegative $\beta(t)$, the auxiliary term has zero direct contribution to the statewise mean. If $CT_K = CQ^*$ at every state, then Q^* is globally asymptotically stable. For any bounded fixed guidance scores, $\beta(t) \to 0$ also implies $Q(t) \to Q^*$.

Exact centered guidance can remain active; imperfect guidance is asymptotically harmless when its weight vanishes.

The effect on an individual action gap is more explicit. In the quadratic region of the Huber loss, $\dot{e}_{ab} = \gamma(P_{h,a} - P_{h,b})(V_{Q,h+1} - V^*_{h+1}) - (1+\beta)e_{ab} + \beta\zeta_{ab}$, where $e_{ab} = (Q_a - Q_b) - (Q^*_a - Q^*_b),\ \zeta_{ab} = (T_a - T_b) - (Q^*_a - Q^*_b)$. For constant β in this quadratic region, once the continuation-value error vanishes, exact guidance changes the decay rate of the gap error from e^{-t} to $e^{-(1+\beta)t}$. Inexact guidance introduces the additional term $\beta\zeta_{ab}$. Thus β controls an acceleration–bias trade-off: stronger guidance speeds gap correction but also amplifies errors in the supplied contrast. A vanishing schedule keeps the transient correction while recovering Q^* asymptotically.

5.2 When completion comparisons transfer across policies

We next isolate the term by which changing the continuation policy can alter an action gap. For any agent policy π and completion policy μ,

$$\begin{aligned}(Q_h^\pi(s,a) - Q_h^\pi(s,b)) - (Q_h^\mu(s,a) - Q_h^\mu(s,b)) \\ = \gamma(P_{h,a} - P_{h,b})(V_{h+1}^\pi - V_{h+1}^\mu).\end{aligned} \tag{3}$$

The right-hand side is the difference in expected continuation gains after the two actions. In particular, the overall quality gap between μ and π is not enough to determine whether a comparison transfers; only the action-dependent part of that improvement changes the gap.

For reuse, suppose a guidance gap ΔT approximates the completion-policy gap ΔQ^μ with error at most e, and define $D_\Pi = \sup_{\pi\in\Pi} \left|\gamma(P_{h,a} - P_{h,b})(V_{h+1}^\pi - V_{h+1}^\mu)\right|$. If $\Delta T > e + D_\Pi$, then $Q_h^\pi(s,a) > Q_h^\pi(s,b)$ for every $\pi \in \Pi$. Thus a large enough guidance margin can certify an ordering over an entire family of later policies. Appendix A.2 decomposes the implemented score error further into model error, Best-of-K emphasis, shrinkage, and continuation-policy change.

5.3 From comparison error to policy return

Three sources of decision loss are missing high-value actions, distorted contrasts, and failure to select the action favored by guidance.

Write $\mathrm{sp}(x) = \max x - \min x$. For a comparison set $\mathcal{C}_h(s)$ containing the selected action a^π, define the coverage loss $c_h(s) = V_h^*(s) - \max_{a\in\mathcal{C}_h(s)} Q_h^*(s,a)$, the contrast error $\epsilon_h(s) = \mathrm{sp}_{\mathcal{C}_h(s)}(T_K - Q_h^*)$, and the guidance shortfall $r_{T,h}(s) = \max_{a\in\mathcal{C}_h(s)} T_K(s,a) - T_K(s,a^\pi)$.

Theorem 2 (Return bound for guided decisions). For any deterministic policy π with state distributions d_h^π,

$$J^* - J(\pi) \le \sum_{h=0}^{H-1} \gamma^h \mathbb{E}_{s\sim d_h^\pi}[c_h(s) + \epsilon_h(s) + r_{T,h}(s)]. \tag{4}$$

For DQN, all legal actions are compared and therefore $c_h = 0$. The auxiliary loss also controls the last term: if the selected action maximizes the learned Q-values over $m \ge 2$ compared actions, then $r_T \le 2h_\delta^{-1}(mL_{\text{value}}/2)$, and in the quadratic region, $r_T \le 2\sqrt{mL_{\text{value}}}$. Hence a small centered loss directly limits how far the learner's decision can fall below the guidance scores. Appendix A.3 proves the bound and Appendix A.4 gives the corresponding calibration for the native slate decoder.

6 Experiments

We ask whether COMPASS improves its value learner, how completion evidence should enter RL, which design choices carry the gain, and whether reuse survives changes in learner and action space.

6.1 Experimental setup

RL4RS. RL4RS uses a learned DIEN user model (Wang et al., 2023; Zhou et al., 2019). Task A has nine choices with feedback after the slate; B has four pages and 36 decisions, with feedback after each page. Both have 284 items, legal-action constraints, and disjoint construction, online-training, and evaluation sessions.

Final greedy policies are evaluated on 8,192 held-out sessions using the original environmental return. We trim the highest 1% of session returns before averaging. Main comparisons use five seeds; matched design controls use three. Unless stated otherwise, COMPASS is built on Double DQN. All methods receive the same online interaction budget and are evaluated with the same protocol.

Each task uses 135,000 prefixes from 50,000 construction sessions to fit its completion model before RL. The main RL4RS configuration uses $K = 64$, $\kappa = 1$, and 500 training epochs, corresponding to approximately 504,000 environment interactions. The guidance weight is initially held at 1, then decays to zero halfway through training. Appendices B.2–B.5 contain the full construction, fitting, learner, and evaluation settings; Appendix D.1 reports computational costs.

KuaiSim. KuaiSim (Zhao et al., 2023) tests preferences on six-item slates with a separately fitted slate model. Each method trains for 2,000 updates. We average each run's final 101 training reports, then report mean and sample SD over five seeds. The main COMPASS configuration uses $K = 1$, $\kappa = 0$, and constant $\beta = 1$. Appendix C gives the full slate construction and learner settings.

6.2 Main Results on RL4RS

The main comparison adds completion supervision to DQN under the same online interaction budget.

Table 1: Final RL4RS held-out return: mean ± sample SD over five seeds, using the common 8,192-session top-1%-trimmed metric. Bold marks the highest mean.

Method	RL4RS-A	RL4RS-B
COMPASS	$\mathbf{152.77 \pm 3.50}$	378.83 ± 37.76
DQN	126.52 ± 3.87	264.70 ± 26.64
PPO	139.90 ± 7.01	$\mathbf{388.03 \pm 50.36}$
A2C	112.32 ± 2.95	252.62 ± 61.94
Rainbow	131.80 ± 3.13	292.23 ± 54.68
IQN	130.33 ± 3.27	266.06 ± 33.46
Graph-DQN	133.91 ± 2.58	154.15 ± 34.20
RRD	139.24 ± 8.97	224.71 ± 37.44
ARES	135.89 ± 13.46	220.35 ± 31.19
COCOA	106.72 ± 3.00	224.49 ± 17.74
VFD	125.54 ± 5.28	215.94 ± 39.82
DiAster	121.23 ± 5.15	186.90 ± 39.08

COMPASS improves DQN by 20.7% on Task A and 43.1% on Task B. On A, this gives COMPASS the highest mean among all evaluated methods. The gain is even larger on the substantially longer Task B, where completion supervision moves DQN from 264.70 to 378.83, bringing the value-based learner close to PPO at 388.03. COMPASS also has a higher median (394.98 versus 381.29) and lower across-seed standard deviation (37.76 versus 50.36) than PPO.

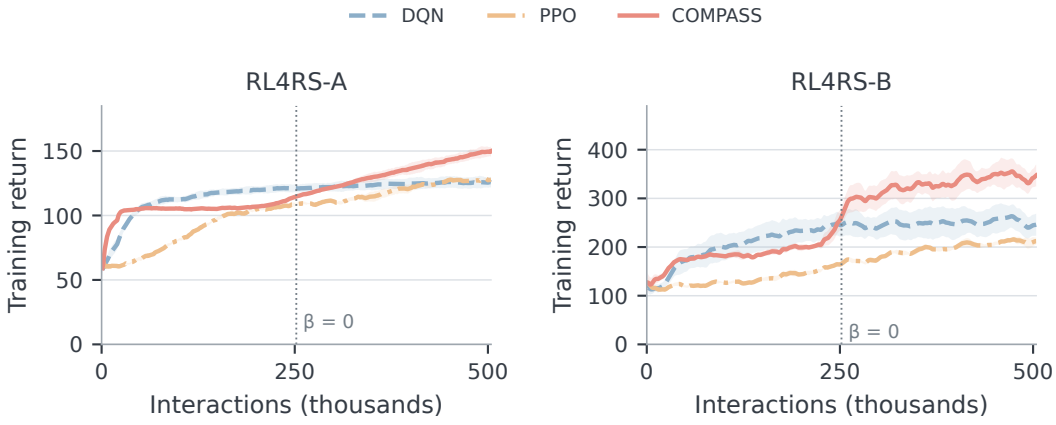


Figure 2: RL4RS training return: five-seed mean and sample SD. The dashed marker shows guidance withdrawal. Full trajectories and smoothing details are in Appendix D.2.

Figure 2 shows the change in learning trajectory; ordinary RL continues after guidance reaches zero.

6.3 How Completion Supervision Works

We keep the evidence fixed and change how the learner consumes it.

Interface. Reward injection adds predicted completion advantages to environmental rewards for TD learning. COMPASS uses an auxiliary comparison loss and leaves the reward process untouched.

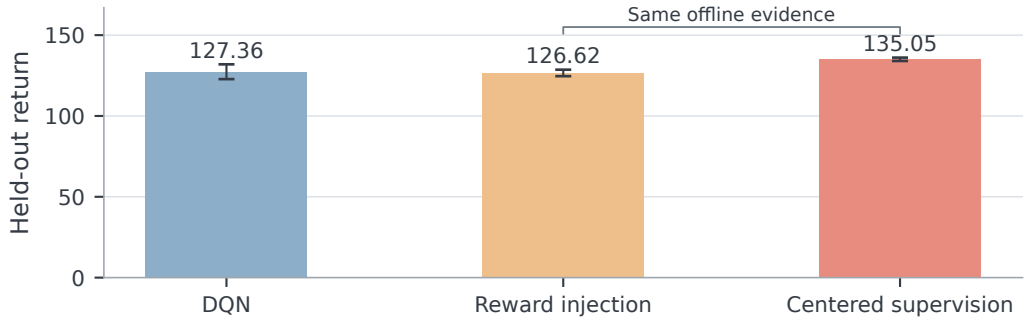


Figure 3: Same evidence, different interfaces: held-out mean and sample SD, seeds 0–2. Reward injection uses coefficient 1; centered supervision uses the step schedule in Appendix B.7.

Matched over seeds 0–2, vanilla DQN reaches 127.36. Reward injection reaches 126.62, producing essentially no improvement. Centered auxiliary supervision reaches 135.05.

The same completion evidence yields no gain through reward injection but helps through centered auxiliary supervision.

Timing. Pretraining is not enough. Comparison pretraining applies the centered objective before online interaction; MC-return pretraining fits raw completion returns. Both then receive the full RL budget.

Comparison pretraining reaches 132.32, and MC-return pretraining reaches 135.88. COMPASS reaches 153.53 on the same three seeds.

Continued queries on replay states are considerably more effective than consuming the evidence only before RL. Appendix B.6 gives the fitting procedures and state-sampling distinction.

Figure 4 groups the controls by the question each addresses.

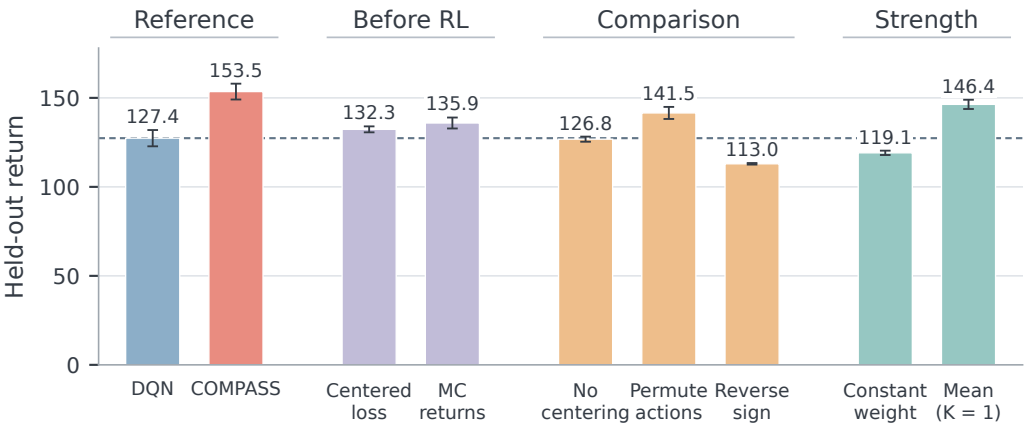


Figure 4: RL4RS-A design controls: mean held-out return and sample SD, seeds 0–2. Groups separate references, pretraining, comparison controls, and guidance strength. Dashed line: DQN.

Centering and correspondence. Centering is indispensable. Removing centering drops return from 153.53 to 126.80, almost exactly the DQN baseline of 127.36.

This wipes out essentially the entire gain: centering is the clearest tested distinction between successful supervision and forcing the learner toward the offline score scale.

Preserving action correspondence yields the strongest performance: COMPASS achieves 153.53, compared with 141.54 under permutation and 127.36 for DQN. Permutation retains a substantial improvement over DQN, but weakens the benefit of completion supervision. Reversing the guidance signs further reduces return to 112.96, below the DQN baseline. Together, these controls support preserving action correspondence and guidance direction in the completion-supervision interface.

Guidance duration and completion statistic. Persistent guidance can be actively harmful. The main RL4RS schedule removes guidance halfway through training. If we instead keep $\beta = 1$ throughout, return falls from 153.53 to 119.12—below even vanilla DQN.

Persistent guidance is harmful here. This is consistent with Section 5.1: guidance can accelerate early gap correction while persistent supervision keeps enforcing its errors. The schedule is a substantive choice in this RL4RS setting.

Best-of-K strengthens COMPASS, but does not carry it. With ordinary mean completion comparisons, COMPASS still reaches 146.35, far above DQN at 127.36. Increasing to the main $K = 64$ configuration raises performance further to 153.53.

Best-of-K improves the signal on RL4RS, but COMPASS already works without it. Appendix B.7 reports the full K-sweep, including its non-monotone behavior.

6.4 Reuse Across Learners and Action Spaces

Backbone reuse. We test backbone reuse by applying the same frozen completion model to IQN.

Across eight paired seeds, COMPASS improves DQN from 128.16 ± 4.13 to 150.32 ± 6.87, and IQN from 135.85 ± 12.70 to 155.25 ± 8.26. COMPASS improves all eight paired seeds for both DQN and IQN. No new completion collection or refitting is required. IQN exposes mean action values to the centered interface and retains its native quantile TD objective.

