SUBSPACE INFERENCE ENABLES ACTIVE PREFERENCE BASED LEARNING OF NEURAL NETWORK REWARD MODELS

Anonymous authorsPaper under double-blind review

ABSTRACT

Reinforcement learning from human feedback (RLHF) has emerged as a powerful approach for aligning decision-making agents with human intentions, primarily through the use of reward models trained on human preferences. However, RLHF suffers from poor sample efficiency, as each feedback provides minimal information, making it necessary to collect large amounts of human feedback. Active learning addresses this by enabling agents to select informative queries, but effective uncertainty quantification required for active learning remains a challenge. While ensemble methods and dropout are popular for their simplicity, they are computationally expensive at scale and do not always provide good posterior approximation. Inspired by the recent advances in approximate Bayesian inference, we develop a method that leverages Bayesian filtering in neural network subspaces to efficiently maintain model posterior for active reward modeling. Our approach enables scalable sampling of neural network reward models to efficiently compute active learning acquisition functions. Experiments on the D4RL benchmark demonstrate that our approach achieves superior sample efficiency, scalability, and calibration compared to ensemble methods and dropout, and leads to competitive offline reinforcement learning policy performance. This highlights the potential of scalable Bayesian methods for preference-based reward modeling in RLHF.

1 Introduction

In recent years, reinforcement learning from human feedback (RLHF) has become the dominant technique for aligning decision-making agents with human intentions (Christiano et al., 2017; Ouyang et al., 2022). The ease of providing preference feedback has been a crucial factor in their popularity as a feedback type for reward modeling, but since each feedback provides at most 1 bit of information, they are also known for their poor sample efficiency; asking a human thousands of comparison questions to learn a reward model is often not scalable.

A core problem of RLHF is active learning, where we want an agent to be judicious about the queries it asks to learn about a human's preferences as efficiently as possible (Sadigh et al., 2017; Casper et al., 2023; Baraka et al., 2025). Many active learning approaches require probabilistic modeling of uncertainty for computing data acquisition functions, making proper uncertainty representation an active area of research (Ovadia et al., 2019; Tran et al., 2020; Papamarkou et al., 2024). While Bayesian methods are well-principled, they are hard to scale to large neural networks (NN) (Izmailov et al., 2021). On the other hand, the simplicity of ensemble method (Dietterich, 2000; Lakshminarayanan et al., 2016) has made it a popular choice for active learning. However, training multiple models can be computationally intensive, especially for large NN reward models.

Due to recent advancements in approximate inference, Bayesian deep learning have become increasingly scalable (Daxberger et al., 2024; Shen et al., 2024). In this work, we develop a method called PreferenceEKF that enables efficient training of Bayesian neural networks for representing reward models in active preference-based reward learning. Specifically, by performing Bayesian filtering in a constructed neural network subspace, we maintain model uncertainty in a compute-and memory-efficient manner. The reduced dimensionality of the subspace enables application of the extended Kalman filter, a classic inference method, for training neural networks. This allows

sampling of arbitrary number of reward models from the model posterior, and use the samples for computing common uncertainty-based acquisition functions such as expected information gain and disagreement (Hennig & Schuler, 2012; Hernández-Lobato et al., 2014; Bıyık et al., 2022).

To the best of our knowledge, we are the first to leverage subspace filtering (Duran-Martin et al., 2022) to train neural network reward models for preference-based reward learning. We compare our posterior inference method to the widely used ensemble method and dropout, and conduct experiments in the D4RL (Fu et al., 2020) benchmark. Our findings are as follows:

- Active reward learning using PreferenceEKF leads to better sample efficiency (in terms of the number of queries required) compared to reward learning from random queries.
- PreferenceEKF performs on par with or better than ensemble method and dropout in terms of sample efficiency and calibration in preference modeling tasks.
- PreferenceEKF's computation time scales better with both model size and number of model samples used for computing acquisition function and posterior predictive distribution compared to ensemble method.
- When the learned rewards are used for policy optimization in offline reinforcement learning tasks (Levine et al., 2020), the reward model learned using active PreferenceEKF resulted in the best overall policy rollout performance compared to reward models learned using ensemble method and dropout.

2 Related Work

Reinforcement learning from human preferences. While early works in reward learning focused on learning from expert demonstrations (Abbeel & Ng, 2004; Finn et al., 2016; Ho & Ermon, 2016), much recent interest has focused on reward learning from pairwise comparisons where human annotators are asked to compare two potential outcomes, e.g., labels, responses, or trajectories (Wirth et al., 2017; Christiano et al., 2017; Brown et al., 2019). Although preference feedback is much easier for annotators to provide than demonstrations, the minimal amount of information contained within a binary preference query necessitates collection of large amounts of feedback data.

Active learning is a widely used approach for minimizing the time-consuming process of collecting human feedback. It is a sequential problem in nature, as it iteratively collects the most useful data sample based on the model's current state, such as parameter posterior uncertainty. (Sadigh et al., 2017; Settles, 2009). While Bayesian methods have been successfully applied to obtain posteriors for active reward learning using lower-dimensional linear and Gaussian process reward models (Bıyık et al., 2022; 2024), it has not been widely adopted for neural reward models, since acquisition functions typically require sampling from the high-dimensional distribution of model parameters. Instead, ensembles have been the key enabler of neural network based active reward learning (Lee et al., 2021b; Christiano et al., 2017). Our work focuses on efficient yet performant posterior inference for active reward learning, without expensive training of multiple independent models.

Uncertainty Quantification for neural networks. Classic Bayesian methods that have been successfully used for neural network uncertainty quantification include Laplace approximation (Daxberger et al., 2024), Hamiltonian Monte Carlo (Neal, 2011), and variational inference (Blei et al., 2017). While not strictly motivated by Bayesian principles, the simplicity of ensemble method (Dietterich, 2000; Lakshminarayanan et al., 2016) and dropout (Srivastava et al., 2014; Gal & Ghahramani, 2016) has made them popular for UQ. While dropout method gets around ensemble method's expense cost of training multiple independent models, it been shown to lead to poor posterior approximation quality (Hron et al., 2018; Osband, 2016).

Bayesian filtering methods, which focuses on inferring hidden states from noisy observations, provide a principled approach to sequential learning, and have been widely used in robotics and signal processing (Thrun et al., 2005; Särkkä & Svensson, 2023). Application of Bayesian filtering for training neural networks (Singhal & Wu, 1988; de Freitas et al., 2000) has only recently been applied to deep neural networks via subspace methods by Duran-Martin et al. (2022) for neural bandits.

