In the appendix, we provide additional details on the derivations of our methods and analysis, in addition to methodological details omitted from the main paper. In Appendix A, we directly derive the formulas necessary for our querying via information gain optimization approach. Appendices B to D present details on our main approach in Algorithm 1. Appendix E provides the arguments needed to justify the claims of Section 4.4. Finally, Appendices F to I present details on our experimental setups, while Appendix J presents a compelling additional synthetic data experiment demonstrating learning a reward function mixture with five modes.

A Information Gain Derivation

We present the derivation of the formula for computing the maximum information gain query Q^*. Assume at a fixed timestep t we have made past query observations $\mathcal{D} = \{Q^{(t')}, x^{(t')}\}_{t'=1}^{t-1}$. The desired query is then

$$Q^* = \arg\max_Q I(X; \Theta \mid Q, \mathcal{D}), \tag{6}$$

where $I(\cdot;\cdot)$ denotes mutual information. Equivalently, denoting conditional entropy with $\mathcal{H}[\cdot \mid \cdot]$, we note

$$I(X; \Theta \mid Q, \mathcal{D}) = \mathcal{H}\left[\{\alpha_m, \omega_m\}_{m=1}^M \mid \mathcal{D}\right] - \mathop{\mathbb{E}}_{P(X|Q,\mathcal{D})}\left[\mathcal{H}\left[\{\alpha_m, \omega_m\}_{m=1}^M \mid Q, X = x, \mathcal{D}\right]\right],$$

which allows us to write the optimization in Eq. (6) equivalently as

$$Q^* = \arg\min_Q \mathop{\mathbb{E}}_{P(X|Q,\mathcal{D})}\left[\mathcal{H}\left[\{\alpha_m, \omega_m\}_{m=1}^M \mid Q, X = x, \mathcal{D}\right]\right].$$

We further simplify this minimization objective by denoting the joint distribution over x and $\theta = \{\alpha_m, \omega_m\}_{m=1}^M$ conditioned on Q and $\mathcal{D}$ as $P(X, \Theta \mid Q, \mathcal{D})$ and expanding the entropy term:

$$\begin{aligned} Q^* &= \arg\min_Q \mathop{\mathbb{E}}_{P(X,\Theta|Q,\mathcal{D})} \log \frac{\Pr\left[X = x \mid Q, \mathcal{D}\right]}{\Pr[X = x \mid Q, \theta]} \\ &= \arg\min_Q \mathop{\mathbb{E}}_{P(X,\Theta|Q,\mathcal{D})} \log \frac{\mathbb{E}_{\theta' \sim \Theta|\mathcal{D}} \Pr\left[X = x \mid Q, \theta'\right]}{\Pr[X = x \mid Q, \theta]}. \end{aligned} \tag{see 4}$$

B Metropolis-Hastings

B.1 Approach

To sample from $\Pr(\Theta \mid \mathcal{D})$ using Eq. (3), we use the Metropolis-Hastings algorithm [62], running N chains simultaneously for H_{MH} iterations. To avoid autocorrelation between samples, unlike in conventional Metropolis-Hastings we only use the last state in each chain as a sample. In contrast, for conventional Metropolis-Hastings, multiple samples would be drawn from a single chain at set intervals after a short *burn-in* period. As we see in Fig. 7, for our multimodal Plackett-Luce posteriors, performing multi-chain Metropolis-Hastings yields posterior samples that are far more evenly distributed across different posterior modes. Thus, to achieve well-distributed posterior samples, we set our effective burn-in period to be $H_{MH} - 1$, taking only the last sample from each chain.

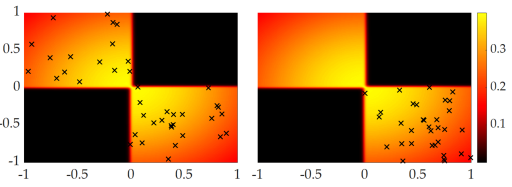


Figure 7: Multi-chain Metropolis-Hastings sampling **(left)** gives more representative samples from the distribution compared to the single-chain variant **(right)**.