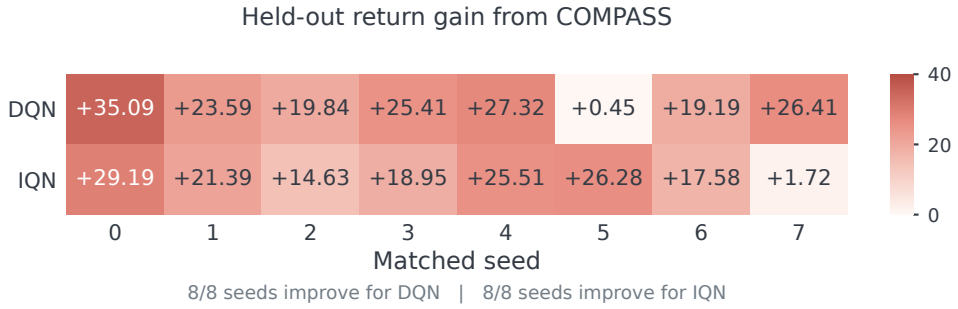


Figure 5: Matched-seed held-out gains (COMPASS minus baseline). All 16 are positive, using the same frozen model. Full returns: Appendix B.8.

Slate preference interface. Finally, we change more than the backbone. KuaiSim uses a different environment, six-item structured actions, and the preference interface from Section 4.3. Its completion scores are fitted directly on entire proposed slates rather than individual actions.

COMPASS improves DDPG from 8.661 ± 0.150 to 9.746 ± 0.236, a 12.5% gain. It also exceeds HAC, the strongest non-COMPASS baseline by five-seed mean, with a 5.6% gain. Every matched COMPASS seed exceeds both DDPG and HAC.

The separately fitted slate-level model supplies ordering rather than numerical value gaps, extending completion supervision to a second learning interface.

The preferred schedule flips on KuaiSim. The slate experiments favor a different schedule from RL4RS: constant guidance works better. Constant supervision exceeds the decaying alternative on all five seeds, by 0.482 return on average. A separate Best-of-K study also favors the simpler statistic here: $K = 1$ gives 9.806 ± 0.170, while $K = 64$ gives 9.703 ± 0.075.

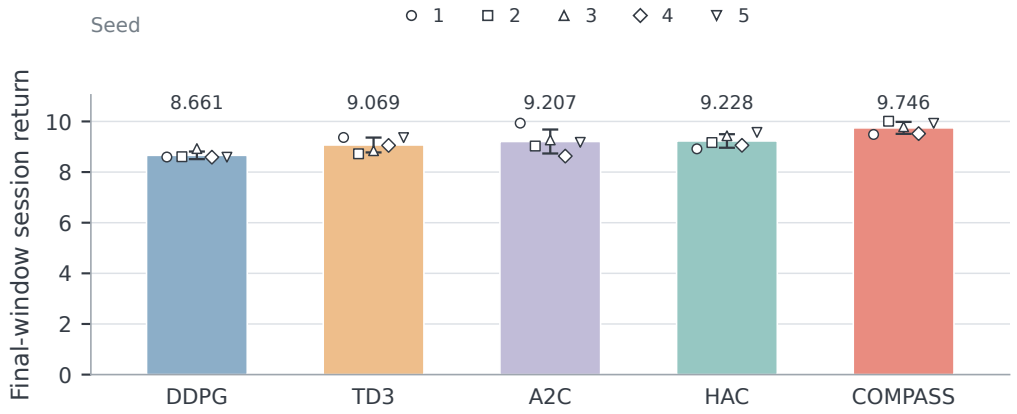


Figure 6: KuaiSim final-window return. Bars and whiskers show five-seed mean and sample SD; symbols show individual seeds. Each run averages its final 101 training reports.

The preferred guidance duration differs across the two settings. Numerical value matching continues to pull toward a target gap, whereas a pairwise preference gradient weakens as the desired ordering gains margin. This distinction is consistent with the effectiveness of sustained preference supervision on KuaiSim.

Appendices C.5–C.6 report the complete schedule and K studies.

7 Conclusion, Limitations, and Future Work

Conclusion. The useful part of a completion is often the comparison it induces between alternatives. COMPASS turns these comparisons into reusable supervision through value contrasts or slate preferences. Across RL4RS and KuaiSim, the results suggest that supervision from an imperfect policy can remain useful after its absolute values become stale, provided that its action comparisons remain valid.

Limitations. COMPASS assumes access to alternative continuations from a shared prefix, limiting use where branching is unavailable or prohibitively expensive. Our value-update analysis assumes independently represented state–action values and population targets; it does not establish neural-network convergence. Unequal continuation gains can make an old ordering stale or reverse it (Section 5.2).

Future work. Selective refreshing could spend new rollout budget on states or action pairs whose margins are most threatened by changing continuation values. Learned world models or generative user simulators could replace exact branching, raising the question of how model error and continuation-policy shift interact. Finally, different learners may benefit from different comparison interfaces, from numerical gaps to weaker ordinal supervision.

Reproducibility Statement

Appendix A provides proofs. Appendices B and C specify the RL4RS and KuaiSim procedures, configurations, and per-seed results. Appendix D gives learning curves and computational records.

AI Use Statement

Large language models were used to assist with editing and polishing the manuscript's language.

REFERENCES

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1806.07857.

Marc G. Bellemare, Georg Ostrovski, Arthur Guez, Philip S. Thomas, and Rémi Munos. Increasing the action gap: New operators for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016. doi: 10.1609/aaai.v30i1.10303.

Cameron B. Browne, Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012. doi: 10.1109/TCIAIG.2012.2186810.

Will Dabney, Georg Ostrovski, David Silver, and Remi Munos. Implicit quantile networks for distributional reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 1096–1105, 2018. URL https://proceedings.mlr.press/v80/dabney18a.html.

Amir-massoud Farahmand. Action-gap phenomenon in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 24, 2011. URL https://papers.nips.cc/paper_files/paper/2011/hash/013d407166ec4fa56eb1e1f8cbe183b9-Abstract.html.

Noah Golowich and Ankur Moitra. Can Q-learning be improved with advice? In *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pp. 4548–4619. PMLR, 2022. URL https://proceedings.mlr.press/v178/golowich22a.html.

Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado P. van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, and Rémi Munos. Hindsight credit assignment. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/195f15384c2a79cedf293e4a847ce85c-Abstract.html.

Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. doi: 10.1609/aaai.v32i1.11796.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, Gabriel Dulac-Arnold, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Deep Q-learning from demonstrations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 3223–3230, 2018. doi: 10.1609/aaai.v32i1.11757.

Ian Holmes and Min Chi. Attention-based reward shaping for sparse and delayed rewards. *arXiv preprint arXiv:2505.10802*, 2025. URL https://arxiv.org/abs/2505.10802.

Haoxin Lin, Hongqiu Wu, Jiaji Zhang, Yihao Sun, Junyin Ye, and Yang Yu. Episodic return decomposition by difference of implicitly assigned sub-trajectory reward. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13808–13816, 2024. doi: 10.1609/aaai.v38i12.29287.

Thomas Mesnard, Theophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Thomas S Stepleton, Nicolas Heess, Arthur Guez, Eric Moulines, Marcus Hutter, Lars Buesing, and Remi Munos. Counterfactual credit assignment in model-free reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 7654–7664, 2021. URL https://proceedings.mlr.press/v139/mesnard21a.html.

Alexander Meulemans, Simon Schug, Seijin Kobayashi, Nathaniel Daw, and Gregory Wayne. Would I have gotten that reward? Long-term credit assignment by counterfactual contribution analysis. In *Advances in Neural Information Processing Systems*, 2023. doi: 10.52202/075280-3006.

Zhizhou Ren, Ruihan Guo, Yuan Zhou, and Jian Peng. Learning long-term reward redistribution via randomized return decomposition. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2111.13485.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and Shogi by planning with a learned model. *Nature*, 588:604–609, 2020. doi: 10.1038/s41586-020-03051-4.

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

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=A6Y7AqlzLW.

Yee Whye Teh, Victor Bapst, Wojciech M. Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, pp. 4496–4506, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/0abdc563a06105aee3c6136871c9f4d1-Abstract.html.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016. doi: 10.1609/aaai.v30i1.10295.

Kai Wang, Zhene Zou, Minghao Zhao, Qilin Deng, Yue Shang, Yile Liang, Runze Wu, Xudong Shen, Tangjie Lyu, and Changjie Fan. RL4RS: A real-world dataset for reinforcement learning based recommender system. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2935–2944. ACM, 2023. doi: 10.1145/3539618.3591899.

Kesen Zhao, Shuchang Liu, Qingpeng Cai, Xiangyu Zhao, Ziru Liu, Dong Zheng, Peng Jiang, and Kun Gai. KuaiSim: A comprehensive simulator for recommender systems. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-1945.

Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. Deep interest evolution network for click-through rate prediction. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pp. 5941–5948, 2019. doi: 10.1609/aaai.v33i01.33015941.

A Theory Details

This appendix provides the proofs and supporting calculations for Section 5. Appendix A.1 studies the centered value update and proves Theorem 1. Appendix A.2 derives the effect of changing the continuation policy and gives sufficient conditions for reusing a comparison. Appendix A.3 proves the return bound and calibrates the auxiliary losses to decision error. Appendix A.4 verifies the corresponding condition for the native slate decoder. Appendix A.5 gives the Best-of-K formulas and finite-sample estimates used in the experiments.

Throughout, the horizon is finite, rewards are bounded, and terminal values are zero.

A.1 Value dynamics and convergence

At stage h and state s, let $q = Q_h(s,\cdot),\ T = T_{K,h}(s,\cdot)$ denote the value and guidance vectors over the $m = |\mathcal{A}(s)|$ legal actions. Define the centering matrix $C = I - \frac{1}{m}\mathbf{1}\mathbf{1}^\top$, so that $C^\top = C,\ C^2 = C,\ C\mathbf{1} = 0$. With the Bellman target y held fixed during differentiation, the statewise objective is

$$\ell(q) = \frac{\|q-y\|_2^2}{2m} + \frac{\beta}{m}\sum_{a=1}^{m} h_\delta\left([C(q-T)]_a\right).$$

Let $\psi_\delta(x) = h'_\delta(x) = \min\{\delta, \max\{-\delta, x\}\}$. The gradient is $\nabla_q \ell = \frac{1}{m}\left[q - y + \beta C\psi_\delta(C(q-T))\right]$. Absorbing the common factor $1/m$ into the time scale gives the continuous-time update in Eq. (2): $\dot{Q} = BQ - Q - \beta(t)C\psi_\delta(C(Q-T_K))$. The finite-horizon Bellman operator and ψ_δ are Lipschitz. For any bounded measurable schedule $\beta(t)$, the resulting system therefore has a unique absolutely continuous solution from every finite initialization.

Statewise mean.

Multiplying the update by $\mathbf{1}^\top/m$ removes the guidance term: $\dot{\bar{q}} = \bar{y} - \bar{q}$. Thus the auxiliary gradient has no direct effect on the statewise mean. The Bellman target y, however, still depends on guided values at successor states.

Action-gap dynamics.

For two legal actions a, b, define $e_{ab} = (q_a - q_b) - (q_a^* - q_b^*)$ and $\zeta_{ab} = (T_a - T_b) - (q_a^* - q_b^*)$. Let $x = C(q-T)$. Define the secant slope $\omega_{ab} = \frac{\psi_\delta(x_a) - \psi_\delta(x_b)}{x_a - x_b} \in [0,1]$. When $x_a = x_b$, choose any value in $[0,1]$. If both residuals lie in the quadratic region of the Huber loss, then $\omega_{ab} = 1$.

Since $x_a - x_b = e_{ab} - \zeta_{ab}$, subtracting the two action updates gives $\dot{e}_{ab} = b_{ab} - (1+\beta\omega_{ab})e_{ab} + \beta\omega_{ab}\zeta_{ab}$, where $b_{ab} = \gamma(P_{h,a} - P_{h,b})(V_{Q,h+1} - V^*_{h+1})$. Let $\Lambda(t) = t + \int_0^t \beta(u)\omega_{ab}(u)\,du$. The integrating-factor solution is

$$e_{ab}(t) = e^{-\Lambda(t)}e_{ab}(0) + \int_0^t e^{-[\Lambda(t)-\Lambda(u)]}\left[b_{ab}(u) + \beta(u)\omega_{ab}(u)\zeta_{ab}\right]du.$$

If the continuation-value error is zero, β is constant, and both residuals remain in the Huber quadratic region, then

$$e_{ab}(t) = e^{-(1+\beta)t}e_{ab}(0) + \frac{\beta}{1+\beta}\left(1 - e^{-(1+\beta)t}\right)\zeta_{ab}.$$

For exact guidance, $\zeta_{ab} = 0$, so the action-gap error decays at rate $1+\beta$. With imperfect guidance, the limiting bias is determined by the supplied gap error.

Proof of Theorem 1.

Let $E_h(t) = \max_{s,a}|Q_h(s,a) - Q_h^*(s,a)|$, $E_H(t) = 0$, and define $\varepsilon_h = \max_s \mathrm{sp}\left(T_{K,h}(s,\cdot) - Q_h^*(s,\cdot)\right)$, where $\mathrm{sp}(x) = \max_i x_i - \min_i x_i$.

Fix a state attaining a positive maximal error and write $v = q - q^*,\ f = T - q^*$. If coordinate a attains the positive maximum, then for every j, $x_j - x_a = (v_j - v_a) + (f_a - f_j) \le \varepsilon_h$. By monotonicity and the unit Lipschitz constant of ψ_δ, $[C\psi_\delta(x)]_a \ge -\varepsilon_h$. The corresponding reverse inequality holds for a coordinate attaining a negative maximal error. Since the error in the next-stage Bellman target is at most γE_{h+1}, $\frac{d}{dt}E_h \le -E_h + \gamma E_{h+1} + \beta(t)\varepsilon_h$ almost everywhere.

Because the maximum of finitely many absolutely continuous coordinates is itself absolutely continuous, the inequality remains valid when the maximizing coordinate changes. Backward comparison with the resulting triangular linear system gives

$$E_h(t) \le e^{-t} \sum_{j=0}^{H-h-1} \frac{(\gamma t)^j}{j!} E_{h+j}(0)$$

$$+ \int_0^t e^{-(t-u)} \sum_{j=0}^{H-h-1} \frac{[\gamma(t-u)]^j}{j!} \beta(u)\varepsilon_{h+j}\, du.$$

If the centered guidance is exact, then $\varepsilon_h = 0$ for all h, and Q^* is an equilibrium. The finite polynomial multiplying e^{-t} converges to zero. Moreover, since $\gamma \le 1$, $\max_h E_h(t) \le \max_h E_h(0) e^{-t} \sum_{j=0}^{H-1} \frac{t^j}{j!} \le \max_h E_h(0)$, which gives stability.

Now suppose the guidance scores are bounded and $\beta(t) \to 0$. Split the convolution term at any $U > 0$. The contribution from $[0, U]$ tends to zero as $t \to \infty$. The contribution from $[U, t]$ is bounded by a constant multiple of $\sup_{u \ge U} \beta(u)$. Taking $t \to \infty$ and then $U \to \infty$ yields $Q(t) \to Q^*$. For constant β, the same bound gives $\limsup_{t\to\infty} E_h(t) \le \beta \sum_{j=0}^{H-h-1} \gamma^j \varepsilon_{h+j}$. This proves Theorem 1.

Discrete updates.