Instead of deriving epistemic uncertainty from posterior inference, a separate line of work has focused on leveraging nonparametric statistics techniques such as the bootstrap to perform UQ (Efron, 1992), and has successfully applied this technique for exploration in deep reinforcement learning (Osband et al., 2018; 2016). The same group of authors have also leveraged joint predictions for

UQ, and has applied the idea to finetuning large language models (Osband et al., 2023b;a). Our work leverages Bayesian filtering to train neural network reward models in active reward learning settings, where we focus primarily on parameter uncertainty instead of joint prediction uncertainty.

Subspace methods for neural networks. While there exists a vast literature on decreasing neural network size for efficient training and serving via architecture search (Elsken et al., 2019), quantization (Gholami et al., 2021), and pruning (Frankle & Carbin, 2022), we focus only on works that enable tractable inference in the reduced model. Specifically, there is growing evidence that the number of parameters required to fit a neural network is much smaller than its total parameter count; optimization and inference in the subspace spanned by these parameters offer not only computational efficiency, but also tractability of applying Bayesian methods for neural network training (Fort et al., 2020; Larsen et al., 2022). These parameters are found either as a subset of neural network parameters, or within a lower-dimensional subspace of the parameters.

Methods focusing on parameter subsets typically apply Bayesian methods such as Bayesian linear regression or variational inference to the last layer of the neural network, and point estimation methods like stochastic gradient descent (SGD) for intermediate layers (Snoek et al., 2015; Harrison et al., 2023; Brunzema et al., 2024). On the other hand, subspace methods typically constructs the low-dimensional subspaces via either random projection or singular value decomposition of SGD iterates of the full network; any inference or optimization technique such as sliced sampling (Izmailov et al., 2020) or SGD Li et al. (2018) can then be applied in the subspace in a tractable manner.

3 **PRELIMINARIES**

108

110

111

112

113

114

115

116

117

118

119 120

121

122

123

124

125

126 127 128

129 130

131

132

133 134

135

136

137 138

139

141

142

143

144 145

146

147 148 149

150

151

152

153

154

156

157

159

160

161

Preference-based reward modeling. We consider a Markov decision process (MDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma \rangle$ with state space \mathcal{S} , action space \mathcal{A} , transition function \mathcal{T} , reward function $r: \mathcal{S} \to \mathbb{R}$, and discount factor $\gamma \in [0,1)$. We assume access to a dataset of trajectories $\mathcal{D}^{traj} = \{\tau_1, \dots, \tau_N\}$, where each trajectory τ_i is a sequence of T steps $\tau_i = \{(s_{i,t}, a_{i,t}, s_{i,t+1})\}_{t=0}^{T-1}$, with each step consisting of state $s_t \in \mathcal{S}$, action $a_t \in \mathcal{A}$, and next-state $s_{t+1} \in \mathcal{S}$. In preference-based reward modeling, we do not assume access to a reward function. Instead, our task supervision comes from annotators who provide binary preference labels over pairwise trajectory comparisons queries, and the goal is to learn the annotator's reward function that informed their preference labels.

Formally, an annotator takes a trajectory pair query $Q_i = (\tau_a^i, \tau_b^i)$, and returns a preference label $y_i = \mathbb{1}(\tau_a^i \succ \tau_b^i) \in \{0,1\}$ according to their internal reward function r. Given a dataset of queries and responses $\mathcal{D} = \{Q_i, y_i\}_i$, a widely-used approach for preference learning is to approximate r with a parameterized reward model r_{θ} via maximum likelihood estimation, where the likelihood $p_{\theta}(y \mid \tau_a, \tau_b)$ is typically defined using the Bradley-Terry (BT) model (Bradley & Terry, 1952),

$$p_{\theta}(y \mid \tau_a, \tau_b) = p_{\theta}(\tau_a \succ \tau_b) = \frac{\exp(\beta \cdot \mathcal{R}(\tau_a))}{\exp(\beta \cdot \mathcal{R}(\tau_a)) + \exp(\beta \cdot \mathcal{R}(\tau_b))}.$$
 (1)

In particular, β is a temperature parameter that models noisily optimal behavior of an annotator, and $\mathcal{R}(\tau_i)$ is the return of trajectory τ_i where the per-timestep reward is computed using a neural network-based RM r_{θ} , i.e., $\mathcal{R}(\tau_i) = \sum_{t=0}^{T-1} r_{\theta}(s_{i,t})$ (Lee et al., 2021a). ¹

Information-theoretic active learning. We adopt the InfoGain acquisition function from Bıyık et al. (2022) for active preference-based reward learning, which assumes a distribution over RM parameters $p(\theta)$ such that, given a query-response pair (Q, y) the predictive distribution is given by $p(y \mid Q) = \mathbb{E}_{p(\theta)}[p(y \mid Q, \theta)]$. It selects the query Q_i that maximizes expected information gain on θ by maximizing the mutual information between a query's answer label y_i and θ :

$$Q_{i}^{*} = \underset{Q_{i}}{\operatorname{arg \, max}} I\left(\boldsymbol{\theta}; y_{i} \mid Q_{i}, \boldsymbol{b}^{i-1}\right)$$

$$= \underset{Q_{i}}{\operatorname{arg \, max}} H\left(y_{i} \mid Q_{i}, \boldsymbol{b}^{i-1}\right) - \mathbb{E}_{\boldsymbol{\theta}}\left[H(y_{i} \mid \boldsymbol{\theta}, Q_{i})\right]$$
(2a)
$$(2b)$$

$$= \underset{Q_i}{\operatorname{arg max}} H\left(y_i \mid Q_i, \boldsymbol{b}^{i-1}\right) - \mathbb{E}_{\boldsymbol{\theta}} \left[H(y_i \mid \boldsymbol{\theta}, Q_i)\right]$$
 (2b)

where I is the mutual information, H is the Shannon entropy (Cover & Thomas, 2006), and b^{i-1} $p(\theta \mid \mathcal{D}_{1:i-1})$ is the posterior distribution over RM parameters after learning from (i-1) queries.

¹This formalism extends to state or state-action RMs, and whole trajectories or partial trajectory segments. Our experiments use state-based RM and partial trajectories.

