MERA: MATRIX-ENTROPY RERANKING ACTOR-CRITIC FOR SEQUENTIAL RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential reranking is a critical final stage in recommender systems. Given K candidates and N display positions, it makes sequential decisions over $K!/(K-N)!$ discrete ordered slates. Existing methods often optimize proxy scores, fusion weights, or latent controls before decoding them into slates, creating an action-interface mismatch. We propose Matrix-Entropy Reranking Actor-Critic (MERA), which models reranking as a Markov decision process with display-or-discard assignments as actions. Using the item–slot assignment probabilities of ranking policies, we define matrix-entropy-regularized Bellman operators and establish their contraction properties. We characterize the marginal actor problem's first-order stationarity through a KKT decomposition into an inner entropic transport problem and an outer projected score-alignment condition. MERA parameterizes feasible assignment plans using $O(KN)$ scores and solves the inner problem with Sinkhorn scaling. Critic action gradients at hard assignments sampled by Gumbel perturbation and Hungarian rounding guide projected score alignment. On two datasets with three simulators, MERA improves mean total session reward by up to 14.43% over the strongest of seven baselines per setting, with competitive session-level exposure diversity. MERA maintains low inference latency as candidate sets grow, supporting deployment in industrial reranking systems. Our code is available at `https://anonymous.4open.science/r/MERA_code-CD4D/`.

1 INTRODUCTION

Reranking is the last decision layer of a recommender system. In industrial pipelines, upstream retrieval, pre-ranking, and ranking narrow the item pool to hundreds of candidates, from which reranking selects and orders tens of items under tight latency constraints (Ma et al., 2021; Song et al., 2022). Prior work has studied list-wise refinement, slate optimization, and diversity-oriented reordering (Carbonell & Goldstein, 1998; Ai et al., 2018; Pei et al., 2019; Ie et al., 2019b). In sequential recommendation, exposure decisions shape subsequent interactions, affecting click reward, session depth, and diversity across requests (Wang et al., 2022; Zhao et al., 2023; Liu et al., 2024; Xue et al., 2025; Su et al., 2025).

This setting exposes an action-interface mismatch in many recommender systems. Existing methods optimize future engagement or multi-task utility through control variables such as item scores, fusion weights, or item-wise policies (Zheng et al., 2018; Chen et al., 2019; Zhang et al., 2022; Liu et al., 2024). These variables specify how to produce a ranking, while users respond to the resulting ordered slate. Position constraints, competition between candidates, within-slate diversity, and decisions about which items to omit are coupled in the displayed list (Carbonell & Goldstein, 1998; Ribeiro et al., 2012). We represent these coupled decisions directly in the policy's action space.

Resolving this action-interface mismatch presents two challenges. First, selecting an ordered slate of N items from K candidates already yields $K!/(K-N)!$ actions per request, and repeated decisions across a session further compound this complexity through the possible sequences of ranking actions. Second, small perturbations to proxy actions can cause abrupt slate changes after decoding, making exploration and policy improvement difficult to align.

We propose **MERA** (**M**atrix **E**ntropy **R**eranking **A**ctor-Critic), an actor–critic framework that represents ranking actions as item–position assignments. MERA uses a $K \times (N+1)$ score matrix with

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.

N display columns and a discard column for unshown candidates. The actor maps these scores to a feasible item–slot plan, and constrained rounding produces the hard slate executed by the environment. The critic learns the executed slate's long-term value, aligning value learning with user exposure.

To integrate ranking actions and exploration into sequential control, we formulate a matrix-entropy-regularized Markov decision process. Following maximum-entropy RL (Ziebart et al., 2008; Haarnoja et al., 2017; 2018), we augment the discounted return with entrywise entropy of the policy's item–slot marginals and establish contraction of the corresponding Bellman operators. For actor optimization, a KKT characterization separates the marginal-plan objective's first-order conditions into an inner entropic transport problem and an outer projected score-alignment condition. Sinkhorn scaling solves the inner problem, while critic action gradients at sampled hard assignments provide targets for projected score alignment.

MERA improves mean Total reward up to 14.43% over the strongest baseline in every evaluated setting, while retaining competitive session-level exposure diversity. These gains are accompanied by stronger positive user responses and median inference latency below 8 ms. Ablations reveal interactions among stochastic execution, position-aware value learning, and projected actor updates; controlled perturbations show more gradual changes in MERA's executed slates and rewards.

Contributions. (1) **Action representation.** We formulate an MDP over compact item–slot display-or-discard assignments. (2) **Theory-guided optimization.** We establish matrix-entropy Bellman operators and derive a KKT-guided actor update coupling entropic transport with projected score alignment. (3) **Evaluation.** We demonstrate gains in session reward and positive user responses with competitive exposure diversity and millisecond-scale inference. Ablations and perturbation analyses assess exploration, critic design, and actor optimization.

2 Related Work

List-wise reranking and slate recommendation. Reranking revises an initial ranked list under list-level criteria. MMR promotes diversity (Carbonell & Goldstein, 1998), while neural rerankers such as DLCM and PRM model list context and item interactions (Ai et al., 2018; Pei et al., 2019). SetRank learns permutation-invariant list representations (Pang et al., 2020), and counterfactual learning-to-rank addresses biased exposure logs (Agarwal et al., 2019). SlateQ decomposes slate value for tractable optimization (Ie et al., 2019b). These works motivate list-level decisions and spatial diversity (Ribeiro et al., 2012), but usually treat reranking as static refinement or rely on value decompositions.

Long-term recommendation and RL-based user optimization. RL recommenders optimize delayed utility beyond immediate clicks. Representative systems include DRN for news (Zheng et al., 2018), RL-to-rank for search (Hu et al., 2018), page-wise recommendation (Zhao et al., 2018), and top-N off-policy correction (Chen et al., 2019). Later work studies multi-objective utility and long-term engagement (Zou et al., 2019; Zhao et al., 2020), user/session modeling (Pancha et al., 2022; Wang et al., 2022), simulators (Zhao et al., 2023), and immediate–long-term trade-offs (Zhang et al., 2022; Liu et al., 2024). Recent reranking agents further explore user retention and long-horizon reward (Cai et al., 2023; Xue et al., 2025). Their actions are usually scores, fusion weights, latent controls, or item policies. MERA instead controls the final exposure assignment.

Differentiable assignment and structured ranking actions. Structured relaxations make ranking and matching trainable by gradients. Doubly stochastic assignment layers have been used for permutation policies (Emami & Ranka, 2018), and perturbation-based relaxations extend reparameterized learning to latent matchings (Mena et al., 2018). NeuralSort relaxes sorting with continuous surrogates (Grover et al., 2019), and fast differentiable sorting uses permutahedral projections (Blondel et al., 2020). Optimal-transport ranking and SOFT top-N connect differentiable ranking to entropic transport (Cuturi et al., 2019; Xie et al., 2020).

Maximum-entropy reinforcement learning. Maximum-entropy RL defines entropy-regularized control objectives (Ziebart et al., 2008). Soft Q-learning connects this view to energy-based policies (Haarnoja et al., 2017), and soft actor-critic gives a practical off-policy actor–critic algorithm (Haarnoja et al., 2018). In ordered top-N reranking, exact policy entropy lives on a combinatorial action space.

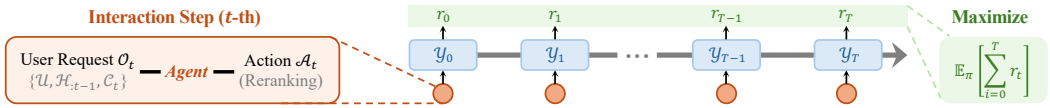


Figure 1: An overview: session-wise item reranking as a sequential interaction.

3 Problem Formulation

We model session-wise reranking as an episodic Markov decision process with a random terminal time T, as illustrated in Figure 1. At interaction step t, the agent receives a request state

$$\mathcal{O}_t = (\mathcal{U}, \mathcal{H}_{:t-1}, \mathcal{C}_t),$$

where $\mathcal{U}$ denotes static user features, $\mathcal{H}_{:t-1}$ denotes the within-session interaction history up to time $t-1$, and $\mathcal{C}_t$ is the candidate shortlist supplied to the assignment stage. We write $K := |\mathcal{C}_t|$ for the shortlist size and assume a fixed number N of exposed positions. At each step, the agent decides a reranking action $X_t \in \mathcal{X}_{K,N}$, whose formal definition is given in Eq. (2).

Objective. Executing X_t generates user feedback $\mathcal{Y}_t$, which updates the interaction history before the next request or session termination. Feedback may contain clicks and other behavioral signals. We map it to a scalar reward $r_t = r(\mathcal{Y}_t)$ and optimize the discounted session return

$$J(\pi) := \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} \gamma^t r_t\right], \qquad \gamma \in (0,1). \tag{1}$$

Equivalently, one may append an absorbing terminal state after the session ends and view Eq. (1) as an infinite-horizon discounted objective with zero reward after termination.

4 Matrix-Entropy Reranking Actor-Critic

MERA combines an assignment representation of ranking actions with matrix-entropy value learning and a constrained actor update. Figure 2 summarizes the actor–critic pipeline, and Algorithm 1 details training and inference. We first describe the action space and the actor–critic interface, then develop the regularized objective and its optimization.

4.1 Structured Action Modeling

MERA represents a reranking action as an item–position assignment on a shortlist of K items, with $K \geq N$. Each item is assigned either to one of the N exposed positions or to a discard slot for items that are not displayed. Let $\mathbf{1}_m$ denote the all-ones vector of length m, and set $\mu := \mathbf{1}_K \in \mathbb{R}^K$ and $\nu := [\mathbf{1}_N^\top,\ K-N]^\top \in \mathbb{R}^{N+1}$. The hard partial-assignment set and item–slot plan set are

$$\begin{aligned} \mathcal{X}_{K,N} &:= \left\{ X \in \{0,1\}^{K\times(N+1)} : X\mathbf{1}_{N+1} = \mu,\ X^\top \mathbf{1}_K = \nu \right\}, \\ \mathcal{T}_{K,N} &:= \left\{ P \in \mathbb{R}_{\geq 0}^{K\times(N+1)} : P\mathbf{1}_{N+1} = \mu,\ P^\top \mathbf{1}_K = \nu \right\}. \end{aligned} \tag{2}$$

For a hard assignment, the row constraints assign each item once, the first N column constraints fill each exposed position once, and the discard column accommodates the remaining $K-N$ items. When $K = N$, the discard column can be omitted. The set $\mathcal{T}_{K,N}$ is the transport polytope induced by these marginals. Because the transport constraint matrix is totally unimodular and the marginals are integral, $\mathcal{T}_{K,N} = \mathrm{conv}(\mathcal{X}_{K,N})$. Thus every feasible plan $P \in \mathcal{T}_{K,N}$ describes the item–slot marginals of at least one distribution over hard assignments. Any $X \in \mathcal{X}_{K,N}$ is a discrete reranking action, with the displayed list read from its first N columns. For a randomized ranking policy, the item–slot marginal $P^\pi \in \mathcal{T}_{K,N}$ records assignment probabilities: $P^\pi_{ij} = \Pr(X_{ij} = 1)$ is the probability of assigning item i to slot j. Changes in P^π move assignment probability among displayed positions and the discard slot.

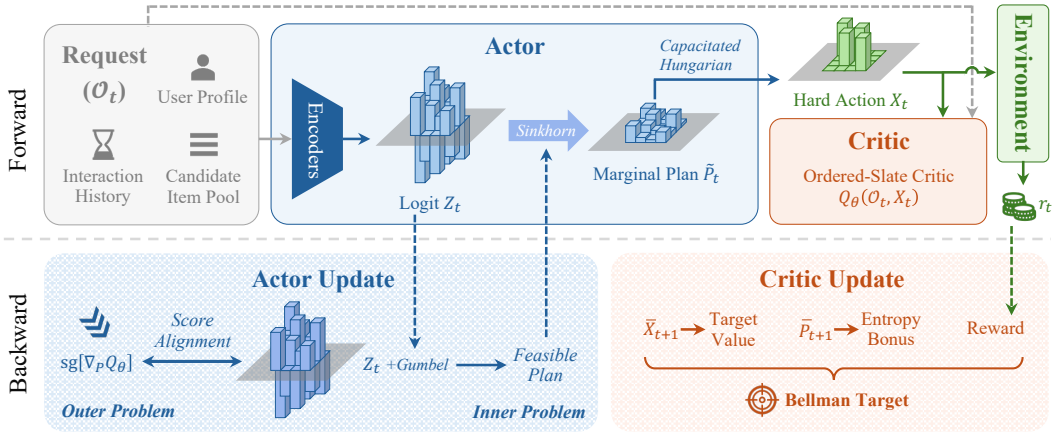


Figure 2: An overview of the MERA framework. The actor maps each request to a feasible item–slot marginal plan and a rounded hard reranking action, while the critic evaluates the executed action with a marginal-plan matrix-entropy bonus.