Consider the synchronous iteration $Q^{n+1} = Q^n + \eta\left[BQ^n - Q^n - \beta_n C\psi_\delta(C(Q^n - T))\right]$, with $0 \le \beta_n \le B,\ 0 < \eta \le (1+B)^{-1}$. For a fixed Bellman target, the difference between two updates can be written as $[I - \eta(I + \beta_n CDC)]\,(q - \tilde{q})$, where D is a diagonal secant matrix satisfying $0 \preceq D \preceq I$. Its Euclidean operator norm is at most $1 - \eta$.

For exact comparisons, define $D_h^n = \max_s \|Q_h^n(s,\cdot) - Q_h^*(s,\cdot)\|_2$ and $M = \max_s |\mathcal{A}(s)|$. Then $D_h^{n+1} \le (1-\eta)D_h^n + \eta\gamma\sqrt{M}\, D_{h+1}^n$. Backward induction from $D_H^n = 0$ gives convergence.

For inexact scores, the bound $\|C\psi_\delta(\cdot)\|_2 \le \delta\sqrt{M}$ adds the term $\eta\beta_n\delta\sqrt{M}$. If $\beta_n \to 0$, this perturbation vanishes and the same backward-induction argument applies.

These statements concern synchronous tabular value updates. They do not establish convergence of the nonlinear neural implementations used in the experiments.

A.2 Changing the continuation policy

For any policy π, $Q_h^\pi(s,a) = r_h(s,a) + \gamma P_{h,a} V_{h+1}^\pi$. Subtracting the Bellman equations for policies π and μ after the same initial action gives $Q_h^\pi(s,a) - Q_h^\mu(s,a) = \gamma P_{h,a}\left(V_{h+1}^\pi - V_{h+1}^\mu\right)$. Subtracting the corresponding equations for actions a and b yields Eq. (3):

$$\begin{aligned}(Q_h^\pi(s,a) - Q_h^\pi(s,b)) - (Q_h^\mu(s,a) - Q_h^\mu(s,b)) \\ = \gamma(P_{h,a} - P_{h,b})(V_{h+1}^\pi - V_{h+1}^\mu).\end{aligned}$$

Thus an action gap is unchanged whenever the two branches receive equal expected continuation gains.

Implemented-score error.

Let $A_K^\mu = Q_K^\mu - V_K^\mu$ denote the population Best-of-K completion advantage. Insert this quantity into the implemented score error: $T_K - Q^* = (T_K - A_K^\mu) + (Q_K^\mu - Q^\mu) + (Q^\mu - Q^*) - V_K^\mu$.

For a comparison set $\mathcal{C}$, the shared reference term vanishes under the action span, giving

$$\mathrm{sp}_{\mathcal{C}}(T_K - Q_h^*) \le \mathrm{sp}_{\mathcal{C}}(T_K - A_{K,h}^{\mu}) + \mathrm{sp}_{\mathcal{C}}(Q_{K,h}^{\mu} - Q_h^{\mu}) + \gamma\, \mathrm{sp}_{a\in\mathcal{C}}\, P_{h,a}(V_{h+1}^* - V_{h+1}^{\mu}).$$

The three terms correspond to:

1. estimation, fitting, and optional shrinkage;
2. the change from the mean completion return to Best-of-K;
3. unequal continuation gains between μ and the target policy.

The second term is zero when $K = 1$.

Uniform reuse over a policy family.

Consider two actions a, b. Let $\Delta T = T(s,a) - T(s,b)$ and $\Delta Q^{\mu} = Q^{\mu}(s,a) - Q^{\mu}(s,b)$. Suppose $|\Delta T - \Delta Q^{\mu}| \le e$. For a policy family Π, define $D_{\Pi} = \sup_{\pi\in\Pi} \left|\gamma(P_{h,a} - P_{h,b})(V_{h+1}^{\pi} - V_{h+1}^{\mu})\right|$. If $\Delta T > e + D_{\Pi}$, then $Q_h^{\pi}(s,a) > Q_h^{\pi}(s,b)$ for every $\pi \in \Pi$.

This follows directly from Eq. (3) and the triangle inequality. A single simultaneous error bound is sufficient even if the later policy is selected after observing the completion evidence.

For finite transition distributions, $|\gamma(P_a - P_b)U| \le \gamma\, \mathrm{TV}(P_a, P_b)\, \mathrm{sp}(U)$. The bound follows by subtracting the midpoint of the range of U.

Example 1: a weak completion policy with an exact action gap.

Consider an undiscounted two-step MDP. At the initial state s_0, actions a and b lead deterministically to states x and y, respectively, with zero immediate reward.

At x, second-stage actions ℓ, u have rewards $(1, M+1)$, and at y, $(0, M)$, with $M > 0$.

Let the completion policy choose b initially and ℓ thereafter. Its initial intervention values are $Q^{\mu}(s_0,a) = 1,\ Q^{\mu}(s_0,b) = 0$. The optimal values are $Q^*(s_0,a) = M+1,\ Q^*(s_0,b) = M$. Hence $Q^{\mu}(s_0,a) - Q^{\mu}(s_0,b) = Q^*(s_0,a) - Q^*(s_0,b) = 1$, even though the completion policy can be arbitrarily worse in absolute return as M increases.

More generally, if the continuation chooses u with probability p at both successors, the initial values become $1 + pM$ and pM, and the gap remains 1.

Example 2: ordering reversal.

Modify only the rewards at state x to $(1,1)$, and let $M > 1$. The completion-policy values remain $(1,0)$, whereas the optimal values become $(1, M)$. The action ordering therefore reverses.

The completion returns are deterministic in this example. Additional repeats, or a larger value of K, cannot repair the comparison. The difference between the two examples is the unequal continuation gain after the two initial actions.

A.3 Decision loss and auxiliary-loss calibration

Proof of Theorem 2.

Fix stage h and state s. Let $a_{\mathcal{C}}^* \in \arg\max_{a\in\mathcal{C}_h(s)} Q_h^*(s,a)$, and let $a^{\pi} \in \mathcal{C}_h(s)$ be the action selected by policy π.

Using the definitions from Section 5.3,

$$\begin{aligned} V_h^*(s) - Q_h^*(s,a^{\pi}) &= c_h(s) + (Q_h^* - T_K)(s, a_{\mathcal{C}}^*) \\ &\quad - (Q_h^* - T_K)(s, a^{\pi}) \\ &\quad + T_K(s, a_{\mathcal{C}}^*) - T_K(s, a^{\pi}) \\ &\le c_h(s) + \epsilon_h(s) + r_{T,h}(s). \end{aligned}$$

Along the trajectory induced by π, the Bellman identity telescopes:

$$J^* - J(\pi) = \sum_{h=0}^{H-1} \gamma^h \mathbb{E}_{s\sim d_h^\pi} \left[V_h^*(s) - Q_h^*(s, a^\pi) \right].$$

Substituting the pointwise bound gives

$$J^* - J(\pi) \leq \sum_{h=0}^{H-1} \gamma^h \mathbb{E}_{s\sim d_h^\pi} \left[c_h(s) + \epsilon_h(s) + r_{T,h}(s) \right].$$

The terminal term is zero because terminal values are zero and both policies use the same initial-state distribution.

Random comparison sets are also allowed by conditioning on the realized set, provided that the selected action belongs to it.

Sharpness of the three terms.

Each coefficient in Theorem 2 can be attained separately in a one-step problem.

Take two actions with rewards $(g, 0)$ and let the policy select the second action.

If the comparison set is $\{2\}$, then $(c, \epsilon, r_T) = (g, 0, 0)$. If the full set is used with guidance scores $(0, 0)$, then $(c, \epsilon, r_T) = (0, g, 0)$. If the full set is used with guidance scores $(g, 0)$, then $(c, \epsilon, r_T) = (0, 0, g)$. Each construction attains decision loss g.

Calibration of the centered Huber loss.

Let z denote the learner scores over m compared actions. Let $a_z \in \arg\max_a z_a$, $a_T \in \arg\max_a T_a$, and define $x = C(z - T)$. By definition of the guidance shortfall, $x_{a_z} - x_{a_T} \geq r_T$. If $r_T > 0$, the two coordinates are distinct. Convexity and symmetry of the Huber loss imply

$$\begin{aligned} mL_{\text{value}} &\geq h_\delta(x_{a_z}) + h_\delta(x_{a_T}) \\ &\geq 2h_\delta\left(\frac{x_{a_z} - x_{a_T}}{2}\right) \geq 2h_\delta(r_T/2). \end{aligned}$$

Therefore, $r_T \leq 2h_\delta^{-1}\left(\frac{mL_{\text{value}}}{2}\right)$, where

$$h_\delta^{-1}(y) = \begin{cases} \sqrt{2y}, & y \leq \delta^2/2, \\ y/\delta + \delta/2, & y > \delta^2/2. \end{cases}$$

In the quadratic region, $r_T \leq 2\sqrt{mL_{\text{value}}}$. Equality is attained by $T = (r/2, -r/2, 0, \ldots, 0)$, $z = 0$, with ties resolved toward the second action.

Calibration of the slate-preference loss.

For m proposed slates, let $p = \frac{m(m-1)}{2}$ be the number of unordered pairs. Write

$$L_{\text{pair}} = \frac{1}{p} \sum_{i<j} |T_i - T_j| \operatorname{softplus}\left[-\operatorname{sign}(T_i - T_j)(z_i - z_j) \right].$$

Let a_T maximize the guidance score and a_z maximize the actor score. The pair (a_T, a_z) contributes at least $r_T \log 2$ to the numerator because its signed actor margin is nonpositive. Hence $r_T \leq \frac{pL_{\text{pair}}}{\log 2}$. For $m = 1$, both the pairwise loss and the guidance shortfall are zero.

This calibration assumes that the comparison set contains the native actor-score-maximizing slate and that the decoder uses the same item scores, catalog, and slate length. A training-batch average yields a return guarantee only with additional control over the state distribution induced by the executed policy.

A.4 THE NATIVE SLATE DECODER

Let u_i be the actor's item score and define $w_i = e^{u_i} > 0$. For a slate $L = (i_1, \ldots, i_\ell)$, the decoder likelihood is $P_u(L) = \prod_{j=1}^{\ell} \frac{w_{i_j}}{\sum_{k \notin \{i_1,\ldots,i_{j-1}\}} w_k}$. We show that, at fixed slate length ℓ, the descending top-ℓ list maximizes $P_u(L)$, and therefore also maximizes the normalized log likelihood used in training.

First consider two adjacent selected items a, b, with remaining total weight S before either is selected. The likelihood ratio between the two possible orders is $\frac{P_{ab}}{P_{ba}} = \frac{S-w_b}{S-w_a}$. If $w_a \geq w_b$, then $\frac{P_{ab}}{P_{ba}} \geq 1$. Thus putting the larger-weight item first cannot reduce the slate likelihood.

Next, suppose a selected item b at position j is replaced by an omitted item a with $\Delta = w_a - w_b > 0$. Let S_k denote the original remaining total weight at position k. Then $\frac{P_u(L')}{P_u(L)} = \frac{w_a}{w_b} \prod_{k=j+1}^{\ell} \frac{S_k}{S_k - \Delta} > 1$. All denominators are positive because every remaining catalog contains the item selected at the next position.

Therefore any likelihood-maximizing slate must contain the ℓ largest item weights and order them nonincreasingly. Ties may yield multiple maximizers.

Consequently, if the native top-ℓ slate is included in the comparison set, the maximizing condition used in Appendix A.3 holds for the same item scores and decoder. An action modified by exploration need not satisfy this condition.

A.5 COMPLETION STATISTICS AND ESTIMATION

Best-of-K.

Let F be an integrable return distribution with quantile function q. If $G = q(U)$, $U \sim \mathrm{Unif}(0,1)$, then the maximum of K independent uniforms has CDF u^K. Therefore $B_K(F) = \int_0^1 q(u)\, K u^{K-1}\, du$. For quantile-bin endpoints $0 = e_0 < e_1 < \cdots < e_N = 1$, define $w_i^{(K)} = e_i^K - e_{i-1}^K$. If q_i represents the return in bin i, the Best-of-K mean is $\widehat{B}_K = \sum_i w_i^{(K)} q_i$, and the corresponding variance is $(\sigma_K)^2 = \sum_i w_i^{(K)} (q_i - \widehat{B}_K)^2$. For n sorted empirical returns, we use $e_i = i/n$. This computes the expected maximum of K draws with replacement from the empirical distribution. It is not the average of the largest observed returns.

For independent intervention and reference maxima, the dispersion of their difference is $\sigma_K^A = \sqrt{(\sigma_K^Q)^2 + (\sigma_K^V)^2}$.

Effect on action gaps.

If the action-return distributions differ only by additive shifts, $G_a \overset{d}{=} m_a + Z$, then their Best-of-$K$ gaps satisfy $B_K(F_a) - B_K(F_b) = m_a - m_b$. Best-of-$K$ can, however, change an ordering when the return distributions cross.

For example, a constant return 0.4 has larger mean than a Bernoulli return with success probability 0.1: $0.4 > 0.1$. For $K = 8$, the Bernoulli Best-of-K value is $1 - 0.9^8 \approx 0.570$, which exceeds 0.4.

Sensitivity to distribution error.

Let q and $\hat{q}$ be the quantile functions of F and $\hat{F}$. Then $\left|B_K(\hat{F}) - B_K(F)\right| \leq \int_0^1 |\hat{q}(u) - q(u)| K u^{K-1}\, du$. Since $K u^{K-1} \leq K$, we obtain $\left|B_K(\hat{F}) - B_K(F)\right| \leq K W_1(F, \hat{F})$. The error in an action difference is bounded by the sum of the corresponding two branch bounds. Any shared reference term cancels when taking action differences.

Finite-sample mean comparisons.

Consider N action comparisons fixed before collection. For comparison j, suppose we observe n independent paired repeats $D_{j,i} = G_{j,i}(a) - G_{j,i}(b)$, with branch returns in $[0, R]$.

Let $\bar{D}_j = \frac{1}{n} \sum_{i=1}^{n} D_{j,i}$ and let Δ_j^{μ} denote its population mean.

By Hoeffding's inequality and a union bound, with probability at least $1 - \alpha$,

$$\forall j \in \{1, \dots, N\} : \qquad |\bar{D}_j - \Delta_j^{\mu}| \le R\sqrt{\frac{2\log(2N/\alpha)}{n}}.$$

This bound applies simultaneously to all comparisons fixed before collection.

B RL4RS Implementation and Results

This appendix gives the complete RL4RS experimental protocol used in Section 6. Appendix B.1 specifies the online value-learning procedure. Appendices B.2–B.3 describe offline evidence construction, completion-model fitting, and guidance schedules. Appendix B.4 defines final evaluation. Appendix B.5 records the comparison methods and implementation settings. Appendices B.6–B.7 specify the ablation, pretraining, reward-injection, and Best-of-K controls. Appendix B.8 reports backbone reuse and all individual seed results.

B.1 Training procedure

The completion model is fitted before online RL and remains frozen throughout each training run.

For DQN, each update samples a minibatch from replay and computes the ordinary TD objective. Before numerical action masking, the Q-network is evaluated on all legal actions at each replay state. The frozen completion model is queried on the same states and legal actions. Its outputs are detached from the computation graph.