We approximate this acquisition function via sampling as follows:

$$Q_{i}^{*} \doteq \underset{Q_{i}}{\operatorname{arg \, max}} \ \frac{1}{M} \sum_{y_{i} \in \{0,1\}} \sum_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \left(P(y_{i} \mid Q_{i}, \boldsymbol{\theta}) \log_{2} \left(\frac{M \cdot P(y_{i} \mid Q_{i}, \boldsymbol{\theta})}{\sum_{\boldsymbol{\theta}' \in \boldsymbol{\Theta}} P(y_{i} \mid Q_{i}, \boldsymbol{\theta}')} \right) \right)$$
(3)

where Θ is the set of models sampled from the posterior b^{i-1} , and M is the number of drawn samples. This approximation is asymptotically equivalent to Eq. 2b as $M \to \infty$. Due to the necessity of sampling models from the posterior b^{i-1} , the work by Bıyık et al. (2020) has been limited to low-dimensional RMs, such as linear models. We now present our method, PreferenceEKF, which enables sampling of high-dimensional RMs, such as neural networks, that in turn allows us to scalably compute sampling-based acquisition functions like InfoGain to perform active learning.

4 METHOD

Sampling neural network models to approximate acquisition functions as in Eq. 3 can be expensive due to the high-dimensional parameter space of neural networks (Izmailov et al., 2021). Ensemble methods approximate this by training M independent models, which can be infeasible for large M and model sizes (Lakshminarayanan et al., 2016). We leverage the insight that neural networks are overparameterized and that solutions actually live in a much smaller subspace (Li et al., 2018), and perform posterior inference within this subspace. This allows us to sample an arbitrary number of models from a lower-dimensional posterior to approximate Eq. 3, without the overhead of training ensembles. We first show how to use extended Kalman filter (EKF), a widely used filtering algorithm, to train neural network reward models from preference data, then we show how to scale EKF to deep neural networks using subspace methods, as shown in Algorithm 1.

Extended Kalman filter for training neural networks. Using the formulation of sequential Bayesian inference, we perform posterior inference of neural network parameters from streaming data $\mathcal{D}_{1:i-1} = \{(Q_1, y_1), \dots, (Q_{i-1}, y_{i-1})\}$. Starting from some prior belief $\mathbf{b}^0 = p(\boldsymbol{\theta})$ on the parameters, our posterior after observing i samples can be expressed using Bayes' rule as follows:

$$p(\boldsymbol{\theta}_{i} \mid \mathcal{D}_{1:i}) \propto p(\mathcal{D}_{i} \mid \boldsymbol{\theta}_{i}) p(\boldsymbol{\theta}_{i} \mid \mathcal{D}_{1:i-1})$$

$$= \underbrace{p(\mathcal{D}_{i} \mid \boldsymbol{\theta}_{i})}_{\text{Measurement}} \underbrace{\int p(\boldsymbol{\theta}_{i} \mid \boldsymbol{\theta}_{i-1})}_{\text{Dynamics}} \underbrace{p(\boldsymbol{\theta}_{i-1} \mid \mathcal{D}_{1:i-1})}_{\text{Previous posterior}} d\boldsymbol{\theta}_{i-1}$$
(4)

where $p(\boldsymbol{\theta}_{i-1} \mid \mathcal{D}_{1:i-1})$ is the posterior belief over parameters after observing i-1 samples, which is combined with a parameter dynamics model and measurement model to form the posterior after observing the i^{th} example. This formulation naturally allows for a recursive estimation scheme where model parameters can be updated by observing samples one at a time. To make computing Eq. 4 tractable, we assume additive Gaussian noise for both the dynamics model $p(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}) = \mathcal{N}(\boldsymbol{\theta}_i \mid g(\boldsymbol{\theta}_{i-1}), \mathbf{U})$ and the measurement model $p(\mathcal{D}_i \mid \boldsymbol{\theta}_i) = \mathcal{N}(y_i \mid h(\boldsymbol{\theta}_i, Q_i), \mathbf{V})$, where $\mathbf{U} \in \mathbb{R}^{|\boldsymbol{\theta}| \times |\boldsymbol{\theta}|}$ and $\mathbf{V} \in \mathbb{R}^{|\boldsymbol{y}| \times |\boldsymbol{y}|}$ are prespecified Gaussian noise covariance matrices.

We treat neural network parameters as hidden states, and model the state dynamics $g: \mathbb{R}^{|\theta|} \to \mathbb{R}^{|\theta|}$ using an identity function. For preference learning, we model measurements $h: \mathbb{R}^{|\theta|} \times \mathbb{R}^{|Q|} \to \mathbb{R}^{|y|}$ using BT model $p_{\theta}(\tau_a \succ \tau_b)$ computed using the learned RM r_{θ} (Eq. 1). Assumptions on additive Gaussian noise and nonlinear dynamics and measurement functions make the neural network inference objective in Eq. 4 solvable in closed-form with the EKF algorithm, where the posterior takes a Gaussian form $b^i = \mathcal{N}(\mu_i, \Sigma_i)$ where $\mu_i \in \mathbb{R}^{|\theta|}$ and $\Sigma_i \in \mathbb{R}^{|\theta| \times |\theta|}$.

Subspace inference. Inference using EKF directly in the parameter space of a neural network is difficult, as the size of the covariance matrix Σ_i of the Gaussian posterior scales in $O(|\theta|^2)$. We instead perform EKF in a learned subspace of the NN: we denote the full space parameter as θ and subspace parameter as z, where $|z| \ll |\theta|$, resulting in posterior $b^i = \mathcal{N}(\mu_i', \Sigma_i')$ where $\mu_i' \in \mathbb{R}^{|z|}$ and $\Sigma_i' \in \mathbb{R}^{|z| \times |z|}$. We further assume an affine mapping $\theta(z) = \mathbf{A}z + \theta_*$ that allows us to transform between the subspace and the full space. Here θ_* is initialized via SGD on a small warm-up dataset in the full space. $\mathbf{A} \in \mathbb{R}^{|\theta| \times |z|}$ is a fixed projection matrix obtained from applying SVD to the SGD iterates ran in the full space, as shown on Line 8 through Line 10. 2

²The projection matrix can also be computed via random projections (Li et al., 2018), but we found that the SVD approach (Izmailov et al., 2020) led to better empirical performance. See Fig. 3b for an ablation.