4.2 Actor and Critic Interfaces

Actor. Given a request $\mathcal{O}_t$, the actor produces item–slot logits and maps them to a feasible plan $P_\phi(\mathcal{O}_t) \in \mathcal{T}_{K,N}$. A hard-action sampler maps this plan to a distribution over executed assignments, $\mathsf{S}(P_\phi(\mathcal{O}_t)) \in \Delta(\mathcal{X}_{K,N}),\ X_t \sim \mathsf{S}(P_\phi(\mathcal{O}_t))$. Here $\Delta(\mathcal{X}_{K,N})$ denotes the distributions over feasible hard assignments. We denote the resulting executed policy by π_ϕ. The actor's parameterized plan P_ϕ guides sampling, while the policy-induced marginal P^{π_ϕ} records the item–slot probabilities induced by sampling; these matrices can differ. Section 4.4 specifies the sampler and update.

Critic. Given the request and a realized assignment, our ordered-slate critic (Ordered-Q) predicts $Q_\theta(\mathcal{O}_t, X_t)$, the regularized long-term value of executing the displayed slate induced by X_t. Ordered-Q reconstructs the displayed item sequence, adds position embeddings to the item representations, and uses self-attention to model item–position and cross-item relationships. The resulting slate representation is combined with the request state to predict its value. Critic learning uses the hard assignments executed in the environment; its differentiable assignment input also provides a local action gradient for actor optimization. Detailed network parameterization is given in Appendix C.1.

4.3 Maximum Matrix-Entropy Reinforcement Learning

Standard reinforcement learning maximizes expected discounted reward. Maximum-entropy RL augments this objective with a causal Shannon-entropy bonus $\mathcal{H}(\pi(\cdot \mid \mathcal{O}_t))$ to encourage stochastic policies (Ziebart et al., 2008; Haarnoja et al., 2017; 2018). For sequential reranking, this objective allows multiple item–position arrangements to remain active across requests. The ordered top-N action space $\mathcal{X}_{K,N}$ has cardinality $|\mathcal{X}_{K,N}| = K!/(K-N)!$, making exact entropy evaluation by enumeration cost $O(K!/(K-N)!)$.

MERA uses entrywise matrix entropy to make regularization tractable in the item–slot representation. For any feasible plan $P \in \mathcal{T}_{K,N}$, write $H_{\mathrm{mat}}(P) := -\sum_{i=1}^{K}\sum_{j=1}^{N+1} P_{ij}\log P_{ij}$, with the standard convention $0\log 0 = 0$. Since each row of P sums to one, $H_{\mathrm{mat}}(P)$ is the sum of row-wise categorical entropies and satisfies $0 \le H_{\mathrm{mat}}(P) \le K\log(N+1)$. The column constraints imply the tighter maximum $K\log K - (K-N)\log(K-N)$ when $K > N$. Thus the entropy bonus is bounded for a fixed shortlist size. For any policy π over hard reranking actions, define its item–slot marginal plan as $P^\pi(\mathcal{O}) := \mathbb{E}_{X\sim\pi(\cdot|\mathcal{O})}[X] \in \mathcal{T}_{K,N}$. Equivalently, let $A_i \in [N+1]$ denote the slot assigned to item i. Since a ranking action is the joint assignment $(A_1, \ldots, A_K)$, Shannon entropy subadditivity gives

$$\mathcal{H}(\pi(\cdot \mid \mathcal{O})) = H(A_1, \ldots, A_K \mid \mathcal{O}) \le \sum_{i=1}^{K} H(A_i \mid \mathcal{O}) = H_{\mathrm{mat}}(P^\pi(\mathcal{O})).$$

This inequality connects the entropy of the ranking policy to the row-wise entropies of its true marginal plan. We use the latter to define the ideal matrix-entropy-regularized return

$$J_\alpha(\pi) = \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} \gamma^t \Big(r(\mathcal{Y}_t) + \alpha H_{\mathrm{mat}}(P^\pi(\mathcal{O}_t))\Big)\right], \tag{3}$$

where $\alpha > 0$ controls the entropy bonus relative to reward; the same coefficient is used in the entropic marginal-plan subproblem below. Finite sessions are represented using an absorbing terminal state with zero reward and zero entropy bonus after termination. The corresponding Bellman equations keep the critic on hard reranking actions. Let $\mathbb{E}_p$ denote expectation over $(\mathcal{O}', \mathcal{Y}) \sim p(\cdot \mid \mathcal{O}, X)$ under the transition and feedback law, and $\mathbb{E}_\pi$ over $X' \sim \pi(\cdot \mid \mathcal{O}')$. For a fixed actor π_ϕ, the matrix-entropy policy-evaluation operator is

$$(\mathcal{T}_\alpha^\phi Q)(\mathcal{O}, X) = \mathbb{E}_p\big[r(\mathcal{Y}) + \gamma\left\{\mathbb{E}_{\pi_\phi} Q(\mathcal{O}', X') + \alpha H_{\mathrm{mat}}(P^{\pi_\phi}(\mathcal{O}'))\right\}\big]. \tag{4}$$

For policy improvement, given a hard-action value function $Q(\mathcal{O}, X)$, the matrix-entropy Bellman optimality operator is defined below, where $\sup_\pi$ ranges over feasible hard-action distributions at $\mathcal{O}'$:

$$(\mathcal{T}_\alpha Q)(\mathcal{O}, X) = \mathbb{E}_p\left[r(\mathcal{Y}) + \gamma \sup_\pi \left\{\mathbb{E}_\pi Q(\mathcal{O}', X') + \alpha H_{\mathrm{mat}}(P^\pi(\mathcal{O}'))\right\}\right]. \tag{5}$$

Thus the critic remains a hard-action value function, while the entropy bonus regularizes the marginal plan induced by the next randomized reranking policy.

Proposition 1 (Policy evaluation contraction)**.** *Assume a finite hard-action space $\mathcal{X}_{K,N}$, bounded rewards, bounded entropy bonuses, and an absorbing terminal state with zero reward and zero entropy bonus. Let Q range over bounded functions on the hard-action domain $(\mathcal{O}, X)$. For any fixed actor π_ϕ, the matrix-entropy policy-evaluation operator $\mathcal{T}_\alpha^\phi$ is a contraction with modulus γ under the sup norm. Since $\gamma < 1$, it admits a unique fixed point Q_α^ϕ.*

Proposition 2 (Regularized optimality contraction)**.** *Assume a finite hard-action space $\mathcal{X}_{K,N}$, bounded rewards, bounded entropy bonuses, and an absorbing terminal state with zero reward and zero entropy bonus. Let Q range over bounded functions on the hard-action domain $(\mathcal{O}, X)$. The matrix-entropy optimality operator $\mathcal{T}_\alpha$ is a contraction with modulus γ under the sup norm. Since $\gamma < 1$, it admits a unique fixed point $Q_\alpha^\star$.*

4.4 Decoupled Optimization

Learning alternates between sampled hard-action value updates and constrained actor updates. We begin with the critic's temporal-difference target. Target networks $(\bar\theta, \bar\phi)$ provide a next action and its value estimate. Let $\bar P_{t+1} := P_{\bar\phi}(\mathcal{O}_{t+1})$ and draw $\bar X_{t+1} \sim \mathsf{S}(\bar P_{t+1})$. We use $\hat q_t = r(\mathcal{Y}_t) + \gamma\Big(Q_{\bar\theta}(\mathcal{O}_{t+1}, \bar X_{t+1}) + \alpha H_{\mathrm{mat}}(\bar P_{t+1})\Big)$. The target critic evaluates the sampled hard assignment, and the entropy term uses the target actor's parameterized plan in place of the true policy marginal in Eq. (4). The critic is fitted to this target by minimizing $\mathcal{L}_Q(\theta) = \mathbb{E}\Big[\big(Q_\theta(\mathcal{O}_t, X_t) - \hat q_t\big)^2\Big]$.

Let $\mathcal{B}$ denote the replay buffer and let $d_{\mathcal{B}}$ denote its empirical request marginal. To derive the actor update, we fix the critic Q_θ and consider the following marginal-plan objective:

$$\max_\phi \ \mathbb{E}_{\mathcal{O}\sim d_{\mathcal{B}}}\Big[\mathbb{E}_{X\sim\mathsf{S}(P_\phi(\mathcal{O}))}\big[Q_\theta(\mathcal{O}, X)\big] + \alpha H_{\mathrm{mat}}(P_\phi(\mathcal{O}))\Big] \quad \text{s.t.} \quad P_\phi(\mathcal{O}) \in \mathcal{T}_{K,N}. \tag{6}$$

To separate assignment feasibility from value-driven policy improvement, we study first-order conditions for a fixed request and critic, treating P as a free variable in $\mathcal{T}_{K,N}$. We introduce a free score matrix $Z \in \mathbb{R}^{K\times(N+1)}$ and the projector $\Pi(M) := H_K M H_{N+1}$, where $H_m := I_m - \frac{1}{m}\mathbf{1}_m\mathbf{1}_m^\top$, onto the transport tangent space $\{\Delta \in \mathbb{R}^{K\times(N+1)} : \Delta\mathbf{1}_{N+1} = 0,\ \Delta^\top\mathbf{1}_K = 0\}$. This space contains changes in the plan that preserve its row and column sums. Fixed marginals make row- and column-wise score shifts unidentifiable, and Π removes these directions from the score-alignment residual.

Theorem 1 (KKT Decoupling of the Marginal Actor Problem)**.** *Fix a request $\mathcal{O}$ and assume the mapping $P \mapsto \mathbb{E}_{X\sim\mathsf{S}(P)}[Q_\theta(\mathcal{O}, X)]$ is continuously differentiable on a neighborhood of*

$\mathrm{relint}(\mathcal{T}_{K,N})$. *If* $P \in \mathrm{relint}(\mathcal{T}_{K,N})$ *with* $P_{ij} > 0$ *for all* (i,j)*, then* P *is a first-order stationary point of* $\mathbb{E}_{X\sim \mathsf{S}(P)}[Q_\theta(\mathcal{O},X)] + \alpha H_{\mathrm{mat}}(P)$ *over* $\mathcal{T}_{K,N}$ *if and only if there exists a score matrix* $Z \in \mathbb{R}^{K\times(N+1)}$ *such that the KKT system is decoupled into the following two subproblems:*

$$\textit{Inner problem:} \quad P = \arg\max_{\bar{P}\in\mathcal{T}_{K,N}} \Big\{ \langle Z, \bar{P}\rangle + \alpha H_{\mathrm{mat}}(\bar{P}) \Big\}, \tag{7}$$

$$\textit{Outer problem:} \quad \Pi\big(Z - \nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O},X)\big]\big) = 0. \tag{8}$$

These first-order conditions separate feasible marginal planning in Eq. (7) from projected score alignment in Eq. (8). The inner problem has the Sinkhorn solution below. The outer condition supplies the residual used to construct the actor update.

Proposition 3 (Inner Sinkhorn Solver)**.** *The optimal solution of Eq.* (7) *is* $P = \mathrm{Sinkhorn}_{\mu,\nu}(Z/\alpha)$*. Writing* $G_Z := \exp(Z/\alpha)$*, the generalized Sinkhorn operator is the limit of the differentiable row–column scaling initialized with any positive* $v^{(0)}$*:*

$$\begin{aligned} u^{(\ell+1)} &= \mu \oslash (G_Z v^{(\ell)}), \quad v^{(\ell+1)} = \nu \oslash (G_Z^\top u^{(\ell+1)}), \\ \mathrm{Sinkhorn}_{\mu,\nu}(Z/\alpha) &= \lim_{\ell\to\infty} \mathrm{Diag}(u^{(\ell)}) G_Z \,\mathrm{Diag}(v^{(\ell)}), \end{aligned} \tag{9}$$

where $\oslash$ *denotes elementwise division.*

Matrix entropy thus serves both as a return regularizer and as the smoothing term of the feasible-plan map. Applying the inner solution to the actor's scores gives the parameterization $P_\phi(\mathcal{O}) = \mathrm{Sinkhorn}_{\mu,\nu}(Z_\phi(\mathcal{O})/\alpha)$, and the outer condition aligns these scores with the value gradient after removing row- and column-wise offsets.

We express this condition as the ideal outer residual loss

$$\mathcal{L}_Z(\phi) = \frac{1}{2}\,\mathbb{E}_{\mathcal{O}\sim d_{\mathcal{B}}}\left[\left\| \Pi\Big(Z_\phi(\mathcal{O}) - \nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O},X)\big]\big|_{P=P_\phi(\mathcal{O})} \Big) \right\|_F^2 \right], \tag{10}$$

where $\|\cdot\|_F$ is the Frobenius norm.