Let $q_s(a) = Q_\theta(s, a)$, $t_s(a) = T_K(s, a)$. Both vectors are centered over the legal-action set, $C_s q_s(a) = q_s(a) - \frac{1}{|\mathcal{A}(s)|}\sum_{b\in\mathcal{A}(s)} q_s(b)$, and analogously for $C_s t_s$. The auxiliary loss is evaluated only over legal actions and combined with the TD objective as $L = L_{\mathrm{TD}} + \beta L_{\mathrm{value}}$. Only the Q-network receives gradients from L_{value}. The completion model and all quantities used to construct T_K remain fixed.

Legal-action masking is applied during environment interaction and in the maximization used by the TD target. Numerical masking values are not included in centering or in the auxiliary loss.

For IQN, the same procedure is applied after replacing each scalar action value in L_{value} by the mean of its predicted quantiles. IQN retains its original distributional TD objective.

B.2 RL4RS evidence and completion-model fitting

Data splits.

Source sessions are shuffled before partitioning. Rows $[0, 50000)$ are used for completion construction, $[50000, 58192)$ are reserved for final held-out evaluation, and the remaining sessions are used for online RL.

Each RL4RS task contributes 135,000 construction prefixes. Saved prefix indices identify the state features, recorded intervention action, and associated return records.

Completion rollouts.

For an intervention rollout, the recorded action is forced at the selected prefix and the fixed completion policy acts thereafter. The reference rollout follows the same completion policy directly from the shared prefix.

The completion policy is fitted only on the construction split. Its exploration probability is 0.2. Completion returns use discount factor 1. The decision horizon is 9 for RL4RS-A and 36 for RL4RS-B.

RL4RS-A uses one base completion return per branch and prefix for distribution fitting. RL4RS-B uses all valid adaptive repeats, starting with one repeat and increasing selected prefixes to at most four repeats within each GPU shard.

Collection does not enumerate every legal action at every prefix. Conditional models pool observations across states and actions.

Return-distribution models.

The intervention-return model takes state and action as input, whereas the reference-return model takes state only. Both use hidden widths 256 and 128. Action-conditioned models use a 32-dimensional action embedding.

Each return distribution is represented by 128 quantiles at $\tau_i = \frac{i-\frac{1}{2}}{128},\ i = 1, \ldots, 128$. The models are trained with quantile Huber loss and Adam for 50 epochs.

For the return-distribution fitting split, seed 0 partitions prefixes 90/10 into training and validation subsets. Repeats associated with the same prefix remain in the same partition. Input normalization uses statistics computed on the training prefixes.

Valid training returns from the intervention and reference branches jointly determine a shared target mean and standard deviation. A constant 10^{-6} is added to each normalization denominator. The checkpoint with the lowest validation quantile loss is selected.

Predicted quantiles are transformed back to reward units, sorted, and reweighted according to Appendix A.5 to obtain the Best-of-K advantage and dispersion targets.

Completion-advantage model.

The completion-advantage model uses the same hidden widths, 256 and 128, and the same 32-dimensional action embedding. Hidden layers use ReLU activations.

The model contains separate output heads for the completion advantage and $\log(1 + \sigma_K^A)$. Both outputs are trained with Smooth-L1 loss, with weight 0.5 on the log-dispersion term.

This fitting stage standardizes its inputs and both targets using statistics from its own training split, again adding 10^{-6} to normalization denominators.

Seed 0 reserves 4,096 records for validation. Training runs for 100 epochs, and the checkpoint with the lowest total validation loss is selected.

Predictions from the selected model are transformed back to the original target units. The main RL4RS experiments use $K = 64,\ \kappa = 1$. After checkpoint selection, the completion-advantage model is frozen for all downstream RL runs associated with the task.

Table 2: RL4RS evidence and completion-model fitting configurations.

Configuration	RL4RS-A	RL4RS-B
Construction sessions	50,000	50,000
Covered prefix records	135,000	135,000
Completions used for distribution fitting	Base sample per prefix	All valid repeats
Training/validation return samples per branch	121,500 / 13,500	188,416 / 20,832
Completion-model state input dimension	266	293
Action catalog size	284	284
Action embedding dimension	32	32
Return/advantage model hidden widths	256, 128	256, 128
Quantiles per return distribution	128	128
Return-distribution training/validation records	121,500 / 13,500	121,500 / 13,500
Advantage-model training/validation records	130,904 / 4,096	130,904 / 4,096
Return-distribution / advantage fitting epochs	50 / 100	50 / 100
Selected return-distribution / advantage epoch	5 / 95	3 / 17
Main K/κ	64 / 1	64 / 1

B.3 Online training and guidance schedules

Online learning retains DQN's replay and TD updates on environmental rewards and adds the centered auxiliary loss to the same optimization step.

DQN uses one hidden layer of width 64, Double Q-learning, and no dueling architecture. It uses discount factor 1, one-step TD targets, replay capacity 100,000, and target-network update interval 200 steps.

Training batch size refers to the samples used for optimization and is distinct from the parallel environment batch size.

An epoch is one iteration of the outer training loop and indexes the main guidance schedule. Online budgets and learning-curve axes use cumulative environment interactions recorded during collection.

Table 3: Main RL4RS online-learning settings.

Configuration	RL4RS-A	RL4RS-B
Decision horizon	9	36
Parallel environment batch size	64	64
Training batch size	576	1,024
Training epochs	500	500
Recorded main-run interaction steps	504,000	504,000
Initial guidance weight β_0	1	1
K/κ	64 / 1	64 / 1

Main epoch schedule.

Let $e \in \{0, \ldots, 499\}$ denote the zero-based training epoch. The main experiments use

$$\beta(e) = \begin{cases} 1, & 0 \le e \le 50, \\ \dfrac{1 + \cos(\pi(e-50)/200)}{2}, & 50 < e < 250, \\ 0, & e \ge 250. \end{cases}$$

Guidance is therefore zero throughout the second half of training. Training logs use one-based epoch numbering, so logged epoch 251 is the first epoch with zero guidance.

Interaction-step schedule.

The reward-interface and Best-of-K comparisons in Appendix B.7 use a separate schedule indexed by cumulative environment interactions.

Let n denote the interaction count. We use

$$\beta(n) = \begin{cases} 0.5, & 0 \le n \le 100000, \\ 0.25\left[1 + \cos(\pi(n-100000)/350000)\right], & 100000 < n < 450000, \\ 0, & n \ge 450000. \end{cases}$$

The main epoch schedule and this interaction-step schedule define separate experimental configurations.

Action masking.

Sampling and greedy action selection always obey the legal-action mask. Auxiliary centering uses raw Q-values over legal actions before numerical masking.

Only the Q-network receives auxiliary gradients.

B.4 Final evaluation

Every final policy acts greedily with legal-action masking on all 8,192 held-out sessions, evaluated in four batches of 2,048. Evaluation uses the original environmental rewards.

Let $R_{(1)} \le \cdots \le R_{(8192)}$ denote the pooled session returns sorted in ascending order. The run-level score is $\widehat{J} = \frac{1}{8111} \sum_{i=1}^{8111} R_{(i)}$, which removes the highest 81 returns, corresponding to the top 1%.

Across training seeds, we report the mean and sample standard deviation of these run-level scores with equal weight per seed. Per-seed tables additionally report the median.

For stored session-bootstrap intervals, the 8,192 session returns are resampled with replacement 2,000 times. The 2.5th and 97.5th percentiles of the recomputed trimmed score define the interval. These intervals describe finite-session sampling within a run and are separate from variation across training seeds.

Training curves use behavior-policy returns rather than held-out greedy evaluation. Appendix D.2 specifies their aggregation and smoothing.

B.5 Comparison methods and implementation settings

RL4RS-A and RL4RS-B use the same comparison set: COMPASS and eleven baselines. DQN provides the direct backbone comparison. PPO, A2C, IQN, Rainbow, and Graph-DQN cover alternative policy-optimization procedures, value representations, and network structures. RRD, ARES, COCOA, VFD, and DiAster provide comparisons with alternative treatments of delayed feedback.

The baselines are tuned separately, and the main results use the selected tuned configurations. Standard value-learning baselines include Double Q-learning (van Hasselt et al., 2016), IQN (Dabney et al., 2018), and the RL4RS Rainbow configuration (Hessel et al., 2018). PPO follows its clipped policy-optimization interface (Schulman et al., 2017). The delayed-feedback implementations adapt RRD (Ren et al., 2022), ARES (Holmes & Chi, 2025), COCOA (Meulemans et al., 2023), and DiAster (Lin et al., 2024).

Decision networks.

Decision networks use one hidden layer of width 64, except Rainbow, which uses width 128. These widths refer to the decision backbone; auxiliary models used for return decomposition, attention-based credit allocation, or completion supervision have their own architectures.

PPO and A2C both use discount factor 1, GAE parameter $\lambda = 1$, and value-loss coefficient 0.5.

PPO uses one SGD pass per batch, KL coefficient 0.2, and target KL 0.01.

Graph-DQN uses an item-vector table covering all 284 actions. The item vectors remain frozen during training.

Delayed-feedback methods.

RRD, ARES, COCOA, and DiAster fit auxiliary return models from online trajectories. Their implementation groups every 9 decisions into a page and produces step-level rewards before the trajectories enter replay.

The auxiliary models read 256-dimensional state features and retain the most recent 256 pages. Each auxiliary update samples at most 8 pages. One update is performed after at least 16 new pages have been collected since the preceding auxiliary update.

These auxiliary models use Adam with learning rate 10^{-3}. RRD uses a reward network with two hidden layers of width 128 and a random-subsequence fraction of 0.5.

ARES uses a causal-attention network with width 64, four attention heads, two blocks, and 32-dimensional action embeddings.

COCOA uses a return-conditioned action model and an action-return model, each with two hidden layers of width 128. It samples 8 random actions per state for return comparisons, uses 32-dimensional action embeddings, and divides return inputs by 200.

DiAster uses a single-layer GRU with hidden dimension 128 and an MLP step-reward model with two hidden layers of width 128.

These architectures belong to the auxiliary return models; their downstream decision backbone remains the DQN described above.

Reward-redistribution methods modify the training signal, whereas final evaluation always uses the original environment rewards defined in Appendix B.4. VFD retains the environmental reward and instead adds its auxiliary terms to the value-learning objective.

Table 4: Comparison methods and their RL4RS implementations.

Method	Implementation and settings used in this study
DQN	Masked Double DQN with one-step TD targets and no dueling architecture.
COMPASS	The same decision backbone as DQN, with centered comparison supervision from the frozen completion-advantage model; see Appendix B.3.
PPO	Masked policy and value learning with learning rate 10^{-4}, SGD minibatch size 256, and clipping parameter 0.3.
A2C	Masked actor–critic with learning rate 10^{-4}, entropy coefficient 0.01, and gradient-clipping threshold 10.
IQN	Implicit quantile value representation with 8 current and 8 target quantile samples, cosine embedding dimension 64, and quantile Huber threshold 1.
Rainbow	RL4RS implementation with action masking, hidden width 128, Double Q-learning, one-step targets, one output atom, and both dueling and noisy layers disabled.
Graph-DQN	A hidden-width-64 state network produces a representation whose inner product with each frozen 40-dimensional item vector gives its Q-value.
RRD	Fits page returns using random subsequences, rescales predicted scores into step rewards, and trains DQN on those rewards.
ARES	Predicts page returns from ordered state–action sequences, computes allocation scores through position-specific attention, and trains DQN on the resulting rewards.
COCOA	Combines return-conditioned action prediction with action-return comparisons to generate step rewards within each page, followed by DQN learning.
VFD	Adds a state-value output to the Q-network, uses next-state value in the action TD target, and adds a squared loss between state value and the selected action's Q-value.
DiAster	Decomposes returns with a GRU subtrajectory-return model and an MLP step-reward model, followed by DQN learning.

Every method uses the same session partitions, legal-action constraints, held-out evaluation metric, and approximately 504,000 online environment interactions. The final collection batch is allowed to finish, so the exact interaction endpoint may slightly exceed the nominal budget. Epoch counts can differ because collection sizes differ.

COMPASS's reusable offline evidence is constructed separately as described in Appendix B.2.

B.6 Ablations and pretraining

All design controls in this section use RL4RS-A seeds 0–2, 500 training epochs, and the same 504,000-interaction budget unless stated otherwise.

Design controls.

No centering. The centered residual $C_s Q_\theta - C_s T_K$ is replaced by the raw residual $Q_\theta - T_K$ over legal actions. All remaining settings follow the main configuration.

Permuted action correspondence. Guidance scores are reassigned to legal actions using a seed- and state-dependent permutation. For a fixed seed and state, the permutation remains unchanged across repeated queries.

Reversed comparison signs. Guidance scores are negated: $T_K(s,a) \leftarrow -T_K(s,a)$. **Constant guidance.** We set $\beta = 1$ for all 500 epochs.

Mean-return comparisons. We replace the main $K = 64$ completion statistic by $K = 1$ while retaining $\kappa = 1$ and the remaining main settings.

Pretraining controls.

Both pretraining controls use the existing 135,000 RL4RS-A construction prefixes drawn from 48,006 sessions.

The pretraining Q-network reads 256 state features, uses a tanh hidden layer of width 64, and outputs 284 action values.

Table 5: RL4RS-A design controls: individual seeds and aggregate held-out return.

Setting	Seed 0	Seed 1	Seed 2	Mean ± SD
COMPASS, $K = 64$	157.33	154.63	148.64	153.53 ± 4.45
Mean-return comparisons, $K = 1$	149.34	145.05	144.66	146.35 ± 2.60
Permuted action correspondence	140.55	145.31	138.76	141.54 ± 3.39
Without centering	128.43	125.72	126.25	126.80 ± 1.43
Constant guidance weight	117.75	119.52	120.10	119.12 ± 1.22
Reversed comparison signs	113.43	112.88	112.56	112.96 ± 0.44
DQN	122.24	131.04	128.80	127.36 ± 4.57

For each seed, construction prefixes are split by session into fitting and diagnostic subsets. A session-grouped split reserves 4,801 sessions per seed for diagnostics.

The fitting/diagnostic prefix counts are $121365/13635$ for seed 0, $121519/13481$ for seed 1, and $121566/13434$ for seed 2.

Comparison pretraining.

Comparison pretraining uses the main frozen completion model with $K = 64,\ \kappa = 1$, reconstructed legal-action masks, and the centered Huber loss.

It applies the 27,889 positive guidance weights from the corresponding COMPASS run in their original order. Each update samples 576 fitting states with replacement, giving $16,064,064$ state presentations in total.

Adam uses learning rate 5×10^{-4}, $\epsilon = 10^{-8}$, and gradient clipping at 40. The final update supplies the pretrained parameters.

MC-return pretraining.

MC-return pretraining averages the valid intervention returns associated with each prefix. The stored cache contains 209,248 returns with between one and four repeats per prefix.

State features and recorded actions are joined by prefix index. Squared error is used to fit the corresponding Q-network output directly in raw return units.

Adam uses learning rate 3×10^{-4} and minibatch size 512. Training runs for at most 100 epochs with early-stopping patience 10 on diagnostic MSE. The selected checkpoint is epoch 100 for all three seeds.

Online phase after pretraining.

Both controls copy the pretrained parameters into the online and target Q-networks, initialize a fresh optimizer and empty replay buffer, and start ordinary DQN with zero completion guidance.