For two states in the chain $\Theta = \{\alpha_m, \omega_m\}_{m=1}^M$ and $\Theta' = \{\alpha'_m, \omega'_m\}_{m=1}^M$, our proposal distribution is then

$$g(\Theta' \mid \Theta) = \prod_{m=1}^M \varphi(\omega_m - \omega'_m),$$

where φ is the pdf of the diagonal Gaussian $\mathcal{N}(0, \sigma_{MH} I)$.

B.2 Multimodal Metropolis-Hastings Demonstration

The posterior distribution in the figure is that of a 2-mode Plackett-Luce mixture with fixed uniform mixing coefficients and 1-D weights conditioned on the observations $50 \succ -50$ and $-50 \succ 50$. The single-chain algorithm ran for 2000 steps with a burn-in period of 200 steps after which every 18th sample was selected, while the multi-chain algorithm used 100 chains for 20 iterations each, taking only the last sample from each chain.

C Simulated Annealing

For our simulated annealing, we run N_{SA} chains in parallel for H_{SA} iterations each, returning the best query Q found across each run. We define the transition proposal distribution $g(Q' \mid Q)$ to be a positive constant if Q' and Q differ by one trajectory and 0 otherwise. We run with a starting temperature of T^0_{SA}, cooling by a factor of γ_{SA} with each subsequent iteration past the first.

D Hyperparameters

We use the hyperparameters in Table 1 for the simulated annealing and Metropolis-Hastings algorithms, whose details are provided in Appendix B and Appendix C, respectively.

Table 1: Hyperparameters

Constant	Value
N	100
H_{MH}	200
σ_{MH}	0.15
N_{SA}	10
H_{SA}	30
T^0_{SA}	10
γ_{SA}	0.9

E Proofs and Analysis

E.1 Proof of Corollary 1.1

Corollary 1.1. *A mixture of M Plackett-Luce models with query size K is* generically identifiable *if $M \leq \lfloor \frac{K-2}{2} \rfloor !$.*

Suppose we have such a mixture of M Plackett-Luce models that is not identifiable. Then, there must exist two distinct sets of parameters Θ_1 and Θ_2 such that for every query Q, the induced ranking distributions X_1 and X_2 respectively are identical. But since Θ_1 and Θ_2 are distinct, there is either (1) two mixing coefficients in Θ_1 and Θ_2 that disagree or (2) two items ξ_1 and ξ_2 that have a different difference in rewards across Θ_1 and Θ_2 under one of the reward functions. Let $\bar{Q}$ with corresponding ranking distribution $\bar{X}$ be an arbitrary query in case (1) and an arbitrary query containing ξ_1 and ξ_2 in case (2). Note that $\bar{X}$ is the marginal distribution of the overall Plackett-Luce distribution, which by construction is a mixture of M Plackett-Luce models with parameters Θ_1 and Θ_2, restricted to the items in $\bar{Q}$. But now there are two distinct sets of parameters representing the distribution over the full ranking of Q since we know Θ_1 and Θ_2 differ on the restricted set of items $\Xi' = Q$ (either because they have differing mixing coefficients or because their induced rewards on Ξ' are not a within a constant additive factor of each other since ξ_1 and ξ_2 are in Ξ'). But we know $|\Xi'| = |Q| = K$, so this finding contradicts the fact that $\bar{X}$ must be identifiable by Theorem 1. We conclude every mixture of M Plackett-Luce models is identifiable subject to the query size bounds in the statement of this corollary. □

E.2 Justification for Remark 1

Remark 1. *Greedy selection of queries maximizing information gain in Eq. (4) is not necessarily within a constant factor of optimality.*

Here, we define the optimal adaptive set of queries $\mathcal{D}$ to be the one which, in expectation, minimizes the uncertainty over model parameters $\mathcal{H}[\Theta \mid \mathcal{D}]$. It is a well-known result that for *adaptive submodular* functions, greedy optimization yields results that are within a constant factor $\left(1 - \frac{1}{e}\right)$ of optimality [63]. While our mutual information objective in Eq. (4) is adaptive submodular in the non-adaptive setting (where all queries Q are selected before observing their results), in our adap-

tive setting these guarantees no longer hold (conditional entropy is only submodular with respect to conditioned variables if those variables are unobserved).