Gumbel–Sinkhorn sampling and score alignment. The outer residual requires the gradient of expected hard-action value with respect to the marginal plan. We construct a local target from the critic's action gradient at a hard assignment sampled using Gumbel perturbations, Sinkhorn scaling (Mena et al., 2018), and Hungarian rounding. Let $Z_\phi(\mathcal{O})$ be the clean item–slot scores, draw entrywise Gumbel noise $\varepsilon_{ij} \overset{\text{i.i.d.}}{\sim} \mathrm{Gumbel}(0,1)$, set $\bar{Z}_\phi(\mathcal{O}) := Z_\phi(\mathcal{O}) + \varepsilon$, and define

$$\widetilde{P}_\phi(\mathcal{O},\varepsilon) := \mathrm{Sinkhorn}_{\mu,\nu}\big(\bar{Z}_\phi(\mathcal{O})/\alpha\big), \qquad \widetilde{X}_\phi(\mathcal{O},\varepsilon) := \mathcal{M}_{\mu,\nu}\Big(\widetilde{P}_\phi(\mathcal{O},\varepsilon)\Big),$$

where $\mathcal{M}_{\mu,\nu}$ denotes Hungarian rounding. The same coefficient α sets the sharpness of the perturbed Sinkhorn plan. The perturbed scores produce a feasible continuous plan, which is rounded to the hard slate used for execution and value evaluation. With the straight-through representation of this hard sample,

$$\widetilde{X}_\phi^{\mathrm{st}} := \mathrm{sg}\Big[\widetilde{X}_\phi - \widetilde{P}_\phi\Big] + \widetilde{P}_\phi, \qquad \nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}[Q_\theta(\mathcal{O},X)]\big|_{P=P_\phi(\mathcal{O})} \approx \nabla_X Q_\theta(\mathcal{O}, \widetilde{X}_\phi^{\mathrm{st}}),$$

where $\mathrm{sg}[\cdot]$ preserves the forward value and blocks gradients through its argument. The critic's action gradient at the sampled hard assignment provides a biased local surrogate for the marginal-space value gradient. MERA detaches this target and minimizes the squared projected residual between the clean scores Z_ϕ and the target. The actor-loss gradient therefore flows only through the clean score branch, as detailed in Appendix C.8. Proofs of the KKT characterization and the Sinkhorn solution are given in Appendix C.7 and Appendix C.6.

5 Experiments

We evaluate MERA on whole-session reranking, where each recommendation action updates the user state and the objective is cumulative click reward. We first compare session return, exposure diversity, and inference cost, then examine the contribution of each component and how changes in the policy's action representation affect the displayed slate. Implementation settings are given in Appendix A.

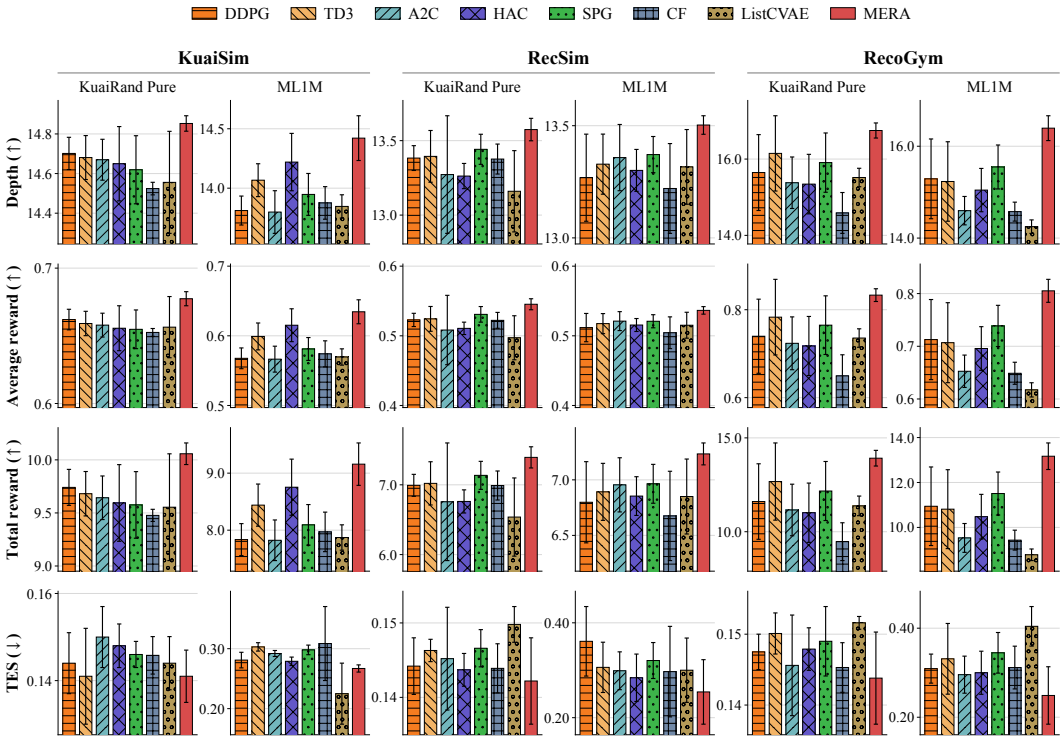


Figure 3: Whole-session performance across three simulators and two datasets. Rows show Depth, Average reward, Total reward, and TES; each simulator spans KuaiRand Pure and ML1M columns. Bars show five-seed means with standard deviations. Y-axis ranges vary by panel.

5.1 Experimental Setup

Simulators and datasets. We evaluate in three simulators including KuaiSim (Zhao et al., 2023), RecSim (Ie et al., 2019a), and RecoGym (Rohde et al., 2018), each on KuaiRand Pure and MovieLens-1M (ML1M) datasets. KuaiRand Pure contains unbiased short-video logs with random exposure. KuaiSim combines learned multi-behavior responses with a leave model; RecSim and RecoGym model evolving satisfaction and latent preferences, respectively. At each request, the agent reranks candidates into a slate of 20 items and optimizes click reward. Dataset statistics and simulator details are in Appendix A.1.

Baselines. RL baselines include DDPG (Lillicrap et al., 2019) and TD3 (Fujimoto et al., 2018), which map continuous hyper-actions to item scores; A2C (Mnih et al., 2016), a policy-gradient method; HAC (Liu et al., 2023), which learns latent actions; and SPG (Emami & Ranka, 2018), which uses a Sinkhorn policy. Request-level baselines include CF (Kang & McAuley, 2018; Zhao et al., 2023) for pointwise scoring and ListCVAE (Jiang et al., 2019) for slate generation via a conditional variational autoencoder. All methods share the same item pool, observation interface, slate length, and click objective within each simulator. RL methods use the same state-encoder architecture and training budget. SPG and MERA form a learned per-request shortlist of 100 items before assignment-based decoding; CF and the hyper-action baselines rank the full pool, while ListCVAE decodes position-specific item distributions. MERA's network parameterization and method-specific training settings are given in Appendix C.1 and Appendix A.4.

Evaluation metrics. We report *Depth*, the average number of requests served before leaving; *Average reward*, the per-request reward; and *Total reward*, the accumulated session reward. We also report *Temporal Exposure Similarity (TES)*, a position-weighted average scaled cosine similarity over all items exposed during a session; lower values indicate less homogeneous, more diverse session-level exposure. We additionally report *Inference latency*, the median policy-inference time per request in milliseconds; lower is better. Metric definitions and timing protocols are in Appendices A.2 and A.6.

Table 1: Component ablation in KuaiSim on KuaiRand Pure and ML1M. A check mark means the corresponding MERA component is enabled; a dash uses the alternative described in the text. Metrics are means over the same five seeds, and lower is better for TES.

	Components			KuaiRand Pure				ML1M			
Method	Stoch.	Ordered-Q	KKT	Depth↑	Avg.↑	Total↑	TES↓	Depth↑	Avg.↑	Total↑	TES↓
T1	–	–	–	14.7566	0.6674	9.8376	0.148	13.7160	0.5974	8.1936	0.289
T2	–	✓	–	14.7266	0.6638	9.7664	0.145	13.7822	0.6007	8.2785	0.283
T3	–	✓	✓	14.7580	0.6666	9.8423	0.143	13.9748	0.6127	8.5424	0.273
T4	–	–	✓	14.7646	0.6666	9.8358	**0.139**	14.0330	0.6184	8.6780	0.275
T5	✓	–	–	14.6934	0.6611	9.7001	0.146	13.6770	0.5814	7.9515	0.280
T6	✓	✓	–	14.8122	0.6721	9.9555	0.145	14.0414	0.6252	8.7784	**0.267**
T7	✓	–	✓	14.8056	0.6701	9.9044	0.141	14.2558	0.6286	8.9617	0.270
MERA	✓	✓	✓	**14.8528**	**0.6774**	**10.0576**	0.141	**14.4212**	**0.6345**	**9.1577**	**0.267**

5.2 Overall Performance

Figure 3 compares four metrics across six simulator–dataset settings. Bars and error bars show means and standard deviations over five seeds; numerical results are in Appendix B.1.

MERA achieves the highest mean Depth, Average reward, and Total reward in all six settings. Against the strongest baseline for each metric, Depth improves by 0.99%–5.44% and Average reward by 2.33%–9.02%. Total reward improves by 3.25%–4.62% in KuaiSim, 3.61%–3.86% in RecSim, and 9.82%–14.43% in RecoGym. Relative gains in Average reward exceed those in Depth in every setting, indicating that higher cumulative returns accompany both longer sessions and more rewarding requests. MERA also ranks among the top two for TES in every setting, with the lowest mean in five, including one tie. Behavioral results in KuaiSim (Appendix B.2) show that MERA leads six of eight comparisons across the two datasets. On KuaiRand Pure, it achieves the highest Comment, Follow, Forward, and Long view rates, with Long view reaching 0.6639 versus the strongest baseline's 0.6551. On ML1M, Like and Star reach 0.6434 and 0.6606, improving by 2.14% and 1.01%, respectively, over their strongest baselines. MERA ranks second on KuaiRand Pure Like; its Hate rate is 0.5425, compared with TD3's lowest rate of 0.5407.

Inference latency and scaling. MERA's structured decoder requires $O(KN)$ work per Sinkhorn iteration; rectangular Hungarian rounding takes $O(KN^2)$ in the worst case. Using the 10 ms reranking latency reported for eBay's deployed visual-search system (Yang et al., 2017) as a practical reference, we benchmark six KuaiSim-trained RL policies on KuaiRand Pure with batch size 1 and 20 displayed items. Median latencies are 5.600 ms for DDPG, 5.687 ms for TD3, 4.214 ms for A2C, 4.212 ms for HAC, 7.918 ms for SPG, and 7.885 ms for MERA. MERA and SPG have similar inference cost in this benchmark.

In a separate deterministic inference sweep, MERA takes 7.782, 6.887, and 7.203 ms at shortlist sizes of 100, 200, and 400, respectively. At 100 candidates, median Sinkhorn and Hungarian times are 1.569 and 0.421 ms. The three observed medians span 6.887–7.782 ms and remain below the 10 ms reference. Timing protocols are detailed in Appendix A.6.

5.3 Ablation Study

In KuaiSim, Table 1 compares three design choices: stochastic execution, ordered-slate value learning, and the KKT-style projected actor update. All variants use Sinkhorn scaling followed by Hungarian rounding. Stoch. adds Gumbel perturbations to the item–slot logits before Sinkhorn scaling, so the resulting hard slates explore alternative assignments. With Stoch. disabled, the policy uses deterministic item–slot logits. With Ordered-Q disabled, the critic is a flat MLP-Q over the state embedding concatenated with the flattened $K \times (N+1)$ assignment matrix. With KKT disabled, the actor uses an entropy-regularized straight-through deterministic policy gradient (DPG) update. Thus T1 combines deterministic Sinkhorn–Hungarian execution with MLP-Q and DPG, while MERA enables all three components.

The stochastic–deterministic contrast depends on the critic and actor update. With MLP-Q and DPG, stochastic execution lowers Total reward by 0.1375/0.2421 on KuaiRand Pure/ML1M (T1 to T5);

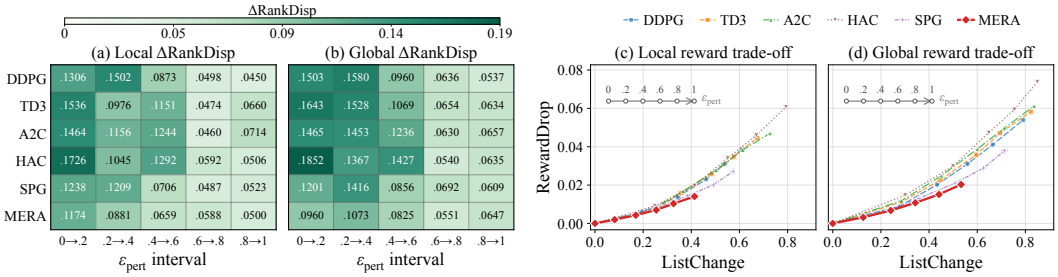


Figure 4: Controlled action perturbation. Panels (a,b) show incremental RankDisp between adjacent perturbation strengths under local and global perturbations. Panels (c,d) plot RewardDrop against ListChange; points move from left to right as perturbation strength increases.

with Ordered-Q and KKT, it raises Total reward by 0.2153/0.6153 (T3 to MERA). MERA achieves the highest mean Depth, Average reward, and Total reward on both datasets.

The components' roles help explain this interaction between exploration and learning from the explored slates. Matrix-entropy regularization encourages the actor to spread marginal mass across item–position assignments, while Gumbel perturbation and rounding expose the critic to alternative feasible lists. Ordered-Q models item–position and cross-item interactions to estimate which of these lists have higher long-term value. The KKT-style update aligns the actor's assignment scores with the critic's local value gradients, while Sinkhorn scaling maintains feasible plans. This connects exploration of complete slates to actor updates based on their estimated values.