Each receives 55,889 online updates and the full environment-interaction budget.

The matched COMPASS runs also receive 55,889 online updates, including 27,889 updates with positive guidance weight.

Pretraining uses cached construction states, whereas continued COMPASS supervision evaluates the frozen completion model on states drawn from the evolving replay distribution.

Table 6: RL4RS-A pretraining controls: individual seeds and aggregate held-out return.

Setting	Seed 0	Seed 1	Seed 2	Mean ± SD
DQN	122.24	131.04	128.80	127.36 ± 4.57
MC-return pretraining followed by DQN	134.64	139.38	133.62	135.88 ± 3.07
Comparison pretraining followed by DQN	134.23	131.14	131.58	132.32 ± 1.67
COMPASS	157.33	154.63	148.64	153.53 ± 4.45

B.7 Reward injection and completion statistics

This section compares reward injection and centered auxiliary supervision using the same underlying RL4RS-A completion evidence.

All runs use seeds 0–2, 500 training epochs, and 504,000 environment interactions. The same return-distribution models are used across the interface comparison.

For each value of K, the corresponding fitted completion-advantage model is shared by the two interfaces.

Reward injection.

The reward-injection arm uses $r'_t = r_t + \widehat{A}_K(s_t, a_t)$ with constant coefficient 1.

No dispersion shrinkage or auxiliary value loss is used in this arm. Final evaluation always uses the original environmental reward.

Centered auxiliary supervision.

The auxiliary-supervision arm uses T_K with $\kappa = 1$ and the interaction-step schedule from Appendix B.3: $\beta_0 = 0.5$, held for 100,000 interactions and then cosine-decayed over 350,000 interactions.

The TD target continues to use the original environmental reward.

Best-of-K sweep.

The sweep evaluates $K \in \{1, 2, 4, 8, 16, 32, 64, 128\}$. For each K, Table 7 reports the mean offline intervention statistic Q_K, reference statistic V_K, and resulting advantage statistic A_K over the 135,000 construction prefixes, followed by the policy returns for reward injection and auxiliary supervision.

Table 7: Evidence interfaces and Best-of-K under the interaction-step schedule.

K	Q_K	V_K	A_K	Reward injection	Auxiliary supervision
1	80.60	82.40	-1.79	128.15 ± 5.76	134.56 ± 9.72
2	89.15	91.02	-1.87	126.42 ± 1.93	127.02 ± 2.53
4	96.39	98.23	-1.84	128.12 ± 4.28	126.57 ± 0.68
8	102.81	104.53	-1.72	123.62 ± 0.36	125.29 ± 2.70
16	108.69	110.18	-1.49	128.21 ± 5.17	129.65 ± 6.44
32	114.18	115.38	-1.20	127.17 ± 3.92	131.67 ± 1.76
64	119.25	120.16	-0.91	126.62 ± 2.01	135.05 ± 1.04
128	123.61	124.24	-0.64	123.55 ± 1.49	135.02 ± 5.27

Unguided DQN on the same three seeds obtains 127.36 ± 4.57. Increasing K raises both branch statistics, but neither their difference nor downstream policy return is monotone in K.

The main epoch-schedule experiments and the interaction-step experiments in this section use separate guidance schedules.

B.8 Backbone reuse and individual results

DQN and IQN reuse.

The eight-seed RL4RS-A reuse study uses the same frozen completion-advantage model for DQN and IQN.

DQN applies the centered loss to its scalar action values. IQN applies the same loss to mean action values obtained by averaging its quantile predictions while retaining its native quantile TD objective.

No additional completion collection or completion-model fitting is performed for IQN.

The corresponding learning trajectories are reported in the backbone-reuse figure.

Main RL4RS per-seed results.

Tables 9 and 10 report every seed used in the five-seed main comparison.

Table 8: Eight-seed backbone reuse on RL4RS-A: held-out return.

Seed	DQN	DQN + COMPASS	IQN	IQN + COMPASS
0	122.24	157.33	131.98	161.17
1	131.04	154.63	133.63	155.01
2	128.80	148.64	124.98	139.61
3	127.77	153.18	130.02	148.97
4	122.73	150.05	131.02	156.53
5	134.86	135.30	130.84	157.12
6	129.20	148.38	138.43	156.01
7	128.63	155.04	165.88	167.60
Mean ± SD	128.16 ± 4.13	150.32 ± 6.87	135.85 ± 12.70	155.25 ± 8.26

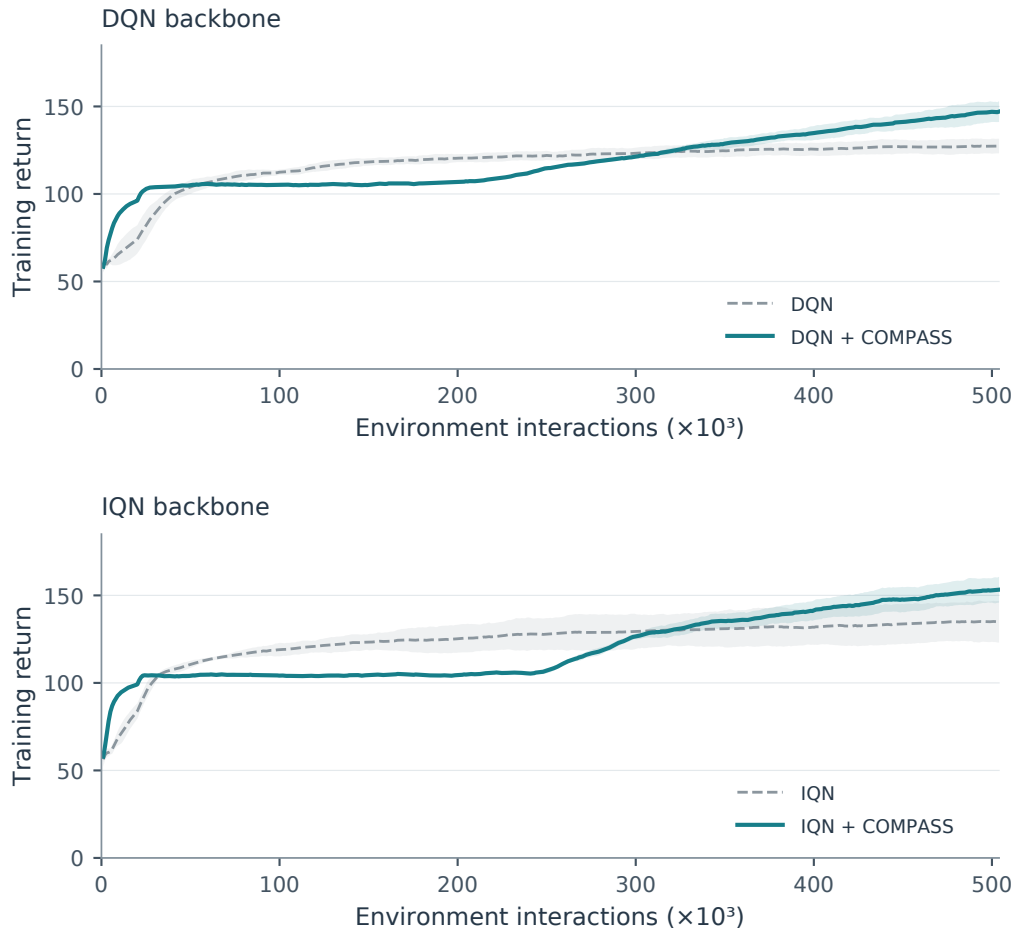


Figure 7: DQN and IQN learning trajectories on RL4RS-A, eight seeds per arm. Lines and bands show mean and sample SD after the per-seed interpolation and smoothing described in Appendix D.2.

Table 9: RL4RS-A held-out return for all five seeds, with mean, sample SD, and median.

Method	Seed 0	Seed 1	Seed 2	Seed 3	Seed 4	Mean ± SD	Median
COMPASS	157.33	154.63	148.64	153.18	150.05	152.77 ± 3.50	153.18
DQN	122.24	131.04	128.80	127.77	122.73	126.52 ± 3.87	127.77
PPO	146.43	143.59	132.55	144.89	132.06	139.90 ± 7.01	143.59
A2C	107.38	113.35	115.01	112.06	113.79	112.32 ± 2.95	113.35
Rainbow	132.78	136.37	128.17	131.92	129.73	131.80 ± 3.13	131.92
IQN	131.98	133.63	124.98	130.02	131.02	130.33 ± 3.27	131.02
Graph-DQN	131.63	131.22	136.64	136.50	133.57	133.91 ± 2.58	133.57
RRD	145.82	135.10	127.32	150.07	137.89	139.24 ± 8.97	137.89
ARES	159.46	129.83	126.75	134.29	129.11	135.89 ± 13.46	129.83
COCOA	105.16	102.33	109.88	108.52	107.72	106.72 ± 3.00	107.72
VFD	120.07	128.04	122.42	133.45	123.74	125.54 ± 5.28	123.74
DiAster	117.81	115.26	127.27	120.01	125.79	121.23 ± 5.15	120.01

Table 10: RL4RS-B held-out return for all five seeds, with mean, sample SD, and median.

Method	Seed 0	Seed 1	Seed 2	Seed 3	Seed 4	Mean ± SD	Median
COMPASS	395.12	346.07	424.29	333.70	394.98	378.83 ± 37.76	394.98
DQN	254.96	232.92	268.34	261.22	306.04	264.70 ± 26.64	261.22
PPO	377.31	323.37	381.29	394.23	463.95	388.03 ± 50.36	381.29
A2C	258.08	207.51	219.38	220.05	358.06	252.62 ± 61.94	220.05
Rainbow	222.17	366.51	315.82	295.54	261.13	292.23 ± 54.68	295.54
IQN	251.58	242.50	302.64	232.33	301.24	266.06 ± 33.46	251.58
Graph-DQN	166.72	163.19	109.67	132.42	198.74	154.15 ± 34.20	163.19
RRD	168.69	259.15	255.22	232.71	207.79	224.71 ± 37.44	232.71
ARES	166.68	232.73	242.07	240.03	220.27	220.35 ± 31.19	232.73
COCOA	217.67	227.91	251.53	202.95	222.39	224.49 ± 17.74	222.39
VFD	249.28	204.39	163.17	261.71	201.18	215.94 ± 39.82	204.39
DiAster	228.11	180.86	144.00	155.38	226.15	186.90 ± 39.08	180.86

C KuaiSim Implementation and Results

This appendix gives the complete KuaiSim protocol used for the slate-preference experiments in Section 6. Appendix C.1 defines the slate supervision objective and proposal construction. Appendix C.2 specifies the environment, learner configurations, and evaluation protocol. Appendix C.3 describes completion collection. Appendix C.4 gives the slate-level completion model and fitting procedure. Appendices C.5–C.6 report policy results, Best-of-K sensitivity, and guidance-duration diagnostics.

C.1 Slate supervision

At each replay state, we construct 64 continuous-action proposals: the deterministic actor output with dropout disabled, the ordinary policy call, 31 local perturbations of the deterministic output, and 31 noise-only proposals. Noise components are sampled independently from $U[0, 0.1)$ and clipped to the actor bounds $[-1, 1]$.

The environment's native item-scoring function and top-six decoder map each proposal to an ordered slate of six distinct items. If multiple proposals decode to the same ordered slate, only the first occurrence is retained.

For a slate $L = (i_1, \ldots, i_6)$, let $u_\theta(s, i)$ denote the actor score for item i. We define the differentiable slate score

$$\ell_\theta(L \mid s) = \frac{1}{6} \sum_{j=1}^{6} \left[u_\theta(s, i_j) - \log \sum_{i \in \mathcal{I} \setminus \{i_1, \ldots, i_{j-1}\}} e^{u_\theta(s,i)} \right].$$

Dropout is disabled while evaluating ℓ_θ, but gradients through the actor are retained. The previous actor mode is restored afterward.

Let $\mathcal{P}(s)$ index all unordered pairs of distinct proposed slates. For $(i, j) \in \mathcal{P}(s)$, define $\Delta T_{ij} = T_K^{\text{slate}}(s, L_i) - T_K^{\text{slate}}(s, L_j)$ and $\Delta \ell_{ij} = \ell_\theta(L_i \mid s) - \ell_\theta(L_j \mid s)$. The pairwise supervision loss is

$$L_{\text{pair}} = \mathbb{E}_{s\sim\mathcal{D}}\left[\frac{\sum_{(i,j)\in\mathcal{P}(s)} |\Delta T_{ij}| \operatorname{softplus}\left[-\operatorname{sign}(\Delta T_{ij})\Delta \ell_{ij}\right]}{\max(1, |\mathcal{P}(s)|)}\right].$$

The denominator counts all distinct pairs, including pairs with zero guidance gap. Zero-gap pairs contribute zero loss. States with fewer than two distinct decoded slates also contribute zero.

The actor minimizes $L_{\text{DDPG,actor}} + \beta L_{\text{pair}}$. Gradients flow through the slate likelihoods to the actor parameters and its user/item representations. The continuous proposals, completion model, and completion-model feature encoder are detached. The critic retains its original TD loss. Recommendations during interaction use the native six-item decoder.

The main KuaiSim configuration uses $K = 1,\ \kappa = 0,\ \beta = 1$ for all 2,000 online updates.

C.2 Environment, learners, and evaluation

KuaiSim produces item-level click feedback after each six-item recommendation slate. The per-step reward is the mean click feedback over the six items, and session return is the sum of these rewards until user departure. Users begin with temper 20 and leave when temper falls below 1. The configured maximum session length is 20. A departed parallel slot is immediately filled by a new session.

All methods use the same frozen response model and item representations. A fixed shuffle with seed 20260916 assigns 17,616 users to online training and reserves 1,958 disjoint users. Reported session returns use the online-training user pool.

Each run first warms up replay for 100 parallel environment steps. It then collects one parallel environment step and performs one learning update per iteration for 2,000 iterations.

The shared state encoder uses three Transformer layers with width 32, four attention heads, and feed-forward width 64. User and item latent dimensions are 16. TD3 uses two Q critics; A2C uses a state-value critic; DDPG, HAC, and COMPASS use one Q critic. Actor and critic weight decay is 10^{-5}. A2C uses entropy coefficient 0.1 and advantage bias 0. HAC uses behavior learning rate 10^{-4}, weight decay 3×10^{-5}, and hyper-actor coefficient 0.1.

Matched DDPG and COMPASS seeds start from identical actor and critic parameters. Departure is terminal for every method. TD targets use $y_t = r_t + \gamma(1 - d_t)\operatorname{stopgrad}(v_{t+1})$, where d_t denotes departure.

The monitor first averages the returns of sessions departing within the same parallel step, then averages the latest ten such records for each report. Reports occur every ten logged steps from step 10 through step 1990.

The run-level final score averages reports 99–199, corresponding to logged steps 990–1990, for a total of 101 reports per seed. Tables report the mean and sample standard deviation across the five training seeds. The final-window aggregation is used only for reported endpoint performance; figure captions specify any shorter smoothing window used to display training trajectories.

C.3 Collecting slate completions

A fixed behavior-cloning policy, denoted BC, is used both to construct prefixes and to complete sessions. BC samples six items without replacement. Completion collection uses the online-training user pool.