F Baselines

F.1 Random

We benchmark against a random agent, wherein at each step the query selected by the agent is a collection of K random items without replacement. We also use the random querying method for comparing the multimodal reward learning with the existing approaches that assume a unimodal reward (e.g. [5]), as it does not introduce any bias in the query selection.

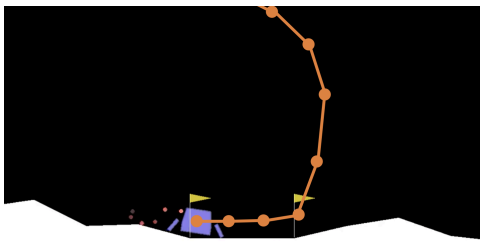
F.2 Volume Removal

Volume removal seeks to maximize the difference between the prior distribution over model parameters and the *unnormalized* posterior. Volume removal notably fails to be optimal in domains where there are similar trajectories [29]. In these settings, querying sets of trajectories with similar features removes a large amount of volume from the unnormalized posterior (since the robot is highly uncertain about their relative quality), yet yields little information about the model parameters (since the human also has high uncertainty). Information gain approaches such as our method are better able to generate trajectories to query for which the robot has high uncertainty while the human has enough certainty to yield useful information for the robot.

G Trajectory Generation

G.1 LunarLander Trajectories

We designed 8 trajectory features based on: absolute heading angle accumulated over trajectory, final distance to the landing pad, total amount of rotation, path length, task completion (or failure) time, final vertical velocity, whether the lander landed on the landing pad without its body touching the ground, and original environment reward from OpenAI Gym. Using these features, we randomly generated 10 distinct reward functions based on the linear reward model and trained a DQN policy [61] for each reward. Finally, we generated 100 trajectories by following each of these 10 policies in the environment to obtain 1000 trajectories in total. We used these trajectories as our dataset for the ranking queries. Fig. 8 presents an example trajectory with extracted scaled and centered features.



Feature	Value
Mean angle	2.27683634
Total angle	−0.20375356
Distance to Goal	5.41860642
Total rotation	0.25948072
Path length	3.71660086
Final vertical velocity	−0.57097337
Crash time	1.11112885
Score	−0.15500268

Figure 8: Sample LunarLander trajectory (left) with extracted features (right).

G.2 Fetch Trajectories

To design our 351 trajectories, we varied the target shelf (3 variations), the movement speed (3), the grasp point on the banana (3) and where in the shelf it is placed (13). We then designed 12 trajectory features based on these varied parameters and appended another binary feature which indicates whether any object dropped from the shelves on that trajectory.

Specifically, for τ a trajectory, let

$$x_i = \begin{cases} 1 & i \text{ is the target shelf} \\ 0 & \text{otherwise} \end{cases},$$

$y_{\text{grasp}}, y_{\text{height}}, y_{\text{width}}, y_{\text{speed}}$ specify the grasp position and speed, and y_{success} specify whether the robot did not drop any objects from the shelves. Our featurization is then

$$\begin{aligned}\Phi(\tau) = \big(&x_1, x_2, x_3, y_{\text{speed}}, y_{\text{speed}}(1-y_{\text{speed}}), y_{\text{grasp}},\\ &y_{\text{grasp}}(1-y_{\text{grasp}}), y_{\text{height}}, y_{\text{height}}(1-y_{\text{height}}),\\ &y_{\text{width}}, y_{\text{width}}(1-y_{\text{width}}), 1-(y_{\text{grasp}}-y_{\text{width}})^2, y_{\text{success}}\big).\end{aligned}$$

Fig. 9 presents a sample Fetch trajectory with its featurization.

Feature	Value
x_1	1.0000
x_2	0.0000
x_3	0.0000
y_{speed}	0.5000
$y_{\text{speed}}(1-y_{\text{speed}})$	0.2500
y_{grasp}	1.0000
$y_{\text{grasp}}(1-y_{\text{grasp}})$	0.0000
y_{height}	0.7500
$y_{\text{height}}(1-y_{\text{height}})$	0.1875
y_{width}	0.2500
$y_{\text{width}}(1-y_{\text{width}})$	0.1875
$1-(y_{\text{grasp}}-y_{\text{width}})^2$	0.4375
y_{success}	1.0000

Figure 9: Sample Fetch trajectory (left) with extracted features (right).