5.4 Action-Space Perturbation Analysis

We next examine how changes in a trained policy's action representation translate into changes in the displayed slate. This mechanism-focused study covers the six RL policies shown in Figure 4. In KuaiSim on KuaiRand Pure, we perturb the policy output before decoding. For DDPG, TD3, A2C, and HAC, a denotes the item-scoring weights, with the bias held fixed; for SPG and MERA, a denotes the item-position logits before Sinkhorn scaling and Hungarian rounding. We form $a_\varepsilon = a + \varepsilon_{\text{pert}} d$, where d is the perturbation direction and the reported sweep uses perturbation strengths of 0, 0.2, 0.4, 0.6, 0.8, and 1.0. Local directions perturb one random coordinate by either +1 or -1, while global directions use a dense centered-Gumbel vector normalized to unit root-mean-square magnitude over the active coordinates.

Let $L^0 = (\ell_1^0, \ldots, \ell_N^0)$ and $L^\varepsilon = (\ell_1^\varepsilon, \ldots, \ell_N^\varepsilon)$ denote the clean and perturbed lists. We report item turnover by $\text{ListChange} = 1 - |\{\ell_i^0\}_{i=1}^N \cap \{\ell_i^\varepsilon\}_{i=1}^N|/N$, average normalized displacement by RankDisp with missing clean-list items assigned displacement 1, and RewardDrop as the decrease in Average reward from L^0 to L^ε. Among these six policies, Figure 4(a,b) shows that MERA generally has smaller RankDisp increments at low-to-medium strengths. Figure 4(c,d) places MERA near the lower-left frontier of the RewardDrop–ListChange curves for both local and global perturbations. Together, the panels show that MERA's executed lists and their associated rewards change more gradually under the tested perturbations.

6 Conclusion

MERA models sequential reranking as an MDP over item–position assignments. Matrix-entropy Bellman operators connect exploration with value learning, while a KKT decomposition guides Sinkhorn scaling and projected score updates using critic action gradients at sampled hard assignments. Across six simulator–dataset settings, MERA improves mean total session reward by 3.25%–14.43% over the strongest baseline, with longer sessions, higher per-request reward, and competitive exposure diversity. Ablations reveal interactions among stochastic execution, ordered-slate value learning, and projected updates; controlled perturbations show more gradual changes in rankings and rewards. Median inference latency stays below 8 ms in the reported benchmarks. These results support assignment-based control for sequential reranking.

REFERENCES

Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 5–14, 2019.

Qingyao Ai, Keping Bi, Jiafeng Guo, and W Bruce Croft. Learning a deep listwise context model for ranking refinement. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 135–144, 2018.

Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. Fast differentiable sorting and ranking. In *International conference on machine learning*, pp. 950–959. PMLR, 2020.

Qingpeng Cai, Shuchang Liu, Xueliang Wang, Tianyou Zuo, Wentao Xie, Bin Yang, Dong Zheng, Peng Jiang, and Kun Gai. Reinforcing user retention in a billion scale short video recommender system. In *Companion Proceedings of the ACM Web Conference 2023*, pp. 421–426, 2023.

Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 335–336, 1998.

Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. Top-k off-policy correction for a reinforce recommender system. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pp. 456–464, 2019.

Marco Cuturi, Olivier Teboul, and Jean-Philippe Vert. Differentiable ranking and sorting using optimal transport. *Advances in neural information processing systems*, 32, 2019.

Patrick Emami and Sanjay Ranka. Learning permutations with sinkhorn policy gradient. *arXiv preprint arXiv:1805.07010*, 2018.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. Stochastic optimization of sorting networks via continuous relaxations. *arXiv preprint arXiv:1903.08850*, 2019.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pp. 1352–1361. PMLR, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Yujing Hu, Qing Da, Anxiang Zeng, Yang Yu, and Yinghui Xu. Reinforcement learning to rank in e-commerce search engine: Formalization, analysis, and application. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 368–377, 2018.

Eugene Ie, Chih-wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. RecSim: A configurable simulation platform for recommender systems. *arXiv preprint arXiv:1909.04847*, 2019a.

Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Tushar Chandra, and Craig Boutilier. Slateq: A tractable decomposition for reinforcement learning with recommendation sets. In *IJCAI*, volume 19, pp. 2592–2599, 2019b.

Ray Jiang, Sven Gowal, Yuqiu Qian, Timothy A. Mann, and Danilo J. Rezende. Beyond greedy ranking: Slate optimization via List-CVAE. In *International Conference on Learning Representations*, 2019.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. IEEE, 2018.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

Shuchang Liu, Qingpeng Cai, Bowen Sun, Yuhao Wang, Ji Jiang, Dong Zheng, Peng Jiang, Kun Gai, Xiangyu Zhao, and Yongfeng Zhang. Exploration and regularization of the latent action space in recommendation. In *Proceedings of the ACM Web Conference 2023*, pp. 833–844, 2023.

Ziru Liu, Shuchang Liu, Zijian Zhang, Qingpeng Cai, Xiangyu Zhao, Kesen Zhao, Lantao Hu, Peng Jiang, and Kun Gai. Sequential recommendation for optimizing both immediate feedback and long-term retention. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1872–1882, 2024.

Xu Ma, Pengjie Wang, Hui Zhao, Shaoguo Liu, Chuhan Zhao, Wei Lin, Kuang-Chih Lee, Jian Xu, and Bo Zheng. Towards a better tradeoff between effectiveness and efficiency in pre-ranking: A learnable feature selection based approach. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, pp. 2036–2040, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380379.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. *arXiv preprint arXiv:1802.08665*, 2018.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Nikil Pancha, Andrew Zhai, Jure Leskovec, and Charles Rosenberg. Pinnerformer: Sequence modeling for user representation at pinterest. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 3702–3712, 2022.

Liang Pang, Jun Xu, Qingyao Ai, Yanyan Lan, Xueqi Cheng, and Jirong Wen. Setrank: Learning a permutation-invariant ranking model for information retrieval. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 499–508, 2020.

Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, et al. Personalized re-ranking for recommendation. In *Proceedings of the 13th ACM conference on recommender systems*, pp. 3–11, 2019.

Marco Tulio Ribeiro, Anisio Lacerda, Adriano Veloso, and Nivio Ziviani. Pareto-efficient hybridization for multi-objective recommender systems. In *Proceedings of the sixth ACM conference on Recommender systems*, pp. 19–26, 2012.

David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. RecoGym: A reinforcement learning environment for the problem of product recommendation in online advertising. *arXiv preprint arXiv:1808.00720*, 2018.

Jinbo Song, Ruoran Huang, Xinyang Wang, Wei Huang, Qian Yu, Mingming Chen, Yafei Yao, Chaosheng Fan, Changping Peng, Zhangang Lin, Jinghe Hu, and Jingping Shao. Rethinking large-scale pre-ranking system: Entire-chain cross-domain models. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, pp. 4495–4499, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392365.

Zhiyuan Su, Sunhao Dai, and Xiao Zhang. Revisiting clustering of neural bandits: Selective reinitialization for mitigating loss of plasticity. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 2690–2701, 2025.

Yuyan Wang, Mohit Sharma, Can Xu, Sriraj Badam, Qian Sun, Lee Richardson, Lisa Chung, Ed H Chi, and Minmin Chen. Surrogate for long-term user experience in recommender systems. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 4100–4109, 2022.

Yujia Xie, Hanjun Dai, Minshuo Chen, Bo Dai, Tuo Zhao, Hongyuan Zha, Wei Wei, and Tomas Pfister. Differentiable top-k with optimal transport. *Advances in neural information processing systems*, 33:20520–20531, 2020.

Zhenghai Xue, Qingpeng Cai, Bin Yang, Lantao Hu, Peng Jiang, Kun Gai, and Bo An. Auro: Reinforcement learning for adaptive user retention optimization in recommender systems. In *Proceedings of the ACM on Web Conference 2025*, pp. 391–401, 2025.

Fan Yang, Ajinkya Kale, Yury Bubnov, Leon Stein, Qiaosong Wang, Hadi Kiapour, and Robinson Piramuthu. Visual search at eBay. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2017. doi: 10.1145/3097983.3098162.

Qihua Zhang, Junning Liu, Yuzhuo Dai, Yiyan Qi, Yifan Yuan, Kunlun Zheng, Fan Huang, and Xianfeng Tan. Multi-task fusion via reinforcement learning for long-term user satisfaction in recommender systems. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 4510–4520, 2022.

Kesen Zhao, Shuchang Liu, Qingpeng Cai, Xiangyu Zhao, Ziru Liu, Dong Zheng, Peng Jiang, and Kun Gai. Kuaisim: A comprehensive simulator for recommender systems. *Advances in Neural Information Processing Systems*, 36:44880–44897, 2023.

Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM conference on recommender systems*, pp. 95–103, 2018.

Xiangyu Zhao, Xudong Zheng, Xiwang Yang, Xiaobing Liu, and Jiliang Tang. Jointly learning to recommend and advertise. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3319–3327, 2020.

Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. Drn: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 world wide web conference*, pp. 167–176, 2018.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

Lixin Zou, Long Xia, Zhuoye Ding, Jiaxing Song, Weidong Liu, and Dawei Yin. Reinforcement learning to optimize long-term user engagement in recommender systems. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2810–2818, 2019.

A Appendix: Experimental Details

This appendix lists the implementation details used for the experiments in Section 5.

A.1 Simulator and Data Processing

Table 2: Raw public dataset statistics. Density is interactions divided by users times items.

Dataset	# Users	# Items	# Interactions	Density
KuaiRand Pure	27,285	7,551	1,436,609	0.70%
ML1M	6,400	3,706	1,000,208	4.22%

Table 2 reports raw public statistics; the following counts describe the processed rollout data used by the simulator.

After simulator-specific preprocessing, the KuaiRand Pure[1] rollout data contain 19,574 users, 5,659 items, and 1,341,250 interaction records, split into 1,124,729/89,160/89,160 simulator training, validation, and test records. User features are 13 one-hot categorical fields with total dimension about 74; item features include video type, music type, upload type, and tag with total dimension about 70. The behavioral signals reported in KuaiSim are click, long view, like, comment, forward, follow, and hate.

After the same preprocessing, the ML1M[2] rollout data used by the simulator contain 6,040 users, 3,706 items, and 2,000,418 interactions. User features contain gender and age with total dimension 9; item features are movie genres with dimension 18. The behavioral signals reported in KuaiSim are click, like, and star.

KuaiSim. Both datasets use the same learned immediate-response model family in KuaiSim (Zhao et al., 2023). The response model is trained with binary cross-entropy, learning rate 10^{-4}, ℓ_2 coefficient 0.01, batch size 128, maximum history length 100, and 10 epochs. Its Transformer history encoder uses 2 layers, 4 heads, feed-forward dimension 64, and dropout 0.1. The user and item latent dimensions are both 32; the response-model encoding dimension is 64 on KuaiRand Pure and 16 on ML1M.

RecSim. RecSim (Ie et al., 2019a) supports configurable user responses and latent-state dynamics. Our whole-session adaptation models evolving satisfaction and exposure memory. Click probabilities combine user–item affinity, satisfaction, and position effects, and the latent state is updated after each request.

RecoGym. RecoGym (Rohde et al., 2018) models browsing and recommendation feedback through evolving latent preferences. Our adaptation generates clicks from latent user–item affinity, item bias, and position effects. User interests evolve through their previous state, clicked items, and stochastic drift.

Both adaptations use each dataset's profiles, histories, and candidate features. Their simulator-specific item attributes and factors are derived from pretrained KuaiSim item encodings. They retain the whole-session history updates and temper-based termination while replacing the response dynamics.

A.2 Whole-Session Environment

The three simulator settings use a common GPU whole-session interface. The slate size is $N = 20$, the maximum number of requests per episode is 20, the initial temper is 20, and the episode batch size is 32. The optimized reward corresponds to the first feedback dimension, i.e., click reward on both datasets. We analyze other behavior rates in KuaiSim. For fairness, all methods receive the same fixed candidate pool at every request. The candidate set is the full item pool loaded by the simulator. The hyper-action baselines score the full pool and directly select the top-N items, while SPG and

[1] https://kuairand.com/

[2] https://grouplens.org/datasets/movielens/1m/

MERA first form a learned per-request top-K shortlist from the same full pool and then perform assignment-based slate decoding.

The intra-slate metric is EILD. For a slate $R = (i_1, \ldots, i_N)$, let s_{kl} denote the item similarity between positions k and l, and define $w_{kl} = 0$ if $k = l$ and $w_{kl} = 1/\log_2(|k-l|+1)$ otherwise. The simulator uses

$$\mathrm{EILD@}N(R) = 1 - \frac{1}{N}\sum_{k=1}^{N}\frac{\sum_{l=1}^{N} w_{kl}s_{kl}}{\sum_{l=1}^{N} w_{kl}},$$

so high EILD indicates low similarity among nearby exposed positions. In KuaiSim, the simulator item-correlation coefficient is $\rho_{\mathrm{sim}} = 0.4$ for KuaiRand Pure and $\rho_{\mathrm{sim}} = 0.3$ for ML1M. During environment feedback sampling, the simulator subtracts the EILD-based similarity penalty from each point-wise response probability:

$$p_{t,n,b} = \max\{0,\ \sigma(\ell_{t,n,b}) - \rho_{\mathrm{sim}}\, c_{t,n}\},$$

where $\ell_{t,n,b}$ is the immediate-response logit for behavior b at position n, and $c_{t,n}$ is the position-aware intra-slate similarity of that item to the slate. The user temper update is clipped to $[-2, 0]$, and a user leaves when the current temper is below 1. The RecSim and RecoGym adaptations use the same termination rule with their own response dynamics and no intra-slate similarity penalty.