Table 11: KuaiSim task and online-training settings.

Setting	Value
Items per recommendation	6
Parallel sessions	32
Initial temper	20
Item-correlation coefficient	0.2
Reward	Mean click feedback within the slate
TD discount factor	0.9
Warm-up collection	100 parallel environment steps
Learning iterations	2,000
Replay minibatch size	128
Replay capacity	100,000
Target-network soft-update coefficient	0.01
Actor action-head hidden widths	256, 64
Critic hidden widths	256, 64
State-encoder and critic dropout	0.1
Training seed labels	1, 2, 3, 4, 5

Table 12: Actor and critic learning rates in KuaiSim.

Method	Actor learning rate	Critic learning rate
DDPG	10^{-4}	10^{-3}
TD3	10^{-5}	10^{-3}
A2C	10^{-5}	10^{-3}
HAC	10^{-5}	10^{-3}
COMPASS	10^{-4}	10^{-3}

For the user-disjoint fitting and validation split, we hash the decimal-encoded user ID with SHA-256. The first four bytes are interpreted as a little-endian integer, and users whose resulting value is zero modulo five are assigned to validation. All remaining users are assigned to fitting. Log rows are sampled without replacement within the corresponding partition.

For each sampled row, we restore the logged history, reset temper to 20, advance BC by 0, 4, or 8 steps, and retain the resulting state only if it is nonterminal.

Each prefix receives eight distinct intervention slates. Four are sampled from the BC item probabilities and four are sampled uniformly from the item catalog. Items are sampled without replacement within each slate.

For an intervention branch, we restore the prefix, force the selected slate once, and then follow BC until departure. The shared reference branch restores the same prefix and follows BC directly from the current decision.

Returns begin at the prefix and use discount factor $\gamma = 0.9$. For corresponding intervention and reference repeats, the prefix state and completion random seed are reset to the same values, pairing stochastic completion noise across branches.

For R paired repeats, the $K = 1$ slate target is

$$\widehat{A}_1(s, L) = \frac{1}{R} \sum_{r=1}^{R} \left[G^{(r)}(s, L) - G^{(r)}_{\text{ref}}(s) \right].$$

The fitting split uses $R = 16$, whereas validation uses $R = 128$. The additional validation repeats are used to obtain more stable model-selection targets.

All 4,608 saved prefixes are distinct, and fitting and validation users do not overlap.

The resulting slate-completion model is reused across all online seeds in the main KuaiSim experiment.

Table 13: KuaiSim completion evidence by fitting and validation partition.

Quantity	Fitting	Validation
Distinct users	3,335	468
Total prefixes	4,096	512
Prefixes at depth 0	1,408	192
Prefixes at depth 4	1,408	160
Prefixes at depth 8	1,280	160
Intervention slates per prefix	8	8
State–slate pairs	32,768	4,096
Repeats per branch	16	128
Completion trajectories	589,824	589,824

C.4 Fitting the slate advantage model

The frozen BC encoder produces a 154-dimensional state feature vector. Six state dimensions whose standard deviation over fitting prefixes is at most 10^{-6} are removed, leaving 148 state features.

Each item is represented by 102 features obtained by concatenating its frozen BC embedding with public item metadata. The catalog contains 5,659 items.

State features are normalized using fitting-prefix statistics. Item features are normalized using statistics over the public catalog. A constant 10^{-6} is added to each normalization denominator. The saved normalization statistics are reused unchanged during online RL.

For state s and slate L, the fitted model is

$$\widehat{A}_\phi(s, L) = \frac{1}{6} \sum_{i \in L} f_\phi(\tilde{x}(s), \tilde{e}_i) - b_\phi(\tilde{x}(s)).$$

The shared state–item network f_ϕ uses hidden widths 128 and 64. The state baseline b_ϕ uses hidden width 64. Both networks use SiLU activations.

The predictor is additive over items and therefore invariant to item order within the slate. It is fitted to slate-level return differences and can score item combinations not observed during completion collection. The actor likelihood in Appendix C.1 remains order-sensitive.

For prefix s, let $L_1, \ldots, L_8$ denote the eight intervention slates and $a_{s1}, \ldots, a_{s8}$ their targets. Define $e_{sj} = \widehat{A}_\phi(s, L_j) - a_{sj}$ and $\bar{e}_s = \frac{1}{8} \sum_{j=1}^{8} e_{sj}$. Each minibatch contains all eight slates associated with each of 64 prefixes. The fitting objective is

$$L_{\text{fit}} = \frac{1}{|\mathcal{B}| s_A^2} \sum_{s \in \mathcal{B}} \left[\frac{1}{8} \sum_{j=1}^{8} (e_{sj} - \bar{e}_s)^2 + 0.1 \bar{e}_s^2 \right],$$

where s_A^2 is the sample variance of the 32,768 fitting targets.

The first term fits within-state slate differences. The offset penalty constrains the statewise baseline. Prefixes are shuffled each epoch, and checkpoint selection uses the lowest within-state centered validation MSE.

After checkpoint selection, the completion model, feature statistics, and item representations are frozen. With $\kappa = 0$, its predicted advantage directly supplies the slate-level guidance score.

Validation diagnostics are computed on the 512 model-selection prefixes. Correlation is computed after centering predictions and targets within each prefix. Ordering agreement uses all $\binom{8}{2} = 28$ slate pairs per prefix.

For the gap-filtered diagnostic, a pair is retained when the magnitude of its estimated mean return difference exceeds two standard errors, with standard errors estimated from the 128 paired validation repeats.

Table 14: KuaiSim completion-advantage model fitting settings.

Setting	Value
Optimizer	Adam
Learning rate	10^{-3}
Weight decay	10^{-5}
Batch size	64 prefixes, with 8 slates each
Training epochs	80
Model initialization and data-shuffling seed	61
Selection criterion	Within-state centered validation MSE
Selected epoch	7

Table 15: Validation diagnostics for the KuaiSim completion-advantage model.

Metric	Value
Within-state centered MSE	0.001309
Correlation of centered predictions and targets	0.5890
Ordering agreement over all slate pairs	70.07%
Pairs whose absolute mean return gap exceeds two standard errors	45.61%
Ordering agreement on those pairs	82.06%

C.5 Policy results and Best-of-K sensitivity

The main KuaiSim experiment uses $K = 1,\ \kappa = 0,\ \beta = 1$. Table 16 reports the final-window scores over five seeds. Figure 8 shows the corresponding training trajectories.

Table 16: KuaiSim final-window session returns, five seeds.

Seed	DDPG	TD3	A2C	HAC	DDPG + COMPASS
1	8.595	9.366	9.933	8.918	9.485
2	8.607	8.722	9.029	9.166	10.009
3	8.929	8.844	9.271	9.434	9.791
4	8.588	9.053	8.631	9.057	9.518
5	8.588	9.360	9.174	9.567	9.926
Mean $\pm$ SD	8.661 ± 0.150	9.069 ± 0.293	9.207 ± 0.473	9.228 ± 0.268	9.746 ± 0.236

Extended Best-of-K sweep.

The Best-of-K sweep uses the same 4,096 fitting prefixes and 512 validation prefixes as the main experiment, but increases the number of repeats to 256 per branch.

The expanded evidence contains 10,616,832 completion trajectories, including 9,437,184 trajectories in addition to those used for the main completion set.

For sorted branch returns $g_{(1)} \le \cdots \le g_{(256)}$, the empirical Best-of-$K$ target is

$$\widehat{B}_K = \sum_{j=1}^{256} \left[\left(\frac{j}{256} \right)^K - \left(\frac{j-1}{256} \right)^K \right] g_{(j)}.$$

Each value of K receives its own completion model fitted with the procedure in Appendix C.4. The online learner settings and five training seeds are unchanged across K.

Because the sweep uses 256 repeats per branch, its $K = 1$ model is not identical to the main-experiment $K = 1$ model.

Mean return is highest at $K = 1$ in this sweep; increasing tail emphasis does not improve the five-seed mean.

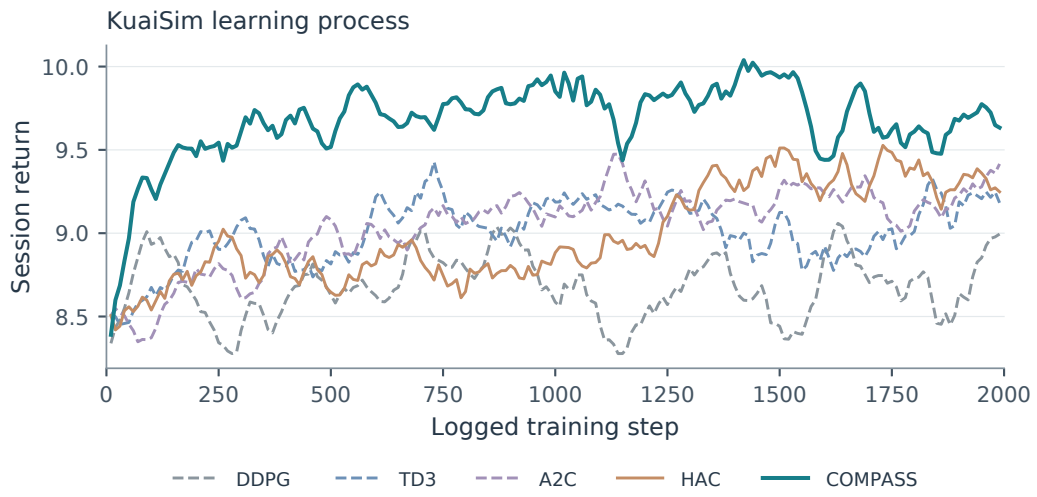


Figure 8: KuaiSim training trajectories for the five main methods. Lines show the across-seed mean after trailing-five-report smoothing. These are training-monitor returns; Table 16 uses the final-window aggregation defined in Appendix C.2.

Table 17: KuaiSim Best-of-K sensitivity on the expanded completion evidence.

K	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Mean $\pm$ SD
1	9.641	10.077	9.734	9.857	9.721	9.806 $\pm$ 0.170
4	9.498	9.846	9.779	9.657	9.947	9.745 $\pm$ 0.174
8	9.669	9.996	9.679	9.545	9.809	9.740 $\pm$ 0.171
16	9.456	9.867	9.661	9.769	9.854	9.721 $\pm$ 0.170
32	9.487	9.883	9.720	9.553	9.854	9.699 $\pm$ 0.177
64	9.661	9.727	9.796	9.600	9.731	9.703 $\pm$ 0.075

C.6 Guidance duration and preference agreement

The constant- and decaying-guidance arms use the same frozen slate-completion model, matched actor and critic initializations, the same five seeds, and identical replay, environment, and optimization settings. Only the guidance schedule differs.

The constant arm uses $\beta_n = 1$ for all 2,000 learning updates.

The decaying arm uses

$$\beta_n = \begin{cases} 1, & 0 \le n \le 200, \\ \frac{1}{2}\left[1 + \cos(\pi(n-200)/800)\right], & 200 < n < 1000, \\ 0, & n \ge 1000. \end{cases}$$

Both arms run for 2,000 updates.

To measure the constraints encountered during training, we record the unweighted comparison loss and pairwise ordering agreement every 32 learning updates and again at update 2,000.

Ordering agreement is computed over nonzero-guidance-gap pairs within each replay batch and is not weighted by $|\Delta T|$. Actor-score ties count as disagreements.

Comparison loss is averaged over pairs within each state and then over the 128 replay states in the minibatch.

Table 18: KuaiSim decaying and constant guidance: final-window returns.

Seed	Decay	Constant	Difference
1	9.062	9.485	0.423
2	9.400	10.009	0.609
3	9.583	9.791	0.207
4	8.896	9.518	0.622
5	9.375	9.926	0.551
Mean ± SD	9.263 ± 0.278	9.746 ± 0.236	0.482 ± 0.173

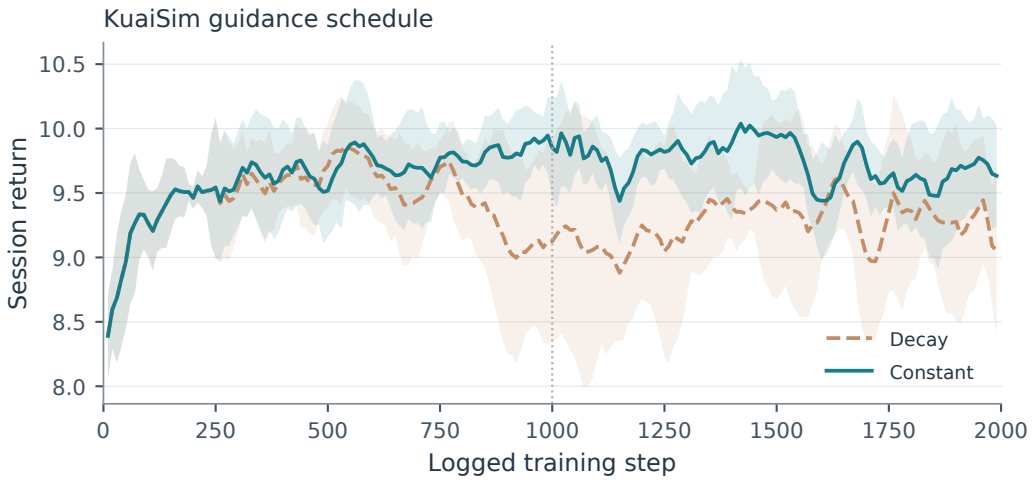


Figure 9: Constant and decaying guidance on KuaiSim. Lines and bands show mean ± one sample standard deviation over five seeds after trailing-five-report smoothing. The dotted line marks the update at which the decaying arm reaches zero guidance.

The reported windows contain 6, 16, and 17 diagnostic records per seed, respectively. Records are first averaged within each seed and then summarized across the five seeds. The two schedule arms share identical early diagnostic records before their schedules diverge.

Table 19: Preference-loss and ordering-agreement diagnostics across training windows.

Learning-update window	Schedule	Comparison loss	Ordering agreement
32–192	Decay / Constant	0.02010 ± 0.00186	73.01% ± 3.05%
512–992	Decay	0.03101 ± 0.01799	73.59% ± 6.78%
512–992	Constant	0.00999 ± 0.00281	82.12% ± 3.03%
1504–2000	Constant	0.00953 ± 0.00287	83.35% ± 3.02%

Under constant guidance, comparison loss decreases after the early window and ordering agreement rises to approximately 83%. Diagnostics for the decaying arm stop once its guidance weight reaches zero. Because replay batches change throughout training, these measurements describe the constraints encountered by the learner rather than convergence on a fixed comparison set.

D Learning Curves and Computation

This appendix reports completion-collection cost, online runtime, and the full RL4RS training trajectories underlying the aggregate results in Section 6.

D.1 Offline construction and online cost

Each RL4RS task uses 135,000 construction prefixes. Distribution fitting uses 270,000 intervention/reference returns in RL4RS-A and 418,496 in RL4RS-B. Completion-trajectory lengths depend on prefix position.

The main KuaiSim completion set contains 1,179,648 trajectories, 12,453,550 branch interactions, and 17,792 prefix-advancement interactions. The extended Best-of-K sweep adds 9,437,184 completion trajectories and 99,443,573 branch interactions.