We perform EKF inference in the subspace to obtain an estimate $\mathbf{b}^i = p(\mathbf{z} \mid \mathcal{D}_{1:i})$ after observing each query-response pair $\mathcal{D}_i = (Q_i, y_i)$, then project each model sampled from \mathbf{b}^i back to the full space via affine projection $\theta(\mathbf{z})$ to perform the forward pass of the neural network to predict $\mathbb{I}(\tau_a^i \succ \tau_b^i)$. Predictive distribution for computing InfoGain is similarly computed via sampling followed by projection as $p(y \mid Q) = \mathbb{E}_{p(\mathbf{z})}[p(y \mid Q, \mathbf{A}\mathbf{z} + \theta_*)]$. The belief update procedure is completely deterministic, with the only source of stochasticity coming from sampling of subspace parameters (followed by affine transformation) for computing the acquisition function and posterior predictive distribution.

Active learning using subspace inference. We refer to the ensemble-based approach as Deep-Ensemble and our approach as PreferenceEKF. We also assume a pool-based active learning setting (Settles, 2009) where we denote the set of all possible binary preference queries as \mathcal{P} . ³ For belief initialization (Line 12), whereas PreferenceEKF uses a zero-mean isotropic Gaussian of subspace dimension |z|, DeepEnsemble initializes M independent models each of dimension $|\theta|$.

After belief initialization, the sequential phase of active learning begin. For random querying, Line 14 amounts to simply retrieving a random query from the query pool \mathcal{P} , whereas active learning algorithms computes an acquisition function for the optimal query to retrieve from the pool. The algorithm then receives the corresponding label for the retrieved query from an annotator in Line 15. For belief update (Line 16), whereas PreferenceEKF performs filtering in the constructed subspace only on the most recent query-response pair \mathcal{D}_i , DeepEnsemble trains each of the M models using gradient descent on all data seen so far.

The most common uncertainty-based acquisition function is ensemble disagreement, i.e., pick the query Q_i for which the predicted preference label $\mathbbm{1}(\tau_a^i \succ \tau_b^i)$ has the highest variance across the ensemble. Disagreement has been popular for neural network-based active learning where it is expensive to scale Bayesian methods to high-dimensional settings (Christiano et al., 2017; Lee et al., 2021b), while InfoGain is the current state of the art acquisition function for lower-dimensional reward learning settings (Bıyık et al., 2020; 2024). While our method can be used to compute any sampling-based acquisition functions, we specifically leverage PreferenceEKF's ability to sample from high-dimensional distributions to scale InfoGain to neural network models. Due to the difficulty of sampling from high dimensional parameter distributions and the cost of training multiple models, DeepEnsemble approximates InfoGain by maintaining a small number of independent models. Dropout does so by sampling parameter dropout masks during inference.

Algorithm 1 PreferenceEKF for active preference-based reward learning

```
249
                1: Input:
250
                2: \mathcal{P}: Set of all possible binary preference queries without labels
251
                3: \mathcal{D}^{\text{init}} = \{(Q_i, y_i)\}_{i=1}^{\tau}: Initial preference dataset with \tau query-label pairs
252
                4: B: query budget limit
253
                5: w: number of SGD iterations for subspace construction
254
                6: Procedure:
255
                7: # Subspace Construction
                8: \boldsymbol{\theta}_{1:w} = \operatorname{SGD}(\mathcal{D}^{\operatorname{init}})
256
                                                                                                                              \triangleright Parameter offset: \theta_* \in \mathbb{R}^{|\theta|}
                9: \boldsymbol{\theta}_* = \boldsymbol{\theta}_w
257
              10: \mathbf{A} = \text{SVD}(\boldsymbol{\theta}_{1:w})
                                                                                                                        \triangleright Projection matrix: \mathbf{A} \in \mathbb{R}^{|\boldsymbol{\theta}| \times |\boldsymbol{z}|}
258
              11: # Subspace Inference
259
              12: b^0(z) = \mathcal{N}(\mu'_0, \Sigma'_0)
                                                                                                                           \triangleright Belief initialization: z \in \mathbb{R}^{|z|}
260
              13: for t = 1 : B do
261
                        Q_t = \text{ComputeQuery}(\boldsymbol{b}^{t-1}, \mathbf{A}, \boldsymbol{\theta}_*, \mathcal{P})
                                                                                                                      \triangleright Compute InfoGain: \theta(z) \in \mathbb{R}^{|\theta|}
262
                         y_t = \text{GetLabel}(Q_t)
263
                         \boldsymbol{b}^t = \text{EKF}(\boldsymbol{b}^{t-1}, (Q_t, y_t))
              16:
                                                                                                                                       \triangleright EKF update: z \in \mathbb{R}^{|z|}
264
              17: end for
265
```

³Given a dataset of N trajectories, there would be $|\mathcal{P}| = \binom{n}{2}$ possible pairwise comparison queries.

5 EXPERIMENTS

Baselines and Evaluation. We compare our PreferenceEKF method to both DeepEnsemble and Dropout, and answer the following questions in our experiments: (1) Does active learning with PreferenceEKF lead to more data-efficient and effective preference-based reward learning compared to DeepEnsemble and Dropout? (2) Can the reward models (RMs) sampled from PreferenceEKF's induced posterior be used to for policy optimization via offline RL? (3) Does representing parameter uncertainty $p(\theta \mid \mathcal{D})$ as a subspace distribution lead to computational advantages over representation using ensembles and dropout? We additionally study the model calibration quality of all algorithms, as well as ablate the subspace construction procedure for PreferenceEKF.

In the reward learning experiments, given a limited query budget B, we would like to learn RMs from preference queries as sample-efficiently as possible. Evaluation is done by comparing the Bradley-Terry log-likelihood achieved by a RM on a held-out set of test queries throughout training. To create the preference query pool \mathcal{P} , we randomly sample pairwise partial trajectories from a trajectory dataset \mathcal{D}^{traj} , then generate noisily optimal synthetic labels: for a given pair of trajectories, we compute their returns and sample a preference label according to the BT model (Eq. 1).

In the offline RL experiments, The learned RMs are then used for training parameterized policies $\pi_{\phi}(a \mid s)$ via offline RL. This is done by first labeling the trajectory dataset \mathcal{D}^{traj} with the learned RM: we take the average predicted reward over M models $r_{\theta}^{M}(s_{i,t}) = \frac{1}{M} \sum_{m=1}^{M} r_{\theta}^{m}(s_{i,t})$ for each state, where r_{θ}^{m} is the m^{th} sampled reward model for PreferenceEKF and Dropout, and the m^{th} model in the ensemble for DeepEnsemble. A reward-labeled trajectory takes the form, $\tau_{i} = \{(s_{i,t}, a_{i,t}, s_{i,t+1}, r_{\theta}^{M}(s_{i,t}))\}_{t=0}^{T-1}$, and we train policies on the reward-labeled \mathcal{D}^{traj} using Implicit Q-Learning (IQL) (Kostrikov et al., 2021), an empirically successful offline RL algorithm. We evaluate policies by comparing their empirical rollout returns throughout RL training.