TES measures similarity across the full exposure sequence of a completed session. For session s, let $S_s = (i_{s,1}, \ldots, i_{s,L_s})$ concatenate the displayed slates in chronological order, with $L_s = T_s N$; repeated items are retained as separate exposures. Let $x_i \in \mathbb{R}^d$ be the simulator's item embedding, $z_i = x_i/\|x_i\|_2$, and $\mathrm{sim}(i,j) = z_i^\top z_j/d$. We apply the same distance weights as EILD to positions in this concatenated sequence: $w_{kk} = 0$ and $w_{kl} = 1/\log_2(|k-l|+1)$ for $k \neq l$.

$$\mathrm{TES}(S_s) = \frac{10}{L_s}\sum_{k=1}^{L_s}\frac{\sum_{l=1}^{L_s} w_{kl}\,\mathrm{sim}(i_{s,k}, i_{s,l})}{\sum_{l=1}^{L_s} w_{kl}}.$$

Thus each exposure's similarity to other exposures is normalized by its total weight before averaging over the session. The factor 10 sets the reporting scale, and lower TES indicates less homogeneous exposure. EILD is used within each slate in the simulator's response model, whereas TES includes both within-request and cross-request exposure pairs.

A.3 RL Training Protocol

Main results in all six settings average five random seeds. Policy training and reward evaluation were run on one NVIDIA RTX 5090 GPU with 32 GB memory and 25 vCPUs on an Intel Xeon Platinum 8470Q CPU. The actor–critic methods use replay buffer size 100,000, training iterations 20,000, RL batch size 128, discount factor $\gamma = 0.9$, one update per environment step, policy training starting at step 100, initial and final $\epsilon = 0.01$, elbow $\epsilon = 0.1$, exploration rate 1.0, checkpoint interval 200, and target-network soft update coefficient 0.01. Actor and critic weight decay are both 10^{-5}.

All methods use the same Transformer-based state-encoder architecture. It embeds user id, user features, item id, item features, history item encodings, feedback, and position, then applies a Transformer encoder over the history sequence. The Transformer has 3 layers, 4 heads, feed-forward dimension 64, and dropout 0.1. The state user and item latent dimensions are both 16. The policy encoder dimension is 32 on KuaiRand Pure and 16 on ML1M; the resulting state dimension is 96 on KuaiRand Pure and 48 on ML1M.

A.4 Method-Specific Settings

When batches are padded to a common shortlist size, padded rows are masked out of the exposed columns and forced into the discard column, using the same transport constraints on the valid rows.

CF follows the pointwise collaborative-filtering baseline in KuaiSim (Kang & McAuley, 2018; Zhao et al., 2023). It scores candidates by the inner product between a transformed user state and item embeddings, optimizes a pointwise loss on immediate item feedback, and selects the top-scoring items at evaluation.

ListCVAE (Jiang et al., 2019) learns a reward-conditioned distribution over complete slates using a conditional variational autoencoder. Training combines slate reconstruction with KL regularization

toward a conditional prior. At evaluation, it decodes the prior mean under a target reward and selects items across positions while masking previously selected items. Both CF and ListCVAE collect feedback through online interaction and optimize their request-level objectives on replay samples.

DDPG, TD3, A2C, and HAC use a shared hyper-action actor with dot-product item scoring. The actor MLP has hidden dimensions $[256, 64]$ and outputs a hyper-action whose dimension is the policy encoder dimension plus one. DDPG, TD3, and HAC use an MLP Q-critic over the concatenated state and hyper-action with hidden dimensions $[256, 64]$ and dropout 0.1; A2C instead uses an MLP state-value critic and updates the actor with a TD advantage estimate. The policy-noise variance is 0.1 and the noise clip is 1.0. In KuaiSim, baseline actor and critic learning rates on KuaiRand Pure are 10^{-4} and 10^{-3}; on ML1M they are 10^{-5} and 10^{-3}. TD3 uses twin critics. HAC adds an inverse module that maps the realized slate back to the hyper-action space and optimizes an additional behavior loss.

SPG and MERA both use assignment shortlist size $K = 100$, Sinkhorn iterations 20, actor learning rate 10^{-4}, critic learning rate 10^{-3}, and actor MLP hidden dimensions $[256, 64]$. This K is a learned per-request top-K shortlist formed from the full item pool before Sinkhorn or Hungarian decoding, not an externally fixed candidate set. SPG uses a Sinkhorn partial-assignment actor, entropy coefficient 0.01. MERA additionally applies Gumbel perturbation, Hungarian rounding, a KKT-style actor update using detached critic action-gradient targets, and an ordered-slate critic. Ordered-Q reconstructs the displayed slate from the hard action matrix, adds position embeddings, applies a 2-layer Transformer with 4 heads and feed-forward dimension 64, pools the ordered slate representation, concatenates it with the state, and predicts Q with an MLP of hidden dimensions $[256, 64]$.

KuaiRand Pure uses Sinkhorn temperature 0.7 for SPG and MERA uses matrix-entropy coefficient $\alpha = 0.7$. ML1M uses Sinkhorn temperature 0.2 for SPG and MERA uses matrix-entropy coefficient $\alpha = 0.2$. In MERA, the same α is used for the clean marginal map, the matrix-entropy bonus, and the Gumbel-perturbed Sinkhorn execution path; no separate Gumbel temperature is used. Each training run is evaluated from its saved checkpoint with a no-exploration simulator validation rollout of 1000 steps, logging reward and TES every 100 steps.

A.5 MERA Training and Inference Procedure

Algorithm 1 summarizes the full pipeline used by MERA. Training alternates between simulator interaction and off-policy updates. At interaction time, the actor maps the current request to an item–slot score matrix, maps it to a marginal plan with entropic Sinkhorn scaling, perturbs the scores with Gumbel noise, and rounds the perturbed transport plan to a feasible hard slate by Hungarian matching. Hungarian rounding handles the discard capacity through either replicated discard columns or an equivalent rectangular assignment. The replay buffer stores the executed hard action, because both the simulator feedback and the critic target are defined on the concrete slate shown to the user.

The critic update is a standard bootstrapped hard-action update with an additional matrix-entropy bonus on the next marginal plan. The actor update is different from a vanilla deterministic policy gradient: MERA draws a fresh Gumbel perturbation, samples a hard assignment through Sinkhorn scaling and Hungarian rounding, uses the critic's action gradient as a detached target, and minimizes the projected score residual in Eq. (24). At inference time, the same actor and transport map are used, but the rollout can either disable Gumbel perturbation for deterministic validation or keep it when stochastic exposure is desired.

A.6 Inference Latency Protocol

We benchmark KuaiSim-trained policies on KuaiRand Pure with batch size 1 and 20 displayed items. Each configuration repeatedly processes a fixed input batch with an empty interaction history. Timing covers the policy forward pass, including state and item encoding, full-pool scoring, slate construction, and associated data transfers. Environment stepping and the user-response simulator are outside the timing boundary. CUDA is synchronized before and after each timed call.

The six-method comparison runs sequentially in one process on the same NVIDIA RTX 4090. Each configuration uses 30 warm-up calls followed by three rounds of 200 timed calls; reported medians

Algorithm 1 MERA training and inference

Require: Simulator environment $\mathcal{E}$, replay buffer $\mathcal{B}$, slate size N, shortlist size K, matrix-entropy coefficient α, discount γ

Initialize clean actor score network Z_ϕ, hard-action critic Q_θ, target networks $Z_{\bar{\phi}}, Q_{\bar{\theta}}$, and buffer $\mathcal{B}$
procedure TRAINING
  for each environment step **do**
    Observe request $\mathcal{O}_t$ and candidate set $\mathcal{C}_t$ from $\mathcal{E}$
    Compute clean scores $Z_t \leftarrow Z_\phi(\mathcal{O}_t)$ and marginal plan $P_t \leftarrow \mathrm{Sinkhorn}_{\mu,\nu}(Z_t/\alpha)$
    Sample Gumbel noise ε_t, set $\bar{Z}_t \leftarrow Z_t + \varepsilon_t$, and build $\widetilde{P}_t \leftarrow \mathrm{Sinkhorn}_{\mu,\nu}(\bar{Z}_t/\alpha)$
    Round to a hard assignment $X_t \leftarrow \mathcal{M}_{\mu,\nu}(\widetilde{P}_t)$ and display the non-discard columns as the ordered slate
    Execute X_t in $\mathcal{E}$, observe reward r_t, next request $\mathcal{O}_{t+1}$, and terminal flag d_t
    Store $(\mathcal{O}_t, \mathcal{C}_t, X_t, r_t, \mathcal{O}_{t+1}, d_t)$ in $\mathcal{B}$
    for each gradient update **do**
      Sample a mini-batch from $\mathcal{B}$
      Build next target plan $\bar{P}' \leftarrow \mathrm{Sinkhorn}_{\mu,\nu}(Z_{\bar{\phi}}(\mathcal{O}')/\alpha)$ and sample hard action $\bar{X}'$
      Set $y \leftarrow r + \gamma(1-d)\big(Q_{\bar{\theta}}(\mathcal{O}', \bar{X}') + \alpha H_{\mathrm{mat}}(\bar{P}')\big)$
      Update critic by minimizing $\big(Q_\theta(\mathcal{O}, X) - y\big)^2$
      Resample Gumbel noise, build $\widetilde{X}^{\mathrm{st}}_\phi$, and form $\widetilde{R}_\theta$ using the detached critic-gradient target
      Update actor by minimizing $\widehat{\mathcal{L}}_Z(\phi) = \frac{1}{2}\|\widetilde{R}_\theta\|_F^2$
      Soft-update target networks $Z_{\bar{\phi}}$ and $Q_{\bar{\theta}}$
    end for
  end for
end procedure
procedure INFERENCE(request $\mathcal{O}$, candidates $\mathcal{C}$)
  Compute clean scores $Z \leftarrow Z_\phi(\mathcal{O})$ and $P \leftarrow \mathrm{Sinkhorn}_{\mu,\nu}(Z/\alpha)$
  if stochastic evaluation is enabled **then**
    Sample fresh Gumbel noise ε, set $\bar{Z} \leftarrow Z + \varepsilon$, and set $P_{\mathrm{exec}} \leftarrow \mathrm{Sinkhorn}_{\mu,\nu}(\bar{Z}/\alpha)$
  else
    Set $P_{\mathrm{exec}} \leftarrow P$
  end if
  Return the ordered slate induced by $X \leftarrow \mathcal{M}_{\mu,\nu}(P_{\mathrm{exec}})$ over the non-discard columns
end procedure

pool all 600 measurements. Checkpoints use training seed 11, except SPG, which uses seed 19. MERA retains Gumbel sampling during this comparison.

The separate deterministic MERA sweep uses another RTX 4090, 20 Sinkhorn iterations, rectangular Hungarian rounding, and disabled Gumbel sampling. It reuses the seed-11 checkpoint trained with 100 candidates at inference shortlist sizes of 100, 200, and 400. After 30 warm-up calls per configuration, the 100-candidate setting uses three rounds of 200 timed calls, while the 200- and 400-candidate settings each use one round of 200. Sinkhorn time is measured with CUDA events and Hungarian time with a wall-clock timer; total policy latency includes synchronization and data transfers.

B Appendix: Additional Experimental Results

B.1 Main Metric Summary

Tables 3, 4, and 5 report means and standard deviations for Figure 3. Higher is better for Depth and reward; lower is better for TES. Bold and underlining mark the best and second-best unrounded means in each setting, including ties.

Table 3: Whole-session results in KuaiSim over five seeds, reported as mean ± standard deviation.

	KuaiRand Pure				ML1M			
Method	Depth↑	Avg. reward↑	Total reward↑	TES↓	Depth↑	Avg. reward↑	Total reward↑	TES↓
DDPG	14.702±0.08	0.662±0.01	9.741±0.17	0.144±0.01	13.812±0.12	0.568±0.01	7.835±0.28	0.281±0.01
TD3	14.681±0.11	0.659±0.01	9.681±0.21	**0.141**±0.01	14.067±0.14	0.599±0.02	8.439±0.37	0.303±0.01
A2C	14.670±0.10	0.658±0.01	9.643±0.21	0.150±0.01	13.799±0.18	0.567±0.02	7.823±0.36	0.292±0.01
HAC	14.650±0.19	0.656±0.02	9.594±0.36	0.148±0.01	14.220±0.24	0.615±0.02	8.753±0.49	0.279±0.01
SPG	14.619±0.17	0.655±0.01	9.575±0.31	0.146±0.00	13.947±0.18	0.581±0.02	8.093±0.36	0.298±0.01
CF	14.524±0.03	0.653±0.00	9.477±0.06	0.146±0.00	13.876±0.14	0.574±0.02	7.970±0.35	0.309±0.06
ListCVAE	14.556±0.26	0.656±0.02	9.554±0.50	0.144±0.01	13.846±0.10	0.570±0.01	7.871±0.22	**0.225**±0.05
MERA	**14.853**±0.04	**0.677**±0.01	**10.058**±0.10	**0.141**±0.01	**14.421**±0.19	**0.635**±0.02	**9.158**±0.37	0.267±0.01

Table 4: Whole-session results in RecSim over five seeds, reported as mean ± standard deviation.