Each task's fitted completion model is shared across its downstream runs. The RL4RS-A completion model is additionally shared between the DQN and IQN backbone experiments.

RL4RS timings cover complete rollout collection. For KuaiSim, timing begins after the environment, models, and data pools have been loaded and includes prefix construction, completion rollout, and saving.

Table 20: Completion collection time by environment and evidence set.

Collection stage	GPUs	Min/GPU	GPU-min
RL4RS-A completion collection	2	Approx. 40	Approx. 80
RL4RS-B completion collection	4	Approx. 48	Approx. 192
KuaiSim main evidence	1	3.28	3.28
KuaiSim additional collection for the K-sweep	2	11.96–12.00	23.96

The main KuaiSim slate-advantage model requires an additional 9.8 seconds to train for 80 epochs, perform validation, select a checkpoint, and save the selected parameters. This timing excludes later diagnostics and inference export.

Online KuaiSim timings include replay warm-up, 2,000 learning updates, logging, and checkpoint saving after environment and network construction.

During one COMPASS run, the frozen completion model evaluates 16,384,000 proposed-slate queries through batched forward passes. Together with the auxiliary actor updates, these computations add approximately 99 seconds relative to DDPG in the recorded runs.

Table 21: KuaiSim online training time, mean and sample standard deviation across five seeds.

Method	Recorded time (s, mean $\pm$ SD)
DDPG	239.0 ± 9.9
TD3	236.6 ± 4.6
A2C	256.9 ± 4.6
HAC	301.3 ± 2.7
COMPASS	338.2 ± 1.3

D.2 Training trajectories

The following figures report the individual training trajectories for the main RL4RS methods and the matched three-seed controls.

The horizontal axis is cumulative environment interaction count. The vertical axis is the trainer's behavior-policy episode return and therefore includes exploration.

Runs complete their final collection batch after reaching the nominal 504,000-interaction budget. The largest recorded endpoint is 505,296 interactions.

For each method, trajectories are restricted to the interaction interval covered by all included seeds and interpolated onto a grid spaced every 1,008 interactions. No extrapolation is used.

A trailing 20-point moving average is then applied separately to each seed. Thin lines show individual smoothed trajectories. The emphasized line shows their across-seed mean, and the band shows one sample standard deviation.

The reward-injection training monitor includes the injected reward and therefore uses a separate vertical scale from methods trained only on environmental rewards. Final held-out evaluation for the reward-injection arm uses the original environmental reward.

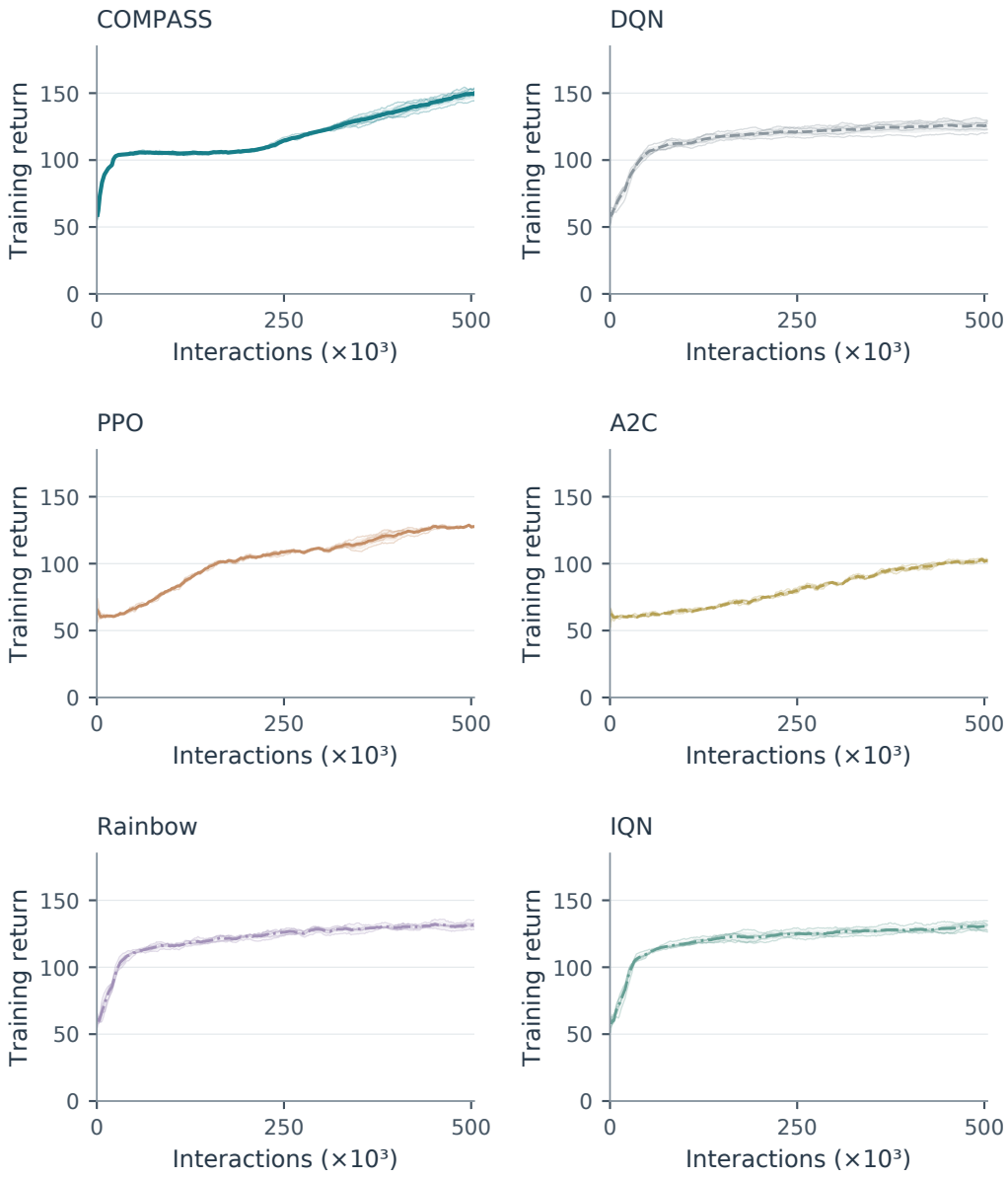


Figure 10: RL4RS-A individual training trajectories for COMPASS, DQN, PPO, A2C, Rainbow, and IQN. Thin lines show each seed after the common trailing-20-point smoothing. The emphasized line and band show the across-seed mean and one sample standard deviation.

The matched design-control trajectories use the seeds and intervention definitions in Appendix B.6. Reward-injection trajectories use the interaction-step configuration in Appendix B.7.

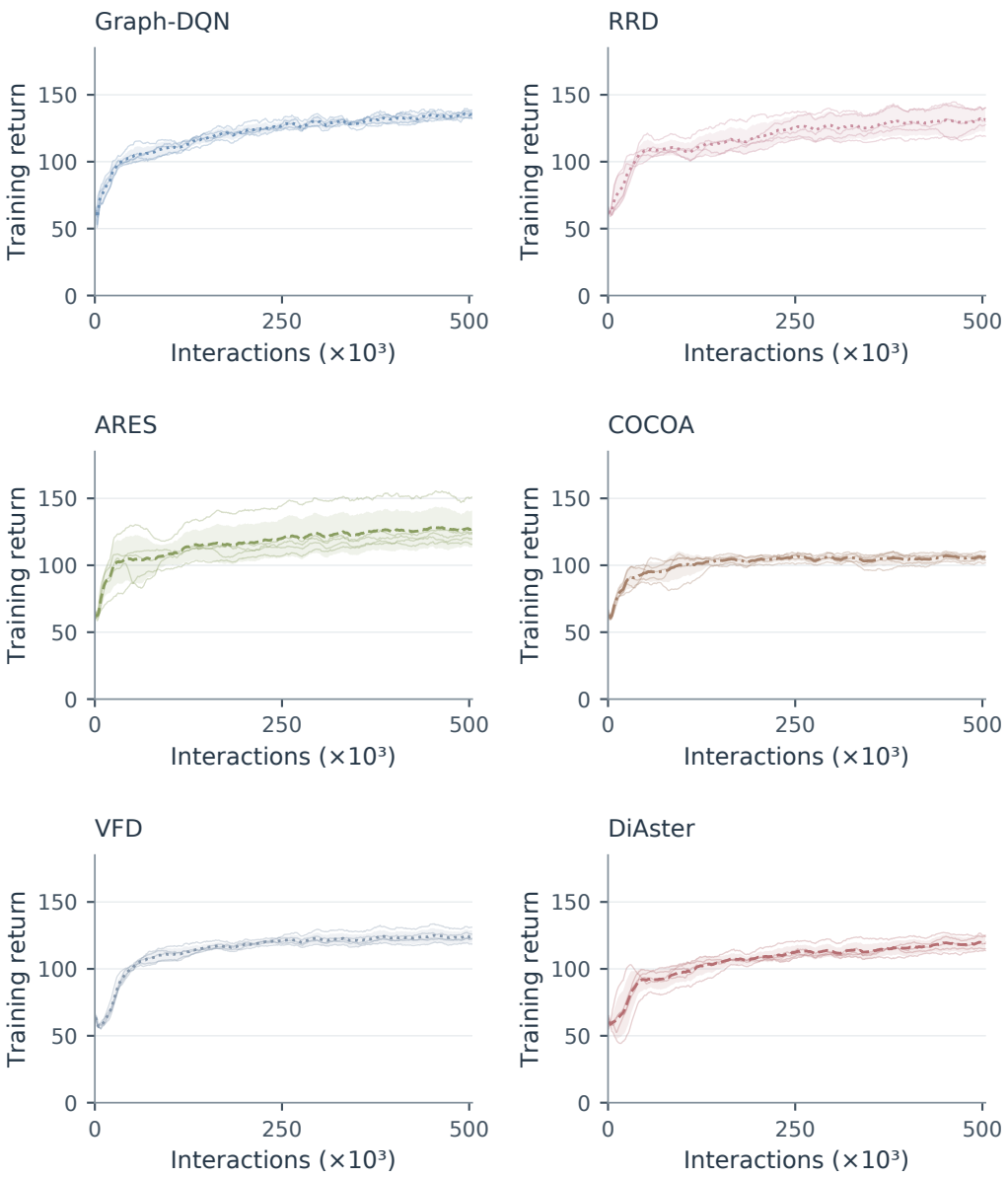


Figure 11: RL4RS-A individual training trajectories for Graph-DQN, RRD, ARES, COCOA, VFD, and DiAster. Plotting follows the same procedure as Figure 10.

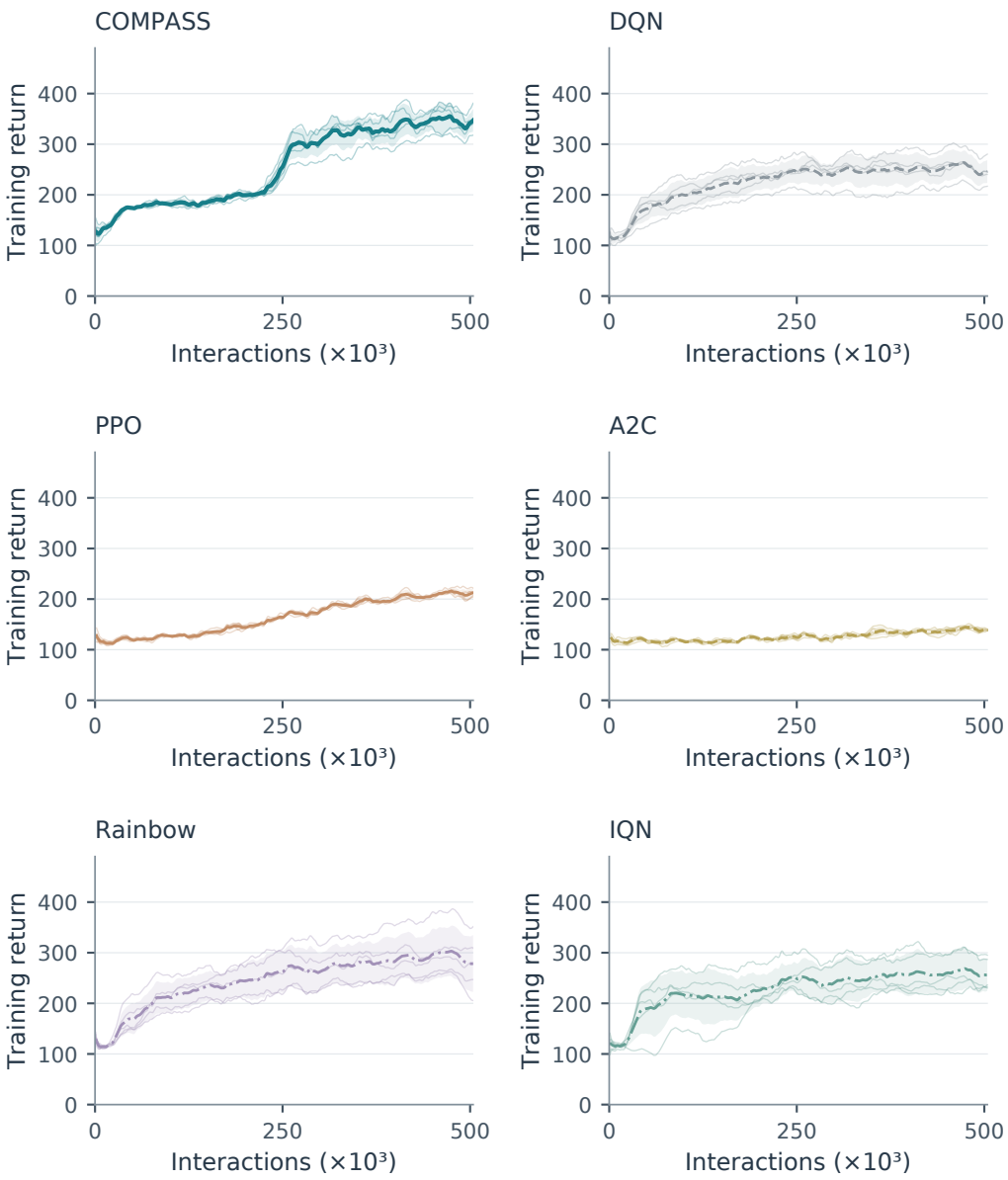


Figure 12: RL4RS-B individual training trajectories for COMPASS, DQN, PPO, A2C, Rainbow, and IQN. Thin lines show each seed after the common trailing-20-point smoothing. The emphasized line and band show the across-seed mean and one sample standard deviation.