Tasks. We focus on simulated control tasks. Specifically, we evaluate our approach in D4RL (Fu et al., 2020), a popular offline RL benchmark. We choose a mixture of environments spanning MuJoCo locomotion (HalfCheetah, Hopper, Walker2d), Adroit Shadow Hand (pen twirling), and Maze2D navigation. Within each environment, we choose trajectory datasets of varying characteristics: MuJoCo trajectories span a range of performance quality, Adroit trajectories are generated by a human operator and a fine-tuned expert-level RL policy, and maze navigation trajectories are collected from policies executed in mazes of varying difficulty. We consider each dataset as a separate task, for a total of 12 tasks.

Implementation Details. Unless otherwise stated, all experiments are done on a single node with 8 NVIDIA RTX A6000 GPUs via sharding, query budget B=60, and partial trajectory segments of length 50. On the belief update step (Line 16), PreferenceEKF learn from only the most recent query-label pair, while DeepEnsemble and Dropout learns from all data seen so far over multiple epochs. Since PreferenceEKF and Dropout can sample arbitrary number M of models, we set M=100; while DeepEnsemble uses a small number of models (M=5) as is commonly done for ensemble-based uncertainty quantification (Ovadia et al., 2019). Dropout method samples dropout masks with probability p=0.3. We refer to Appendix Section A.1 for more details

With the exception of the scaling experiments in Section 5.3 and the ablation experiments in Section 5.5, all reward models are represented as multi-layer perceptrons (MLP) with two hidden layers of 64 units, using subspace dimensionality |z|=200. All methods use the InfoGain acquisition function to ensure fair comparison. We show additional results using the disagreement acquisition function in Appendix Section A.1.1.

5.1 Does active learning with PreferenceEKF lead to sample efficient reward learning?

Given a fixed query budget per task, we evaluate each algorithm over 5 seeds. We use state-based partial trajectories, and compute return of each trajectory as $\mathcal{R}(\tau_i) = \sum_{t=1}^T r_{\theta}(s_{i,t})/T$. We show in Fig. 1a that aggregated over all tasks, active PreferenceEKF achieves higher sample efficiency compared to its random counterpart, requiring 42% fewer queries to reach the random variant's performance. Additionally, both random and active variants of PreferenceEKF outperforms the other two baselines along with their variants. In the appendix, we show in Fig. A.1 that in most

task, active PreferenceEKF outperforms both its random counterpart as well as all other algorithms in terms of sample efficiency and final log-likelihood. See also Section A.1.1 for reward learning results using the disagreement acquisition function. All results here are shown with a rolling average of 5 samples.

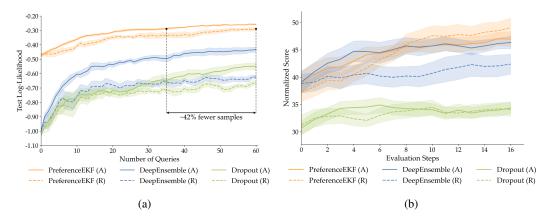


Figure 1: Fig. 1a shows comparison of the (R) andom and (A)ctive learning variants of the algorithms for preference-based reward modeling using the InfoGain acquisition function, aggregated over 12 D4RL tasks (mean±s.e. over 5 seeds). Fig. 1b shows comparison of the reward models learned using random and active variants of PreferenceEKF and DeepEnsemble, aggregated across 12 D4RL tasks in the offline RL setting (mean±s.e. over 5 seeds). See Fig. A.1 and Fig. A.5 in the appendix for per-task results for reward learning and offline RL evaluations results, respectively.

5.2 CAN PREFERENCEEKF REWARD MODELS BE USED FOR POLICY OPTIMIZATION VIA OFFLINE RL?

The goal of the offline RL experiments is to test whether an RM learned from limited number of preference queries can recover the ground-truth reward information, evaluated by whether the learned RM can induce a policy that reaches or exceeds the performance of a policy trained with ground-truth environment rewards. All policies are trained using IQL (Kostrikov et al., 2021) over 5 seeds on the reward-labeled dataset for 1M steps. Evaluation is done via 5 rollouts every 50K steps. In Fig. 1b, we show that when aggregated across all tasks, reward models learned with PreferenceEKF and active DeepEnsemble lead to more performant RL policies compared to other methods, while Dropout greatly underperforms the other methods.

While reward models learned from random variant of PreferenceEKF slightly outperforms that learned from active PreferenceEKF, we note that the final log-likelihood of both variants in Fig. 1a are very close to each other, leading to the observed similar policy optimization performance. For most tasks as shown in Appendix Section A.2 and Fig. A.5, policies trained from active variants of PreferenceEKF and DeepEnsemble reward models have performance on par with each other. All results here are shown with a rolling average of 5 samples.

5.3 Does training using PreferenceEKF scale better than using DeepEnsemble?

Next, we investigate whether extended Kalman filter can serve as a scalable alternative to gradient descent for training neural network reward models using preference data. Computing predictive distributions via sampling (Eq. 3) requires forward passes over M neural networks. We show here the computational advantage of PreferenceEKF in maintaining a subspace parameter posterior distribution $p(\theta \mid \mathcal{D})$ to sample models from, compared to DeepEnsemble's approach of maintaining and training M neural networks explicitly and Dropout's approach of sampling dropout masks. We run all scaling experiments on CPUs as the larger models and ensemble sizes led to out-of-memory errors. Finally, while PreferenceEKF's belief update procedure (Line 16) only requires the most recent query due to EKF being an online estimation algorithm, DeepEnsemble and Dropout train over

3 epochs on all queries observed so far. (Lee et al., 2021b; Christiano et al., 2017). All experiments use subspaces of fixed dimensionality |z| = 200.