	KuaiRand Pure				ML1M			
Method	Depth↑	Avg. reward↑	Total reward↑	TES↓	Depth↑	Avg. reward↑	Total reward↑	TES↓
DDPG	13.383±0.08	0.523±0.01	6.993±0.16	0.144±0.00	13.268±0.19	0.512±0.02	6.798±0.37	0.361±0.07
TD3	13.394±0.17	0.524±0.02	7.021±0.31	0.146±0.00	13.329±0.13	0.518±0.01	6.892±0.26	0.306±0.05
A2C	13.272±0.40	0.508±0.05	6.757±0.84	0.145±0.01	13.358±0.15	0.521±0.01	6.955±0.24	0.299±0.04
HAC	13.262±0.08	0.511±0.01	6.762±0.17	0.144±0.00	13.301±0.09	0.516±0.01	6.854±0.17	0.284±0.05
SPG	13.442±0.10	0.531±0.01	7.136±0.20	0.147±0.00	13.371±0.08	0.521±0.01	6.965±0.18	0.321±0.04
CF	13.376±0.10	0.522±0.01	6.990±0.21	0.144±0.00	13.220±0.20	0.504±0.02	6.675±0.40	0.297±0.10
ListCVAE	13.160±0.27	0.497±0.03	6.538±0.56	0.150±0.00	13.316±0.17	0.515±0.02	6.850±0.34	0.300±0.07
MERA	**13.574**±0.08	**0.545**±0.01	**7.393**±0.15	**0.142**±0.01	**13.503**±0.04	**0.537**±0.01	**7.234**±0.10	**0.254**±0.07

Table 5: Whole-session results in RecoGym over five seeds, reported as mean ± standard deviation.

	KuaiRand Pure				ML1M			
Method	Depth↑	Avg. reward↑	Total reward↑	TES↓	Depth↑	Avg. reward↑	Total reward↑	TES↓
DDPG	15.647±0.99	0.740±0.08	11.607±2.01	0.148±0.00	15.290±0.87	0.713±0.08	10.942±1.75	0.309±0.03
TD3	16.149±0.99	0.783±0.09	12.673±2.06	0.150±0.00	15.232±0.87	0.707±0.08	10.813±1.76	0.331±0.08
A2C	15.380±0.67	0.724±0.06	11.158±1.37	0.146±0.01	14.597±0.31	0.653±0.03	9.532±0.64	0.296±0.04
HAC	15.344±0.78	0.718±0.07	11.012±1.58	0.148±0.00	15.045±0.47	0.696±0.04	10.477±0.99	0.300±0.05
SPG	15.907±0.78	0.765±0.07	12.169±1.57	0.149±0.00	15.548±0.48	0.739±0.04	11.508±0.96	0.345±0.05
CF	14.588±0.54	0.650±0.05	9.470±1.01	0.145±0.00	14.576±0.21	0.649±0.02	9.434±0.44	0.312±0.05
ListCVAE	15.517±0.24	0.736±0.02	11.376±0.52	0.152±0.00	14.249±0.15	0.617±0.01	8.782±0.26	0.404±0.04
MERA	**16.749**±0.20	**0.833**±0.01	**13.918**±0.42	**0.144**±0.01	**16.394**±0.27	**0.805**±0.02	**13.169**±0.59	**0.249**±0.06

B.2 Behavioral Results

Table 6 summarizes behavioral results in KuaiSim, and Table 7 reports its component ablation. Values are five-seed means; higher is better except for Hate. Bold and underlining mark the best and second-best means within each dataset, including ties.

B.3 Sensitivity Analysis

Using KuaiSim on KuaiRand Pure, we analyze MERA's sensitivity to two groups of parameters in Figure 5: model parameters, including the matrix-entropy coefficient α and the actor/critic learning rates, and the simulator environment parameter ρ_{sim}. As described in Appendix A.2, ρ_{sim} is not a tunable MERA parameter; it controls the strength of the intra-slate similarity penalty in the response model and therefore changes the diversity pressure imposed by the environment.

MERA remains stable across the tested model-parameter ranges: performance changes smoothly around the default $\alpha = 0.7$, actor learning rate 10^{-4}, and critic learning rate 10^{-3}, while extreme learning rates produce the expected degradation. Varying ρ_{sim} changes the simulator rather than the policy architecture. As ρ_{sim} increases, the environment applies a stronger similarity penalty, making

Table 6: Behavioral results in KuaiSim.

	KuaiRand Pure						ML1M	
Method	Comment↑	Follow↑	Forward↑	Hate↓	Like↑	Long view↑	Like↑	Star↑
DDPG	0.6117	0.5737	0.5583	0.5408	0.5964	0.6551	0.5863	0.6103
TD3	0.6101	0.5727	0.5563	**0.5407**	0.5955	0.6522	0.6150	0.6372
A2C	0.6099	0.5724	0.5579	0.5430	0.5976	0.6508	0.5896	0.6155
HAC	0.6125	0.5758	0.5640	0.5494	**0.6030**	0.6486	0.6299	0.6540
SPG	0.6084	0.5724	0.5576	0.5423	0.5974	0.6469	0.6019	0.6274
CF	0.6105	0.5685	0.5527	0.5499	0.5991	0.6475	0.5882	0.6085
ListCVAE	0.6048	0.5561	0.5471	0.5488	0.5823	0.6500	0.5923	0.6162
MERA	**0.6187**	**0.5771**	**0.5646**	0.5425	0.5998	**0.6639**	**0.6434**	**0.6606**

Table 7: Behavioral results for the KuaiSim component ablation.

	KuaiRand Pure						ML1M	
Method	Comment↑	Follow↑	Forward↑	Hate↓	Like↑	Long view↑	Like↑	Star↑
MERA	0.6187	**0.5771**	**0.5646**	0.5425	0.5998	0.6639	**0.6434**	**0.6606**
T1	0.6166	0.5747	0.5616	0.5410	0.5992	0.6613	0.6018	0.6062
T2	0.6059	0.5699	0.5478	**0.5323**	0.5872	0.6567	0.6089	0.6146
T3	0.6174	0.5764	0.5636	0.5443	0.6007	0.6607	0.6181	0.6225
T4	0.6169	0.5759	0.5616	0.5422	0.5992	0.6609	0.6222	0.6259
T5	0.6142	0.5748	0.5638	0.5475	**0.6018**	0.6544	0.5885	0.5980
T6	0.6168	0.5758	0.5571	0.5351	0.5956	**0.6670**	0.6374	0.6523
T7	**0.6191**	0.5765	0.5632	0.5409	0.5998	0.6651	0.6316	0.6486

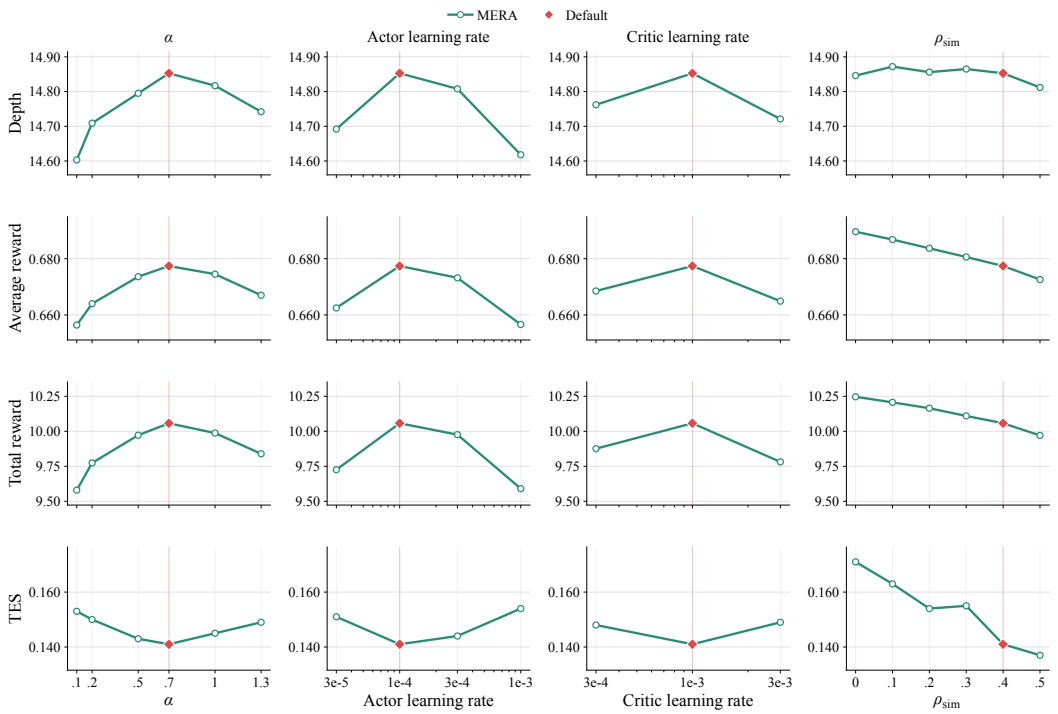


Figure 5: Sensitivity analysis of MERA. We vary α, actor learning rate, critic learning rate, and ρ_{sim}, and report Depth, Average reward, Total reward, and TES.

reward optimization harder while reducing TES. MERA follows this expected reward–diversity tradeoff without abrupt collapse in session depth or return, indicating that its gains are not tied to a brittle hyperparameter choice.

C Appendix: Derivations for MERA

This appendix provides the derivations omitted from the main text. We keep the main body focused on definitions and final statements, while placing the intermediate steps and proofs here.

C.1 Detailed Network Parameterization

This subsection gives one explicit matrix-form implementation of the Z_ϕ-parameterized actor introduced in Section 4.4, together with the hard-action critic interface from Section 4.2. At step t, the request encoder is

$$\begin{aligned} h_t &= f_o(\mathcal{O}_t) \in \mathbb{R}^{d_h}, \\ C_t &= f_c(\mathcal{C}_t) = \big[c_{t,1}^\top, \dots, c_{t,K}^\top\big]^\top \in \mathbb{R}^{K\times d_e}, \\ E &= \big[e_1^\top, \dots, e_{N+1}^\top\big]^\top \in \mathbb{R}^{(N+1)\times d_e}. \end{aligned} \tag{11}$$

Here d_h and d_e denote hidden and embedding dimensions, slots $1,\dots,N$ are exposed positions, slot $N+1$ is the discard slot, $[K]=\{1,\dots,K\}$, $\odot$ denotes elementwise multiplication, and σ is a pointwise activation.

$$\begin{aligned} g_t^{(c)} &= \sigma(W_c h_t + b_c) \in \mathbb{R}^{d_e}, \qquad g_t^{(e)} = \sigma(W_e h_t + b_e) \in \mathbb{R}^{d_e}, \\ \hat{h}_{t,i} &= c_{t,i} \odot g_t^{(c)},\ i \in [K], \qquad \hat{e}_{t,j} = e_j \odot g_t^{(e)},\ j \in [N+1], \\ \hat{H}_t &= \big[\hat{h}_{t,1}^\top, \dots, \hat{h}_{t,K}^\top\big]^\top, \qquad \widetilde{H}_t = \mathrm{Transformer}_\phi^{\mathrm{set}}(\hat{H}_t), \\ [Z_\phi(\mathcal{O}_t)]_{ij} &= \langle \widetilde{h}_{t,i}, \hat{e}_{t,j}\rangle, \qquad Z_\phi(\mathcal{O}_t) \in \mathbb{R}^{K\times(N+1)}. \end{aligned} \tag{12}$$

The feasible marginal plan is obtained by the entropic transport map

$$P_\phi(\mathcal{O}_t) = \mathrm{Sinkhorn}_{\mu,\nu}(Z_\phi(\mathcal{O}_t)/\alpha) \in \mathcal{T}_{K,N}. \tag{13}$$

$$\begin{aligned} R_t &= (X_t^\top C_t)_{1:N,:}, \\ Q_\theta(\mathcal{O}_t, X_t) &= \mathrm{MLP}_\theta\Big(\big[\, h_t;\, \mathrm{MeanPool}\big(\,\mathrm{Transformer}_\theta^{\mathrm{ord}}(R_t)\big)\,\big]\Big). \end{aligned} \tag{14}$$