H Metrics

H.1 MSE

Our metric is

$$\mathcal{M}_{\text{MSE}} = \sum_{m=1}^{M} \|\omega_m^* - \widehat{\omega}_m\|_2^2 \tag{7}$$

where $\Theta^* = \{\alpha_m^*, \omega_m^*\}_{m=1}^M$ and $\widehat{\Theta} = \{\widehat{\alpha}_m, \widehat{\omega}_m\}_{m=1}^M$ and the learned reward weights of the experts are matched with the true weights using the Hungarian algorithm. When the learning model assumes a unimodal reward function, as in our simulations for Fig. 3, we compute the MSE metric as $\sum_{m=1}^M \|\omega_m^* - \hat{\omega}\|_2^2$.

H.2 Log-Likelihood

Formally, we define the log-likelihood metric as

$$\mathcal{M}_{\text{LL}} = \mathbb{E}_{Q\sim\mathcal{Q}}\big[\mathbb{E}_{x\sim P(X|Q)} \log \Pr(x \mid \mathcal{Q}, D)\big] \tag{8}$$

for $\mathcal{Q}$ the uniform distribution across all possible queries and $P(x \mid Q)$ the distribution over the human's response to query Q (as in Eq. (2)). We can compute the inner term

$$\Pr(x \mid Q, \mathcal{D}) = \mathbb{E}_{\Theta|\mathcal{D}}[\Pr(x \mid Q, \Theta)]$$

using Metropolis-Hastings as in Section 4.3 to sample from the posterior $\Pr(\Theta \mid \mathcal{D})$ and computing the inner term with Eq. (2).

H.3 Learned Policy Rewards

Similar to the MSE metric, we match the rewards learned via DQN [61] with the true rewards using the Hungarian algorithm.

I Experimental Setup

I.1 Shelf Descriptions

A picture of each shelf accompanied the following descriptions.

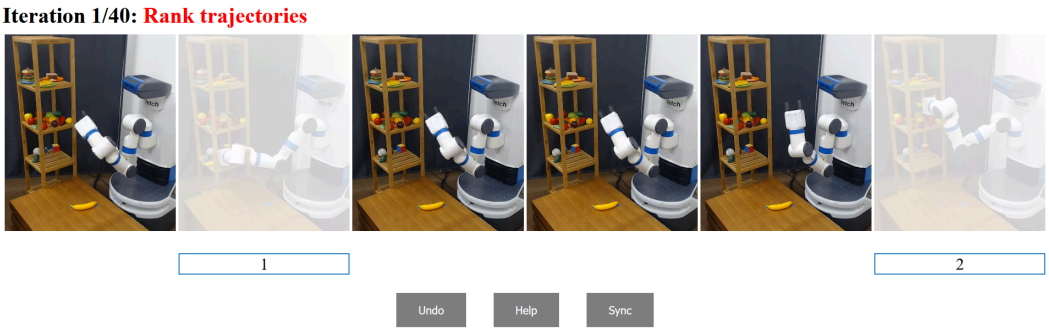


Figure 10: The user interface for the online studies with the real Fetch robot. The user selected the 2nd trajectory as their top choice and the 6th trajectory as the second top.

- The top shelf has some space, but you usually put cooked meals there.
- The middle shelf is for fruits, but it is already full. The robot may accidentally drop other fruits.
- The bottom shelf has a lot of space, but you have been using it for toys.

I.2 User Interface

For both environments, subjects were told they need to rank the six trajectories in each query by clicking on the trajectories starting from the most preferred to the least. The web interface (see Fig. 10) equipped them with "Undo" and "Sync" buttons. "Undo" allowed the subjects to undo a selection they make within a query. "Sync" enabled them to restart all videos in the query.