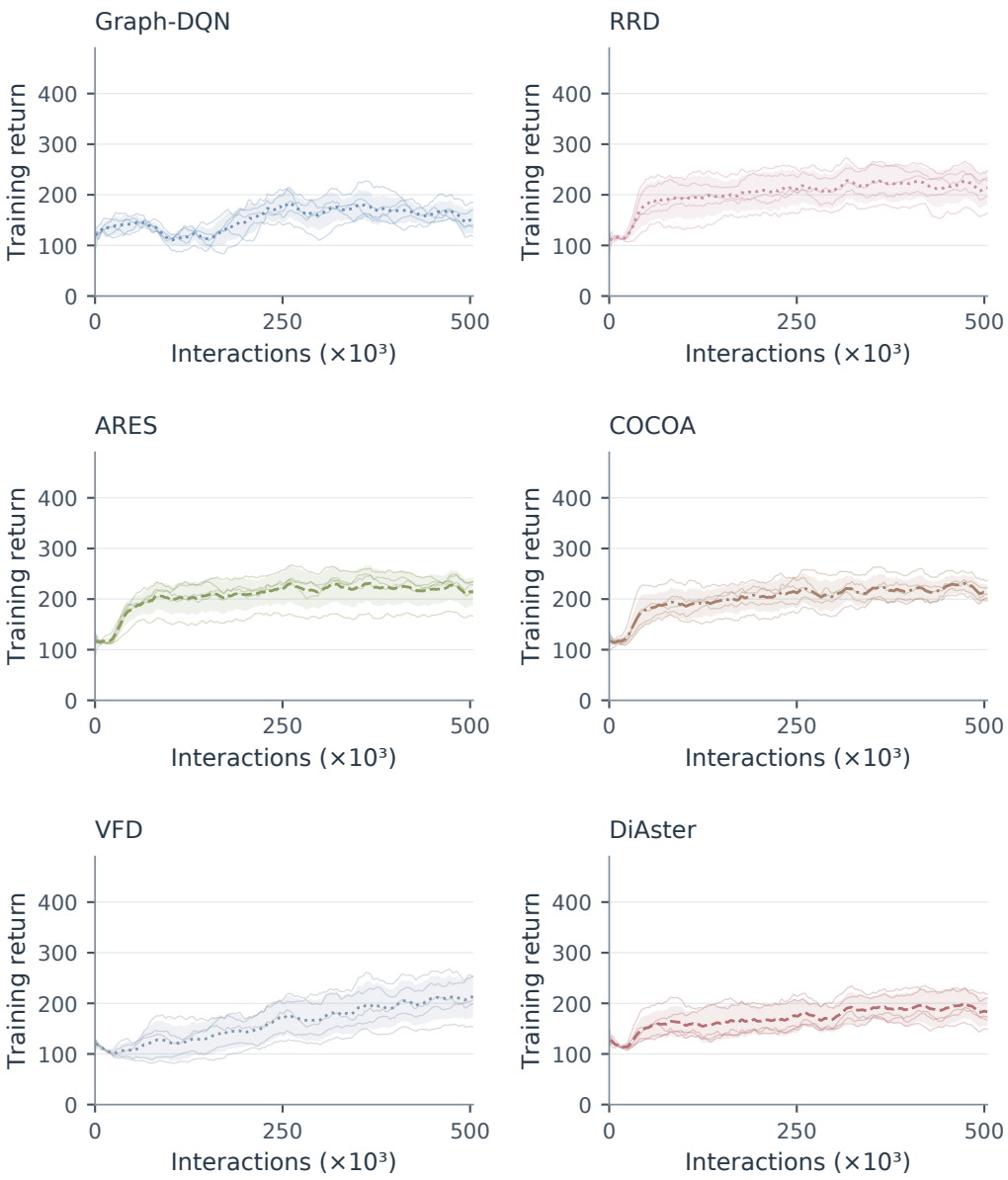


Figure 13: RL4RS-B individual training trajectories for Graph-DQN, RRD, ARES, COCOA, VFD, and DiAster. Plotting follows the same procedure as Figure 12.

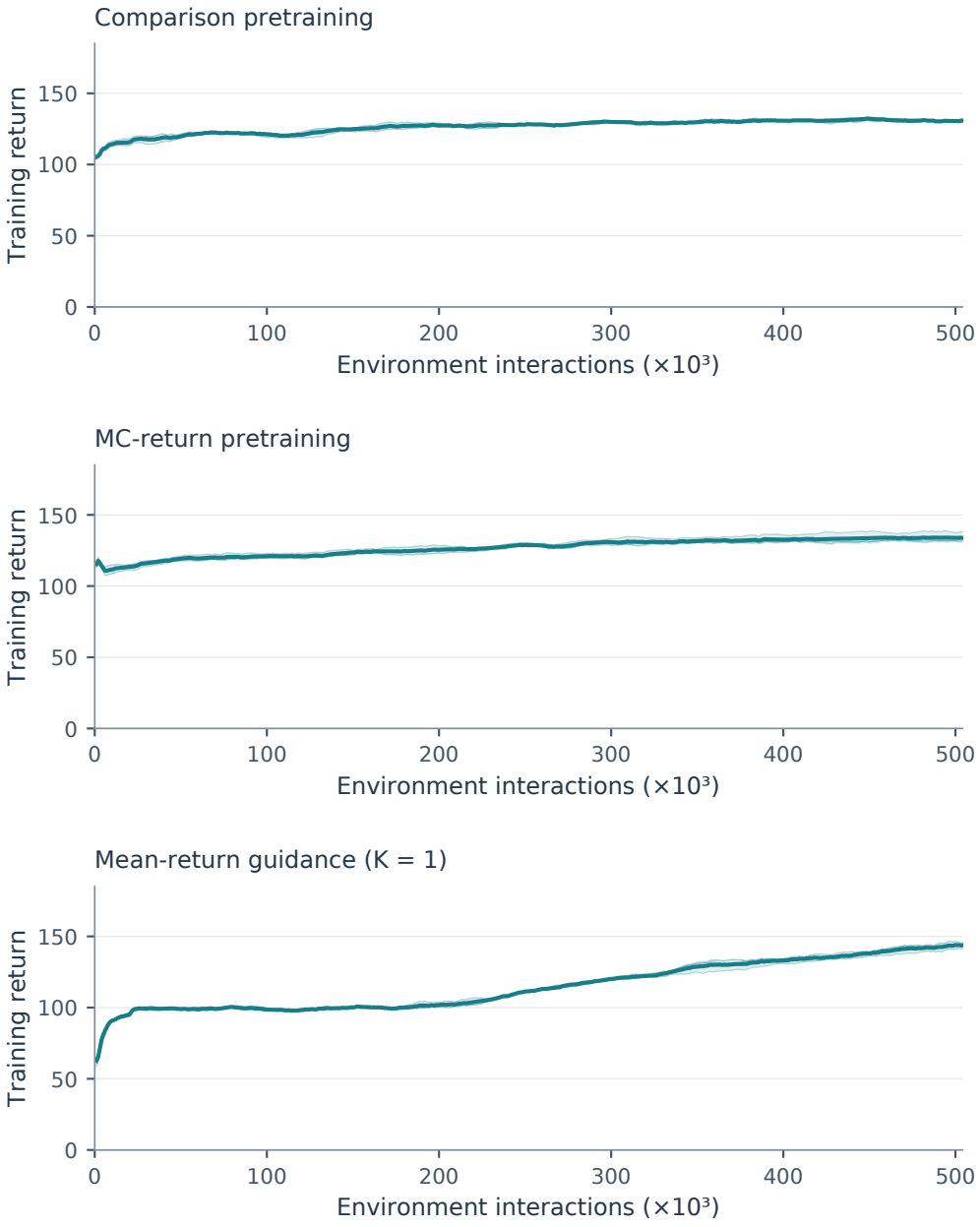


Figure 14: Matched three-seed RL4RS-A controls: individual smoothed trajectories, mean, and sample SD. Reward-injection monitor returns include the injected reward and use a separate vertical scale.

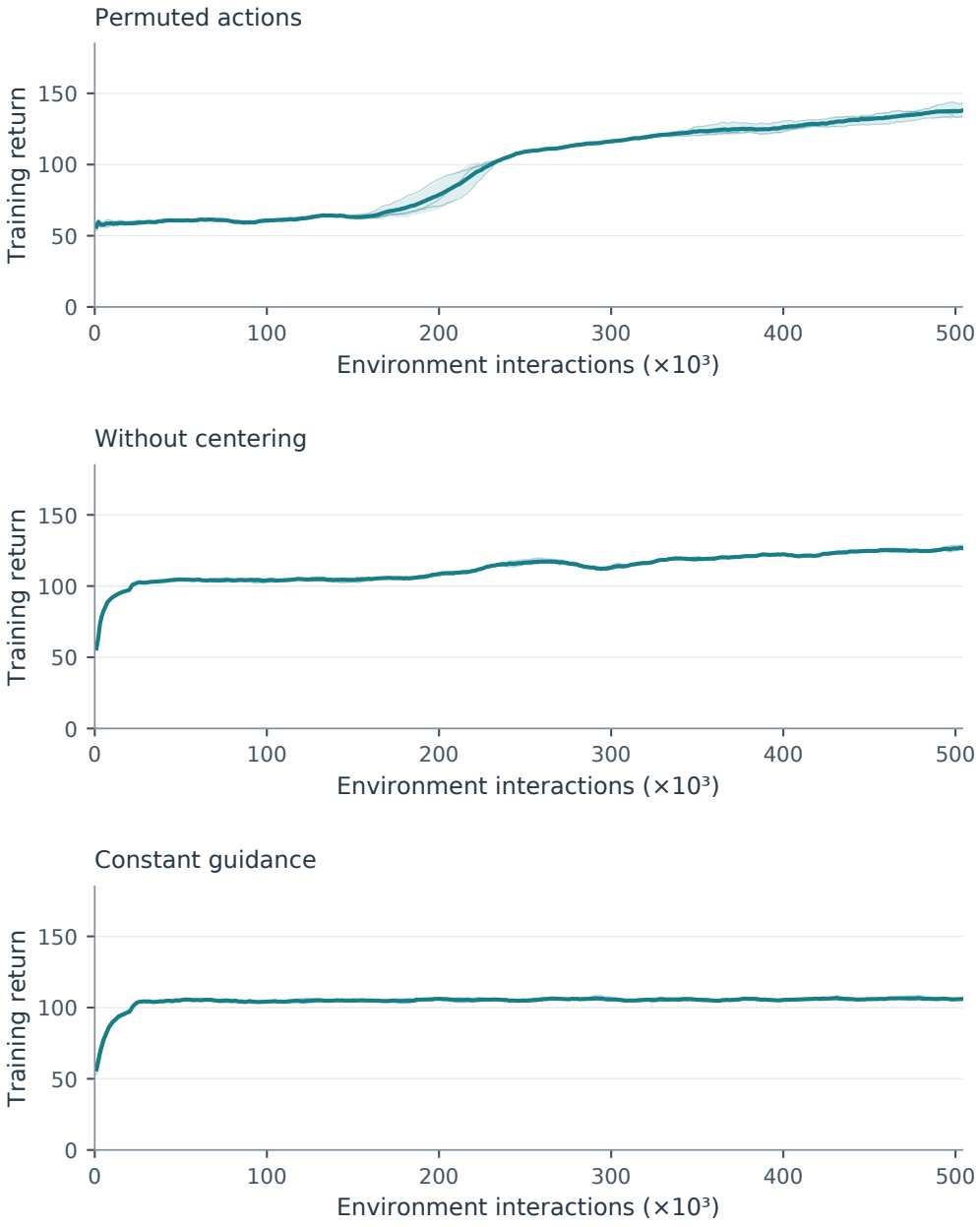


Figure 15: Matched three-seed RL4RS-A controls: individual smoothed trajectories, mean, and sample SD. Reward-injection monitor returns include the injected reward and use a separate vertical scale.

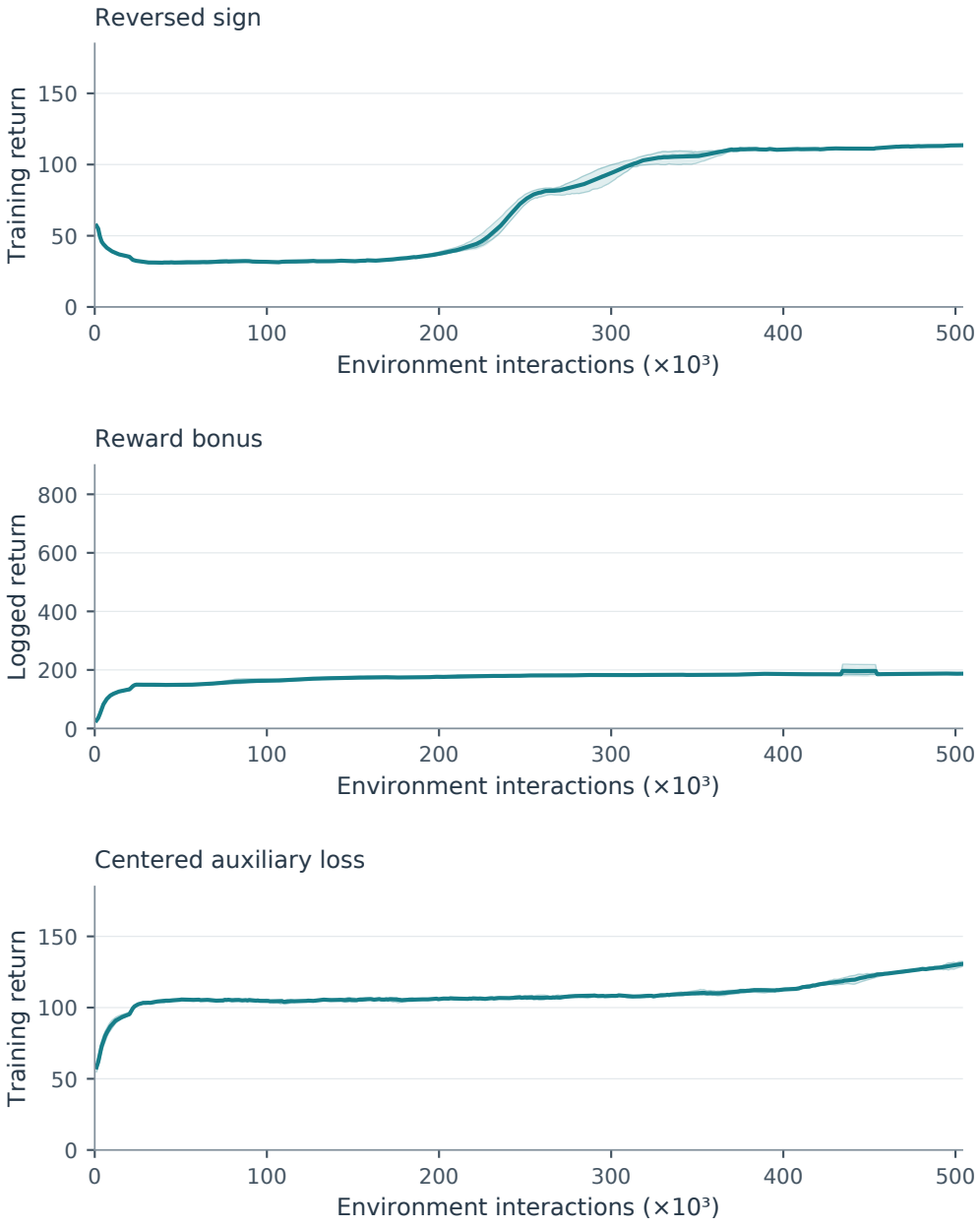


Figure 16: Matched three-seed RL4RS-A controls: individual smoothed trajectories, mean, and sample SD. Reward-injection monitor returns include the injected reward and use a separate vertical scale.