C.2 Practical Sampler

Gumbel-Sinkhorn sampling. We sample hard assignments using Gumbel perturbations and Sinkhorn scaling (Mena et al., 2018), followed by Hungarian rounding. Exact perturb-and-MAP sampling from a Gibbs distribution over feasible assignments would require an independent Gumbel perturbation for every hard assignment,

$$\arg\max_{X\in\mathcal{X}_{K,N}} \big\{\langle Z_\phi(\mathcal{O}), X\rangle + \varepsilon_X\big\}, \qquad \varepsilon_X \overset{\text{i.i.d.}}{\sim} \mathrm{Gumbel}(0,1),$$

whose cardinality is $|\mathcal{X}_{K,N}| = K!/(K-N)!$. This is infeasible for reranking. Mena et al. (2018) instead use low-dimensional entrywise Gumbel perturbations for matchings; they explicitly describe these rank-one perturbations as "ultimately heuristic." We follow this practical approximation by drawing entrywise Gumbel noise $\varepsilon_{ij} \overset{\text{i.i.d.}}{\sim} \mathrm{Gumbel}(0,1)$, setting $\bar{Z}_\phi(\mathcal{O}) := Z_\phi(\mathcal{O}) + \varepsilon$, and computing

$$\widetilde{P}_\phi(\mathcal{O}, \varepsilon) = \mathrm{Sinkhorn}_{\mu,\nu}\big(\bar{Z}_\phi(\mathcal{O})/\alpha\big)\,,$$

where Z_ϕ denotes the clean score output and $\bar{Z}_\phi$ denotes the Gumbel-perturbed score matrix. The same coefficient α defines the Bellman bonus, the clean marginal map, and the sharpness of this perturb-and-round execution path. Hungarian rounding maps the resulting plan to a hard assignment for execution and critic evaluation,

$$\widetilde{X}_\phi(\mathcal{O}, \varepsilon) = \mathcal{M}_{\mu,\nu}\Big(\widetilde{P}_\phi(\mathcal{O}, \varepsilon)\Big)\,, \qquad \mathcal{M}_{\mu,\nu}(P) \in \arg\max_{X\in\mathcal{X}_{K,N}} \langle P, X\rangle.$$

The discard column has capacity $K - N$. Replicating it $K - N$ times gives a square assignment problem. The equivalent rectangular formulation subtracts each item's discard entry from its display entries in the transport plan and solves a $K \times N$ assignment, leaving unselected items in the discard column. We use this rectangular formulation in the inference scaling sweep. We write this hard assignment using the following straight-through representation, whose forward value equals $\widetilde{X}_\phi$:

$$\widetilde{X}^{\mathrm{st}}_\phi(\mathcal{O}, \varepsilon) := \mathrm{sg}\Big[\widetilde{X}_\phi(\mathcal{O}, \varepsilon) - \widetilde{P}_\phi(\mathcal{O}, \varepsilon)\Big] + \widetilde{P}_\phi(\mathcal{O}, \varepsilon).$$

The critic's action gradient at this hard assignment supplies the detached target used in Appendix C.8. In general, $\mathbb{E}[\widetilde{X}_\phi(\mathcal{O}, \varepsilon)] \neq P_\phi(\mathcal{O})$. The practical sampler therefore induces a hard-action distribution whose true marginal need not equal the actor's parameterized marginal plan. The next subsection explains why this mismatch does not change the contraction results and clarifies how matrix entropy relates to the true policy entropy of the Gumbel-induced hard-action distribution.

C.3 Effect of the Gumbel-Induced Hard-Action Distribution

For a request $\mathcal{O}$, the perturb-and-round sampler induces the hard-action policy

$$\pi^{\mathrm{GS}}_\phi(\cdot \mid \mathcal{O}) := \Pr_\varepsilon\Big(\widetilde{X}_\phi(\mathcal{O}, \varepsilon) \in \cdot\Big),$$

and its true item–slot marginal plan is

$$P^{\mathrm{GS}}_\phi(\mathcal{O}) := P^{\pi^{\mathrm{GS}}_\phi}(\mathcal{O}) = \mathbb{E}_{X \sim \pi^{\mathrm{GS}}_\phi(\cdot|\mathcal{O})}[X].$$

In general,

$$P^{\mathrm{GS}}_\phi(\mathcal{O}) \neq P_\phi(\mathcal{O}),$$

because $P_\phi(\mathcal{O})$ is the actor's parameterized item–slot marginal plan, whereas $P^{\mathrm{GS}}_\phi(\mathcal{O})$ is the actual marginal of the rounded hard-action policy.

Matrix entropy versus true policy entropy. Let $A_i \in [N+1]$ denote the slot assigned to item i under $X \sim \pi^{\mathrm{GS}}_\phi(\cdot \mid \mathcal{O})$. Then, exactly as in the main text,

$$\begin{aligned} \mathcal{H}\big(\pi^{\mathrm{GS}}_\phi(\cdot \mid \mathcal{O})\big) &= H(A_1, \ldots, A_K \mid \mathcal{O}) \\ &\leq \sum_{i=1}^{K} H(A_i \mid \mathcal{O}) \\ &= H_{\mathrm{mat}}\big(P^{\mathrm{GS}}_\phi(\mathcal{O})\big)\,. \end{aligned} \tag{15}$$

Hence matrix entropy remains an upper bound on the Shannon entropy of the actual perturb-and-round hard-action policy when it is evaluated at the policy's true marginal plan $P^{\mathrm{GS}}_\phi(\mathcal{O})$.

What fails in general is the identity between the true marginal $P^{\mathrm{GS}}_\phi(\mathcal{O})$ and the actor's parameterized marginal plan $P_\phi(\mathcal{O})$. Consequently, there is no universal order relation between $H_{\mathrm{mat}}(P_\phi(\mathcal{O}))$ and the true policy entropy $\mathcal{H}(\pi^{\mathrm{GS}}_\phi(\cdot \mid \mathcal{O}))$, nor between $H_{\mathrm{mat}}(P_\phi(\mathcal{O}))$ and $H_{\mathrm{mat}}(P^{\mathrm{GS}}_\phi(\mathcal{O}))$. The quantity $H_{\mathrm{mat}}(P_\phi(\mathcal{O}))$ is therefore best interpreted as a tractable surrogate defined on the actor's parameterized marginal plan, rather than as the exact entropy of the perturb-and-round hard-action policy or as a quantity that uniformly upper- or lower-bounds it.

Why Proposition 1 is unchanged. The policy-evaluation proof does not require mean preservation. More generally, for any fixed hard-action policy π and any bounded state bonus $B(\mathcal{O})$ independent of Q, define

$$(\mathcal{T}_{\pi,B}Q)(\mathcal{O}, X) := \mathbb{E}_{(\mathcal{O}', \mathcal{Y}) \sim p(\cdot|\mathcal{O}, X)}\Big[r(\mathcal{Y}) + \gamma\big(\mathbb{E}_{X' \sim \pi(\cdot|\mathcal{O}')}Q(\mathcal{O}', X') + B(\mathcal{O}')\big)\Big].$$

Then the same derivation as Eq. (17) gives

$$\|\mathcal{T}_{\pi,B}Q_1 - \mathcal{T}_{\pi,B}Q_2\|_\infty \leq \gamma\|Q_1 - Q_2\|_\infty,$$

because B cancels in the difference. Therefore Proposition 1 remains valid whether one uses the ideal bonus $B(\mathcal{O}) = \alpha H_{\mathrm{mat}}(P^\pi(\mathcal{O}))$ or the practical bonus $B(\mathcal{O}) = \alpha H_{\mathrm{mat}}(P_\phi(\mathcal{O}))$ paired with the Gumbel-induced hard-action policy π^{GS}_ϕ.

Why Proposition 2 is unchanged. The same observation applies to the optimality operator. For any policy class Π and any family of bounded bonuses $\{B_\pi\}_{\pi\in\Pi}$ independent of Q, define

$$(\mathcal{T}_{\Pi,B}Q)(\mathcal{O},X) := \mathbb{E}_{(\mathcal{O}',\mathcal{Y})\sim p(\cdot|\mathcal{O},X)}\Big[r(\mathcal{Y}) + \gamma \sup_{\pi\in\Pi}\Big\{\mathbb{E}_{X'\sim\pi(\cdot|\mathcal{O}')}Q(\mathcal{O}',X') + B_\pi(\mathcal{O}')\Big\}\Big].$$

Using $|\sup f - \sup g| \le \sup|f-g|$ exactly as in Eq. (19), one obtains

$$\|\mathcal{T}_{\Pi,B}Q_1 - \mathcal{T}_{\Pi,B}Q_2\|_\infty \le \gamma\|Q_1 - Q_2\|_\infty.$$

Hence Proposition 2 does not require the hard-action policy to preserve the parameterized marginal either. Replacing the ideal marginal by the Gumbel-induced hard-action distribution changes the bonus interpretation, but not the contraction argument.

How Theorem 1 should be read for the practical sampler. The KKT decoupling proof depends on the sampler-induced value term only through its gradient and on the entropy term $H_{\mathrm{mat}}(P)$. No step requires the identity $P = \mathbb{E}[X]$. Therefore, for any sampler whose induced value term is differentiable on the relative interior, the same projected KKT condition

$$\Pi\big(\nabla_P \mathbb{E}_{X\sim\mathsf{S}(P)}\big[Q_\theta(\mathcal{O},X)\big] - \alpha\log P\big) = 0$$

and the same decoupling with the entropic transport inner problem follow for the corresponding abstract objective. For the practical Gumbel-Hungarian sampler, Appendix C.8 approximates this gradient with a biased local critic action gradient evaluated at a sampled hard assignment. This changes the optimizer used in training; the theorem should therefore be read as motivation for the residual loss rather than as an exact statement about the implemented gradients.

C.4 Proof of Proposition 1

For a fixed actor π_ϕ, the regularized evaluation operator is

$$\begin{aligned}(\mathcal{T}_\alpha^\phi Q)(\mathcal{O},X) = \mathbb{E}_{(\mathcal{O}',\mathcal{Y})\sim p(\cdot|\mathcal{O},X)}\Big[r(\mathcal{Y}) \\ + \gamma\big(\mathbb{E}_{X'\sim\pi_\phi(\cdot|\mathcal{O}')}Q(\mathcal{O}',X') + \alpha H_{\mathrm{mat}}(P^{\pi_\phi}(\mathcal{O}'))\big)\Big].\end{aligned} \tag{16}$$

For any $(\mathcal{O},X)$, let $\Delta(\mathcal{O},X) := (\mathcal{T}_\alpha^\phi Q_1)(\mathcal{O},X) - (\mathcal{T}_\alpha^\phi Q_2)(\mathcal{O},X)$. Then the contraction proof can be written as one displayed derivation:

$$\begin{aligned}\|\mathcal{T}_\alpha^\phi Q_1 - \mathcal{T}_\alpha^\phi Q_2\|_\infty &= \sup_{(\mathcal{O},X)}|\Delta(\mathcal{O},X)| \\ &= \gamma \sup_{(\mathcal{O},X)}\big|\mathbb{E}_{(\mathcal{O}',\mathcal{Y})}\mathbb{E}_{X'\sim\pi_\phi(\cdot|\mathcal{O}')}\big[Q_1(\mathcal{O}',X') - Q_2(\mathcal{O}',X')\big]\big| \\ &\qquad \text{(the reward and future entropy terms cancel)} \\ &\le \gamma \sup_{(\mathcal{O},X)}\mathbb{E}_{(\mathcal{O}',\mathcal{Y})}\mathbb{E}_{X'\sim\pi_\phi(\cdot|\mathcal{O}')}\big[\big|Q_1(\mathcal{O}',X') - Q_2(\mathcal{O}',X')\big|\big] \\ &\le \gamma\|Q_1 - Q_2\|_\infty.\end{aligned} \tag{17}$$

Hence $\mathcal{T}_\alpha^\phi$ is a γ-contraction and admits a unique fixed point by Banach's fixed-point theorem.

C.5 Proof of Proposition 2

The regularized optimality operator is

$$\begin{aligned}(\mathcal{T}_\alpha Q)(\mathcal{O},X) = \mathbb{E}_{(\mathcal{O}',\mathcal{Y})\sim p(\cdot|\mathcal{O},X)}\Big[r(\mathcal{Y}) \\ + \gamma \sup_{\pi(\cdot|\mathcal{O}')}\Big\{\mathbb{E}_{X'\sim\pi(\cdot|\mathcal{O}')}Q(\mathcal{O}',X') + \alpha H_{\mathrm{mat}}(P^\pi(\mathcal{O}'))\Big\}\Big].\end{aligned} \tag{18}$$

For any $(\mathcal{O},X)$, let $\Delta^\star(\mathcal{O},X) := (\mathcal{T}_\alpha Q_1)(\mathcal{O},X) - (\mathcal{T}_\alpha Q_2)(\mathcal{O},X)$ and

$$\Psi_i(\mathcal{O}') := \sup_{\pi(\cdot|\mathcal{O}')}\big\{\mathbb{E}_{X'\sim\pi(\cdot|\mathcal{O}')}Q_i(\mathcal{O}',X') + \alpha H_{\mathrm{mat}}(P^\pi(\mathcal{O}'))\big\}, \quad i\in\{1,2\}.$$

Then the optimality contraction is obtained from the single multi-line derivation

$$\begin{aligned}\|\mathcal{T}_\alpha Q_1 - \mathcal{T}_\alpha Q_2\|_\infty &= \sup_{(\mathcal{O},X)} |\Delta^\star(\mathcal{O},X)| \\ &= \gamma \sup_{(\mathcal{O},X)} \left|\mathbb{E}_{(\mathcal{O}',\mathcal{Y})}\big[\Psi_1(\mathcal{O}') - \Psi_2(\mathcal{O}')\big]\right| \\ &\quad \text{(by Eq. (18))} \\ &\le \gamma \sup_{(\mathcal{O},X)} \mathbb{E}_{(\mathcal{O}',\mathcal{Y})}\left[\sup_{\pi(\cdot|\mathcal{O}')} \left|\mathbb{E}_{X'\sim\pi(\cdot|\mathcal{O}')}\big[Q_1(\mathcal{O}',X') - Q_2(\mathcal{O}',X')\big]\right|\right] \\ &\le \gamma \|Q_1 - Q_2\|_\infty.\end{aligned} \tag{19}$$

The inequality uses $|\sup f - \sup g| \le \sup|f-g|$, and the entropy terms cancel because they do not depend on the index i. Thus $\mathcal{T}_\alpha$ is also a γ-contraction and admits a unique fixed point.