J Synthetic Experiment

J.1 Testing $M > 2$

For our first experiment with synthetic data, we demonstrate effectiveness of our approach for learning mixtures of more than two Plackett-Luce models. In particular, we evaluate our approaches using 250 sets of five randomly simulated reward weights ($M = 5$, $K = 6$), and trajectory features defined by $\Phi(\xi_{1:10}) \sim \mathcal{N}(0, I)$, $\Phi(\xi_{11:110}) \sim \mathcal{N}(0, 0.1I)$, and $\Phi(\xi_{111:1110}) \sim \mathcal{N}(0, 0.01I)$ where I is the 3×3 identity matrix and $\xi_{n:k}$ refers to the nth through kth trajectory in the environment . This environment models complex multimodal structure in the trajectory feature space, which is common to many robotic settings.

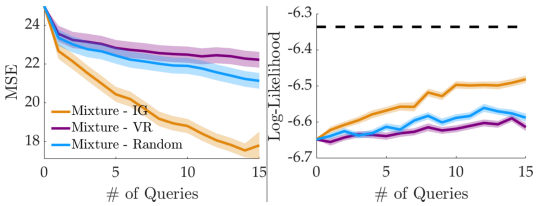


Figure 11: Different querying methods are compared on a synthetic environment (mean±se over 250 runs).

Fig. 11 shows the results of our experiments. We see our approach, Mixture - IG dramatically outperforms the other approaches in both the MSE and log-likelihood metrics.

J.2 Testing Robustness to M Parameter

We also test the robustness of our Mixture - IG approach to misspecified M values. We repeat the previous experiment, testing the Mixture - IG approach varying the misspecified value of M between $M = 1$ (Unimodal) and $M = 3, 5, 7$ (see Fig. 12). We use the log-likelihood metric since MSE is not well-defined for methods with $M \neq 5$ because they learn a mixture of a different number of reward functions from the true synthetic mixture.

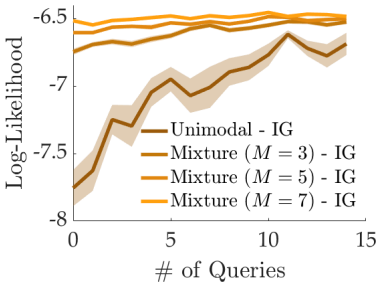


Figure 12: Different values of M for the IG approach are compared (mean±se over 100 runs).

We see the best performance occurs for $M = 5$ and $M = 7$, with only $M = 1$ performing significantly worse. We conclude that in this experiment our Mixture - IG approach is relatively robust to the value of M, as long as a sufficiently large value > 1 is selected.

K Additional Unimodal Baseline

We test an additional baseline on the random queries made during user studies to show the superiority of our learning approach. The additional baseline represents selecting the unimodal reward with fixed norm that maximizes the reward of the top trajectory of each expert-ranked query. We compare this baseline against a learning method that computes the bimodal MLE of the reward function. Formally, for query responses $\mathcal{D} = \left\{Q^{(t')}, x^{(t')}\right\}_{t'=1}^{t}$ with $\xi^{(t')}$ the top trajectory in the ranking $x^{(t')} = (\xi^{(t')}, \ldots)$, we define this baseline to learn the parameters $\theta = \{\alpha, \omega\}$ where $\alpha = 1$ and

$$\tilde{\omega} = \sum_{t'=1}^{t} \Phi(\xi^{(t')})$$
$$\omega = \frac{\tilde{\omega}}{\|\tilde{\omega}\|_2}.$$

Note that we do not vary the querying method in this experiment. Rather, we compare two methods of learning reward weights from the 15 random human queries that were performed by the Mixture - Random algorithm on the Fetch Robot and LunarLander during our user studies, and then evaluate these methods on the 10 random evaluation queries presented to the humans at the end of the experiment. We compare the two methods in terms of the log-likelihood metric. The results are presented in Table 2, with our method denoted as "Mixture MLE" and the new baseline described above denoted as "Baseline".

Table 2: Additional User Study Reward Learning Baseline

	LunarLander		Fetch Robot	
	Baseline	Mixture MLE	Baseline	Mixture MLE
log-likelihood	-8.23 ± 0.31	-5.91 ± 0.18	-5.21 ± 0.22	-4.70 ± 0.35
p-value	$6.2 \cdot 10^{-7}$		0.11	

We see the Mixture MLE method outperforms the Baseline method on both environments, with statistical significance ($p < 0.05$) in LunarLander when conducting paired t-TESTS.