We show in Fig. 2a that given a fixed architecture of a two-layer MLP with 64 units per layer, the runtime of PreferenceEKF for learning a reward model from B=60 queries scales much more gracefully with increasing M compared to DeepEnsemble. While Dropout does not need to maintain multiple independent models, it is still slower than PreferenceEKF as it performs model update in full parameter space instead of a lower-dimensional subspace. Fig. 2b demonstrates that final test log-likelihood favors PreferenceEKF over the other methods, showcasing that our approach maintains consistent performance on top of computational efficiency given increasing M.

Fig. 2c and Fig. 2d show that given fixed number of model parameter samples (M=5) and increasing neural network architecture size, PreferenceEKF scales more gracefully compared to other methods, on top of maintaining test log-likelihood performance. This showcases the scalability of subspace training to not only settings where we need large number of model samples M, but also to settings where we need larger neural networks $|\theta|$.

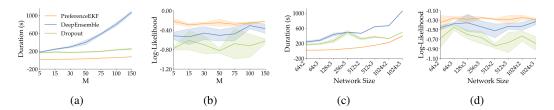


Figure 2: Fig. 2a and Fig. 2b show the effect of scaling the number of model samples M, while Fig. 2c and Fig. 2d show the effect of scaling neural network architecture size in the active learning setting (mean \pm std over 3 seeds). Overall, PreferenceEKF scales more gracefully than the other algorithms, showcasing the advantages of both subspace training and uncertainty representation using subspace distribution over model ensembles and dropout masks.

5.4 Does PreferenceEKF lead to better calibrated models?

While effective representation of parameter uncertainty is crucial for efficient active learning, it is also important for calibration of neural network predictions (Guo et al., 2017; Ovadia et al., 2019). We study whether uncertainty quantification (UQ) using subspace methods leads to better calibrated model predictions compared to UQ using ensemble methods and dropout, as quantified by two commonly used UQ metrics: expected calibration error (ECE) (Naeini et al., 2015; Pavlovic, 2025) and Brier score (Brier, 1950; DeGroot & Fienberg, 1983).

We show in Fig. 3a that active PreferenceEKF has the lowest ECE among all methods, and the lowest Brier score along with active DeepEnsemble, while Dropout consistently underperforms the other methods in these metrics. This highlights the superior quality of posterior approximation achieved by subspace methods. While dropout-based methods gets around the computational cost of ensembles, the resulting uncertainty representation has led to both poorer active learning and UQ results as compared to subspace methods.

5.5 ABLATION STUDY ON SUBSPACE CONSTRUCTION

The method for subspace construction for PreferenceEKF can be modified to 1) use varying dimensionality of the subspace, and to 2) use random projection to generate the subspace basis instead of running SVD on gradient descent iterates (Li et al., 2018; Izmailov et al., 2020). While all of our experiments so far use a fixed dimensionality of |z|=200 with SVD-based construction, we perform an ablation analysis over these choices, as shown in Fig. 3b. We observed that while the SVD-based approach works well for smaller subspace dimensions, the random projection approach can eventually reach performance on par with or even outperform the SVD approach as the subspace dimension increases. This finding is similar to what was observed by Duran-Martin et al. (2022) in bandit settings, highlighting the generality of this result for subspace Bayesian filtering methods used to train neural networks.

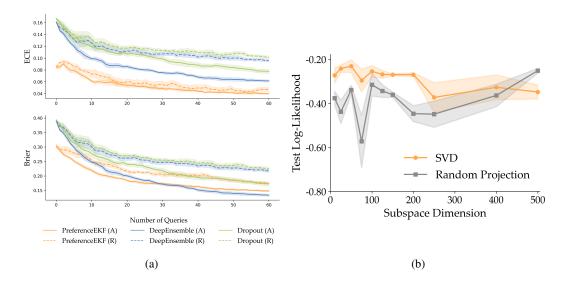


Figure 3: Fig. 3a shows comparison of the (R)andom and (A)ctive variants of the algorithms in model calibration, as evaluated by expected calibration error and Brier score on a test dataset (lower is better for both metrics). Active PreferenceEKF has the lowest overall calibration error, and has Brier score on par with DeepEnsemble. Dropout underperforms under methods in both evaluations. Fig. 3b shows an ablation over the subspace construction technique for PreferenceEKF, as evaluated by log-likelihood on a test dataset (higher is better). Both the UQ experiment and ablation analysis here are performed over 3 seeds (mean \pm std) on the Walker Medium Expert task.

6 Conclusion

In this work, we successfully adopted extended Kalman filters to train neural networks in active preference-based reward modeling setting. We showed several advantages of maintaining a subspace distribution over neural network parameters $p(\theta \mid \mathcal{D})$, in comparison to the widely used method of training multiple independent reward models as an ensemble or sampling dropout masks. Our approach led to more sample efficient active reward learning, more performant RL policy optimization, better runtime scaling with respect to model size and model sample count, and better calibration through higher-quality uncertainty representation. Learning a lower-dimensional distribution of neural network parameters further allowed us to scale the current state art of the art acquisition function for preference-based active reward modeling, InfoGain (B1y1k et al., 2020), from lower-dimensional model settings to deep neural networks.

Limitations and future work. While we found subspace method to be an effective tool for scaling Bayesian filtering methods for neural network training, it is unsure whether this approach will be effective for applying Bayesian methods to foundation model-scale reward models (Mahan et al., 2024; Zhang et al., 2024). Due to the unimodality of the Gaussian distribution that extended Kalman filter maintains, alternative methods may need to be investigated for approximating multimodal posteriors, e.g., learning reward functions from annotators with differing preferences (Poddar et al., 2024; Siththaranjan et al., 2023). We would further like to evaluate uncertainty quantification using the recent works on epistemic neural networks (Osband et al., 2023b), which focuses on joint predictions uncertainty instead of marginal predictive distribution.

While our work primarily focused on improving sample-efficiency of reward modeling in RLHF, we would like to further investigate how learned posterior distribution of reward models can aid in RL policy's exploration and serve as a mechanism for mitigating reward hacking (Yang et al., 2024; Gao et al., 2022; Hadfield-Menell et al., 2017). Finally, due to its sample-efficiency and adaptivity to non-stationary distributions, we believe the subspace filtering method to be a viable candidate for uncertainty quantification and large model finetuning in robot learning domains (Bellemare et al., 2017; Fridovich-Keil et al., 2020; Bobu et al., 2020).

7 REPRODUCIBILITY STATEMENT

Our code is anonymously available in the JAX framework at https://github.com/preferenceEKF2025/preference_ekf. We ensured that all pseudo-randomness has been controlled for via JAX's PRNG implementation. We provide all SLURM launch scripts, visualization scripts, and configuration files with all hyperparameters as part of code release.