C.6 Derivation of Proposition 3

The inner objective is linear plus $\alpha H_{\mathrm{mat}}(P)$ with $\alpha > 0$. Since $-x\log x$ is strictly concave on positive entries and the transport polytope is compact, the maximizer is unique. Because all active row and column marginals are positive, the entropy derivative drives the optimum into the relative interior of $\mathcal{T}_{K,N}$; the nonnegativity inequalities are therefore inactive at the optimum. Let $\mathfrak{L}(P,\xi,\zeta)$ denote the equality-constrained Lagrangian of Eq. (7) with row and column multipliers ξ and ζ for the transport constraints with source marginal μ and target marginal ν. The first-order condition gives

$$\begin{aligned}\mathbf{0} = \frac{\partial \mathfrak{L}}{\partial P_{ij}} &= Z_{ij} - \alpha(1+\log P_{ij}) + \xi_i + \zeta_j \\ &\iff \log P_{ij} = \frac{Z_{ij}+\xi_i+\zeta_j}{\alpha} - 1 \\ &\iff P_{ij} = \exp\left(\frac{Z_{ij}+\xi_i+\zeta_j}{\alpha} - 1\right) = u_i \exp(Z_{ij}/\alpha) v_j \\ &\iff P = \mathrm{Diag}(u)\,\exp(Z/\alpha)\,\mathrm{Diag}(v) \\ &\iff P = \mathrm{Sinkhorn}_{\mu,\nu}(Z/\alpha),\end{aligned} \tag{20}$$

where the constant $\exp(-1)$ is absorbed into the row and column scalings, and the last line is exactly the definition of generalized Sinkhorn scaling over $\exp(Z/\alpha)$ with row marginals μ and column marginals ν.

C.7 Derivation of Theorem 1

For fixed $\mathcal{O}$, assume the mapping $P \mapsto \mathbb{E}_{X\sim\mathsf{S}(P)}[Q_\theta(\mathcal{O},X)]$ is continuously differentiable on a neighborhood of $\mathrm{relint}(\mathcal{T}_{K,N})$. The exact marginal actor problem is

$$\max_{P\in\mathcal{T}_{K,N}} \Big\{\mathbb{E}_{X\sim\mathsf{S}(P)}\big[Q_\theta(\mathcal{O},X)\big] + \alpha H_{\mathrm{mat}}(P)\Big\}.$$

Assume $P \in \mathrm{relint}(\mathcal{T}_{K,N})$ and $P_{ij} > 0$ for all (i,j). Let ξ and ζ denote the Lagrange multipliers for the row and column constraints. The first-order KKT stationarity condition of the exact problem is

$$\begin{aligned}\mathbf{0} &= \nabla_P \mathbb{E}_{X\sim\mathsf{S}(P)}\big[Q_\theta(\mathcal{O},X)\big] - \alpha\big(\log P + \mathbf{1}_K \mathbf{1}_{N+1}^\top\big) \\ &\quad + \xi \mathbf{1}_{N+1}^\top + \mathbf{1}_K \zeta^\top \qquad \text{(KKT stationarity)} \\ &\implies \Pi\big(\nabla_P \mathbb{E}_{X\sim\mathsf{S}(P)}\big[Q_\theta(\mathcal{O},X)\big] - \alpha \log P\big) = 0 \\ &\qquad \text{(row and column shifts vanish under } \Pi\text{)}.\end{aligned} \tag{21}$$

This projected condition is equivalent to first-order stationarity because the constraints are affine and P lies in the interior of the transport polytope. Now suppose Z also satisfies the inner subproblem

$$P = \arg\max_{\bar{P}\in\mathcal{T}_{K,N}} \Big\{\langle Z, \bar{P}\rangle + \alpha H_{\mathrm{mat}}(\bar{P})\Big\}$$

By Proposition 3, this is equivalent to $P = \mathrm{Sinkhorn}_{\mu,\nu}(Z/\alpha)$. Hence there exist row and column shift vectors a and b such that

$$\begin{aligned} \log P &= Z/\alpha + a\mathbf{1}_{N+1}^\top + \mathbf{1}_K b^\top \\ &\Longrightarrow \Pi(\alpha \log P) = \Pi(Z) \\ &\qquad \text{(the row and column shifts are again removed by } \Pi\text{)}. \end{aligned} \tag{22}$$

Substituting into Eq. (21) gives

$$\Pi\big(\nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O}, X)\big] - \alpha \log P\big) = 0 \iff \Pi\big(Z - \nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O}, X)\big]\big) = 0. \tag{23}$$

Therefore, stationarity plus the inner subproblem implies the outer condition in Eq. (8). Conversely, if both subproblems hold, then the equivalence above implies

$$\Pi\big(\nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O}, X)\big] - \alpha \log P\big) = 0,$$

which is exactly the projected KKT condition of the original problem. This proves the exact decoupling.

C.8 Practical Approximation of the Outer Loss

Equation (10) is the parameterized residual of the exact outer condition:

$$\mathcal{L}_Z(\phi) = \frac{1}{2}\,\mathbb{E}_{\mathcal{O}\sim d_{\mathcal{B}}}\left[\left\|\Pi\Big(Z_\phi(\mathcal{O}) - \nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O}, X)\big]\big|_{P=P_\phi(\mathcal{O})}\Big)\right\|_F^2\right].$$

Its core term is the exact outer gradient

$$\nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O}, X)\big]\big|_{P=P_\phi(\mathcal{O})},$$

which is generally unavailable because differentiating the sampler-induced expectation is intractable and the practical perturb-and-round path includes a nondifferentiable rounding step. We therefore optimize a practical surrogate based on Appendix C.2. Let $\bar{Z}_\phi(\mathcal{O}) := Z_\phi(\mathcal{O}) + \varepsilon$, and define

$$\begin{aligned} \widetilde{P}_\phi(\mathcal{O}, \varepsilon) &= \mathrm{Sinkhorn}_{\mu,\nu}\big(\bar{Z}_\phi(\mathcal{O})/\alpha\big), \\ \widetilde{X}_\phi(\mathcal{O}, \varepsilon) &= \mathcal{M}_{\mu,\nu}\big(\widetilde{P}_\phi(\mathcal{O}, \varepsilon)\big), \\ \widetilde{X}^{\mathrm{st}}_\phi(\mathcal{O}, \varepsilon) &= \mathrm{sg}\Big(\widetilde{X}_\phi(\mathcal{O}, \varepsilon) - \widetilde{P}_\phi(\mathcal{O}, \varepsilon)\Big) \;+\; \widetilde{P}_\phi(\mathcal{O}, \varepsilon). \end{aligned}$$

The practical approximation replaces the exact outer gradient by

$$\nabla_P \mathbb{E}_{X\sim \mathsf{S}(P)}\big[Q_\theta(\mathcal{O}, X)\big]\big|_{P=P_\phi(\mathcal{O})} \approx \nabla_X Q_\theta\Big(\mathcal{O}, \widetilde{X}^{\mathrm{st}}_\phi(\mathcal{O}, \varepsilon)\Big),$$

evaluated at the sampled hard assignment. This critic action gradient is a biased local surrogate for the sampler-induced value gradient. Define the per-sample residual

$$\widetilde{R}_\theta(\mathcal{O}, \varepsilon) := \Pi\Big(Z_\phi(\mathcal{O}) - \mathrm{sg}\Big[\nabla_X Q_\theta\Big(\mathcal{O}, \widetilde{X}^{\mathrm{st}}_\phi(\mathcal{O}, \varepsilon)\Big)\Big]\Big).$$

This yields the surrogate loss

$$\widehat{\mathcal{L}}_Z(\phi) = \frac{1}{2}\,\mathbb{E}_{\mathcal{O}\sim d_{\mathcal{B}},\ \varepsilon\sim\mathrm{Gumbel}}\left[\left\|\widetilde{R}_\theta(\mathcal{O}, \varepsilon)\right\|_F^2\right], \tag{24}$$

which approximates Eq. (10).

Gradient of the surrogate actor loss. Let $J^\Pi_\phi(\mathcal{O}) := \partial \Pi(Z_\phi(\mathcal{O}))/\partial\phi$. With the critic-gradient target held fixed by stop-gradient, differentiating the projected residual loss gives

$$\nabla_\phi \widehat{\mathcal{L}}_Z = \mathbb{E}_{\mathcal{O}\sim d_{\mathcal{B}},\ \varepsilon\sim\mathrm{Gumbel}}\Big[J^\Pi_\phi(\mathcal{O})^\top \widetilde{R}_\theta(\mathcal{O}, \varepsilon)\Big], \tag{25}$$

where the critic-gradient target is held fixed during the score update and recomputed at a newly sampled hard assignment for the next actor step. Thus the update fits the projected scores to the critic-gradient target.

D Appendix: Additional Clarifications and Analyses

This appendix provides additional clarifications and analyses that complement the main text. These materials are intended to improve clarity and reproducibility, and to make explicit certain design choices that are implicit in the main presentation.

Downstream utility. Figure 4 examines how policy perturbations affect executed slates. Figure 3 and Appendix B.1 evaluate downstream utility across six settings, where MERA achieves the highest mean Depth and reward in all six settings while remaining competitive in session-level exposure diversity. The behavioral results in Appendix B.2 further characterize positive and negative feedback in KuaiSim.

Exploration–exploitation. The matrix-entropy coefficient α controls both the regularization strength and the sharpness of the Sinkhorn plan. Larger α encourages more diffuse item–slot mass; smaller α makes the plan sharper and more exploitative. Table 1 reports stochastic–deterministic contrasts for each critic and actor-update configuration, while Figure 5 shows performance around the default α.

Marginals versus executed actions. MERA distinguishes the parameterized plan $P_\phi(\mathcal{O})$ from the true marginal of the Gumbel–Sinkhorn/Hungarian hard-action distribution. In general these are not equal, as discussed in Appendix C.3. The entropy term should therefore be interpreted as a tractable plan regularizer, not the exact entropy of the rounded policy. This mismatch does not invalidate the reported rewards, because the simulator, replay buffer, and critic all operate on the executed hard slate X.

Ablation scope. The ablation in Table 1 is scoped to MERA's own design factors under a common Sinkhorn–Hungarian assignment backbone: stochastic execution, ordered-slate critic, and KKT-style update. The purpose is to isolate MERA's internal design choices rather than to expand the external baseline taxonomy. The external baselines cover continuous and latent actions, policy gradients, Sinkhorn policies, pointwise scoring, and conditional slate generation.

Interpretability. MERA's interpretability is at the exposure-plan level: practitioners can inspect logits, marginal mass over item–position–discard assignments, and the final rounded slate. The Gumbel–Sinkhorn and Hungarian steps still introduce stochasticity and rounding, so a single displayed slate is not a complete explanation of the policy. The benefit is that both the pre-rounding plan and the executed action live in auditable item–position coordinates, unlike latent actions whose exposure effect is only visible after a separate decoder.

E Appendix: Broader Impact

MERA is intended as a technical method for sequential reranking. If deployed in recommender systems, assignment-native control could improve user experience by making the displayed slate more directly aligned with the optimization interface and by reducing homogeneous repeated exposure. The simulator results on temporal exposure similarity suggest a possible path toward more varied recommendations, but they should be interpreted as offline evidence rather than a guarantee of online impact.

The same capability can also amplify common risks in recommender systems. Optimizing long-horizon reward may overemphasize engagement signals, reinforce feedback loops, shift exposure among items or creators in unfair ways, or produce unintended user-level outcomes when the simulator is misspecified. Any real deployment should therefore include online monitoring, fairness and exposure audits, privacy-preserving data handling, and explicit safety constraints appropriate to the application domain. MERA is not designed to replace these deployment-specific governance mechanisms.

F APPENDIX: AI USE STATEMENT

We used AI tools to refine the manuscript's layout and formatting, check grammar, and polish the language. The authors take full responsibility for the final manuscript.

G APPENDIX: ETHICS STATEMENT

Our experiments use the public KuaiRand Pure and MovieLens-1M datasets and evaluate recommendation policies in offline simulators. Potential deployment risks include feedback loops from optimizing engagement, unequal exposure across items or creators, and user privacy concerns. Appendix E discusses these risks and corresponding monitoring and auditing measures.

H APPENDIX: REPRODUCIBILITY STATEMENT

The code for all experiments is available at `https://anonymous.4open.science/r/MERA_code-CD4D/`. Appendix A documents data processing, simulator configurations, training settings, and evaluation protocols. Appendix C provides network details and theoretical derivations, and Algorithm 1 summarizes MERA's training and inference procedure.