REFERENCES

- Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Twenty-First International Conference on Machine Learning ICML '04*, pp. 1, Banff, Alberta, Canada, 2004. ACM Press. doi: 10.1145/1015330.1015430.
- Kim Baraka, Ifrah Idrees, Taylor Kessler Faulkner, Erdem Biyik, Serena Booth, Mohamed Chetouani, Daniel H Grollman, Akanksha Saran, Emmanuel Senft, Silvia Tulli, Anna-Lisa Vollmer, Antonio Andriella, Helen Beierling, Tiffany Horter, Jens Kober, Isaac Sheidlower, Matthew E Taylor, and Xuesu Xiao. Human-Interactive Robot Learning: Definition, Challenges, and Recommendations, 2025.
- B.M. Bell and F.W. Cathey. The iterated Kalman filter update as a Gauss-Newton method. *IEEE Transactions on Automatic Control*, 38(2):294–297, February 1993. ISSN 1558-2523. doi: 10. 1109/9.250476.
- Marc G. Bellemare, Will Dabney, and Rémi Munos. A Distributional Perspective on Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 449–458. PMLR, July 2017.
- Erdem Bıyık, Malayandi Palan, Nicholas C. Landolfi, Dylan P. Losey, and Dorsa Sadigh. Asking Easy Questions: A User-Friendly Approach to Active Reward Learning. In *Proceedings of the Conference on Robot Learning*, pp. 1177–1190. PMLR, May 2020.
- Erdem Bıyık, Dylan P. Losey, Malayandi Palan, Nicholas C. Landolfi, Gleb Shevchuk, and Dorsa Sadigh. Learning Reward Functions from Diverse Sources of Human Feedback: Optimally Integrating Demonstrations and Preferences. *The International Journal of Robotics Research*, 41(1): 45–67, January 2022. ISSN 0278-3649. doi: 10.1177/02783649211041652.
- Erdem Bıyık, Nicolas Huynh, Mykel J. Kochenderfer, and Dorsa Sadigh. Active Preference-Based Gaussian Process Regression for Reward Learning and Optimization. *The International Journal of Robotics Research*, 43(5):665–684, April 2024. ISSN 0278-3649. doi: 10.1177/02783649231208729.
- David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017. ISSN 0162-1459, 1537-274X. doi: 10.1080/01621459.2017.1285773.
- Andrea Bobu, Andrea Bajcsy, Jaime F. Fisac, Sampada Deglurkar, and Anca D. Dragan. Quantifying Hypothesis Space Misspecification in Learning From Human–Robot Demonstrations and Physical Corrections. *IEEE Transactions on Robotics*, 36(3):835–854, June 2020. ISSN 1941-0468. doi: 10.1109/TRO.2020.2971415.
- Ralph Allan Bradley and Milton E. Terry. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 0006-3444. doi: 10.2307/2334029.
- Glenn W. Brier. Verification of Forecasts Expressed in Terms of Probability. *Monthly Weather Review*, 78:1, January 1950. ISSN 0027-0644. doi: 10.1175/1520-0493(1950)078 (0001: VOFEIT) 2.0.CO; 2.
- Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating Beyond Suboptimal Demonstrations via Inverse Reinforcement Learning from Observations. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 783–792. PMLR, May 2019.

Paul Brunzema, Mikkel Jordahn, John Willes, Sebastian Trimpe, Jasper Snoek, and James Harrison. Bayesian Optimization via Continual Variational Last Layer Training. In *The Thirteenth International Conference on Learning Representations*, October 2024.

- Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, Tony Wang, Samuel Marks, Charbel-Raphaël Segerie, Micah Carroll, Andi Peng, Phillip Christoffersen, Mehul Damani, Stewart Slocum, Usman Anwar, Anand Siththaranjan, Max Nadeau, Eric J. Michaud, Jacob Pfau, Dmitrii Krasheninnikov, Xin Chen, Lauro Langosco, Peter Hase, Erdem Bıyık, Anca Dragan, David Krueger, Dorsa Sadigh, and Dylan Hadfield-Menell. Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback, July 2023.
- Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- T. M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, Hoboken, N.J, 2nd ed edition, 2006. ISBN 978-0-471-24195-9.
- Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux effortless Bayesian deep learning. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, pp. 20089–20103, Red Hook, NY, USA, June 2024. Curran Associates Inc. ISBN 978-1-7138-4539-3.
- J.F.G. de Freitas, M. Niranjan, and A. H. Gee. Hierarchical bayesian models for regularisation in sequential learning. *Neural Computation*, 12(4):933–953, April 2000. ISSN 1530-888X. doi: 10.1162/089976600300015655.
- DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020.
- Morris H. DeGroot and Stephen E. Fienberg. The Comparison and Evaluation of Forecasters. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 32(1):12–22, 1983. ISSN 0039-0526. doi: 10.2307/2987588.
- Thomas G. Dietterich. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*, pp. 1–15, Berlin, Heidelberg, 2000. Springer. ISBN 978-3-540-45014-6. doi: 10.1007/3-540-45014-9-1.
- Gerardo Duran-Martin, Aleyna Kara, and Kevin Murphy. Efficient Online Bayesian Inference for Neural Bandits. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pp. 6002–6021. PMLR, May 2022.
- Bradley Efron. Bootstrap Methods: Another Look at the Jackknife. In Samuel Kotz and Norman L. Johnson (eds.), *Breakthroughs in Statistics: Methodology and Distribution*, pp. 569–593. Springer, New York, NY, 1992. ISBN 978-1-4612-4380-9. doi: 10.1007/978-1-4612-4380-9_41.
- Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural Architecture Search: A Survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019. ISSN 1533-7928.
- Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided Cost Learning: Deep Inverse Optimal Control Via Policy Optimization. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning Volume 48*, ICML'16, pp. 49–58, New York, NY, USA, June 2016. JMLR.org.
- Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep Ensembles: A Loss Landscape Perspective, June 2020.

- Jonathan Frankle and Michael Carbin. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *International Conference on Learning Representations*, February 2022.
 - David Fridovich-Keil, Andrea Bajcsy, Jaime F Fisac, Sylvia L Herbert, Steven Wang, Anca D Dragan, and Claire J Tomlin. Confidence-Aware Motion Prediction for Real-Time Collision Avoidance. *The International Journal of Robotics Research*, 39(2-3):250–265, March 2020. ISSN 0278-3649. doi: 10.1177/0278364919859436.
 - Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for Deep Data-Driven Reinforcement Learning, April 2020.
 - Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1050–1059. PMLR, June 2016.
 - Leo Gao, John Schulman, and Jacob Hilton. Scaling Laws for Reward Model Overoptimization. In *International Conference on Machine Learning*, October 2022.
 - Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A Survey of Quantization Methods for Efficient Neural Network Inference, June 2021.
 - Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1321–1330. PMLR, July 2017.
 - Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart Russell, and Anca D. Dragan. Inverse Reward Design. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6768–6777, Red Hook, NY, USA, December 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4.
 - James Harrison, John Willes, and Jasper Snoek. Variational Bayesian Last Layers. In *The Twelfth International Conference on Learning Representations*, October 2023.
 - Philipp Hennig and Christian J. Schuler. Entropy Search for Information-Efficient Global Optimization. *Journal of Machine Learning Research*, 13(57):1809–1837, 2012. ISSN 1533-7928.
 - José Miguel Hernández-Lobato, Matthew W. Hoffman, and Zoubin Ghahramani. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.
 - Jonathan Ho and Stefano Ermon. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.
 - Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian Active Learning for Classification and Preference Learning, December 2011.
 - Jiri Hron, Alex Matthews, and Zoubin Ghahramani. Variational Bayesian dropout: Pitfalls and fixes. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 2019–2028. PMLR, July 2018.
 - Pavel Izmailov, Wesley J. Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Subspace Inference for Bayesian Deep Learning. In *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, pp. 1169–1179. PMLR, August 2020.
 - Pavel Izmailov, Sharad Vikram, Matthew D. Hoffman, and Andrew Gordon Gordon Wilson. What Are Bayesian Neural Network Posteriors Really Like? In *Proceedings of the 38th International Conference on Machine Learning*, pp. 4629–4640. PMLR, July 2021.
 - Matthew Thomas Jackson, Uljad Berdica, Jarek Liesen, Shimon Whiteson, and Jakob Nicolaus Foerster. A Clean Slate for Offline Reinforcement Learning, April 2025.
 - Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *CoRR*, December 2014.

- Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline Reinforcement Learning with Implicit Q-Learning. In *International Conference on Learning Representations*, October 2021.
- Aviral Kumar, Joey Hong, Anikait Singh, and Sergey Levine. Should I Run Offline Reinforcement Learning or Behavioral Cloning? In *International Conference on Learning Representations*, October 2021.
- Balaji Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Neural Information Processing Systems*, December 2016.
- Brett W. Larsen, Stanislav Fort, Nic Becker, and Surya Ganguli. How many degrees of freedom do we need to train deep networks: A loss landscape perspective, February 2022.
- Kimin Lee, Laura Smith, Anca Dragan, and Pieter Abbeel. B-Pref: Benchmarking Preference-Based Reinforcement Learning. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 1, December 2021a.
- Kimin Lee, Laura M. Smith, and Pieter Abbeel. PEBBLE: Feedback-Efficient Interactive Reinforcement Learning via Relabeling Experience and Unsupervised Pre-training. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 6152–6163. PMLR, July 2021b.
- Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems, May 2020.
- Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the Intrinsic Dimension of Objective Landscapes. In *International Conference on Learning Representations*, February 2018.
- Scott W. Linderman, Peter Chang, Giles Harper-Donnelly, Aleyna Kara, Xinglong Li, Gerardo Duran-Martin, and Kevin Murphy. Dynamax: A Python package for probabilistic state space modeling with JAX. *Journal of Open Source Software*, 10(108):7069, April 2025. ISSN 2475-9066. doi: 10.21105/joss.07069.
- D. V. Lindley. On a Measure of the Information Provided by an Experiment. The Annals of Mathematical Statistics, 27(4):986–1005, December 1956. ISSN 0003-4851, 2168-8990. doi: 10.1214/aoms/1177728069.
- David J. C. MacKay. Information-Based Objective Functions for Active Data Selection. *Neural Computation*, 4(4):590–604, July 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.4.590.
- Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative Reward Models, October 2024.
- Kevin P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. The MIT Press, Cambridge, Massachusetts, 2023. ISBN 978-0-262-04843-9.
- Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining Well Calibrated Probabilities Using Bayesian Binning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1), February 2015. ISSN 2374-3468. doi: 10.1609/aaai.v29i1.9602.
- Radford M. Neal. MCMC Using Hamiltonian Dynamics. In *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, 2011.
- Ian Osband. Risk versus Uncertainty in Deep Learning: Bayes, Bootstrap and the Dangers of Dropout. *NIPS Workshop on Bayesian Deep Learning*, 192, 2016.
- Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep Exploration via Bootstrapped DQN. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.
- Ian Osband, John Aslanides, and Albin Cassirer. Randomized Prior Functions for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

- Ian Osband, Seyed Mohammad Asghari, Benjamin Van Roy, Nat McAleese, John Aslanides, and Geoffrey Irving. Fine-Tuning Language Models via Epistemic Neural Networks, May 2023a.
- Ian Osband, Zheng Wen, Seyed Mohammad Asghari, Vikranth Dwaracherla, Morteza Ibrahimi, Xiuyuan Lu, and Benjamin Van Roy. Epistemic Neural Networks. In *Thirty-Seventh Conference on Neural Information Processing Systems*, November 2023b.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, December 2022.
- Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 14003–14014, Red Hook, NY, USA, December 2019. Curran Associates Inc.
- Theodore Papamarkou, Maria Skoularidou, Konstantina Palla, Laurence Aitchison, Julyan Arbel, David Dunson, Maurizio Filippone, Vincent Fortuin, Philipp Hennig, José Miguel Hernández-Lobato, Aliaksandr Hubin, Alexander Immer, Theofanis Karaletsos, Mohammad Emtiyaz Khan, Agustinus Kristiadi, Yingzhen Li, Stephan Mandt, Christopher Nemeth, Michael A. Osborne, Tim G. J. Rudner, David Rügamer, Yee Whye Teh, Max Welling, Andrew Gordon Wilson, and Ruqi Zhang. Position: Bayesian Deep Learning is Needed in the Age of Large-Scale AI, June 2024.
- Maja Pavlovic. Understanding Model Calibration A gentle introduction and visual exploration of calibration and the expected calibration error (ECE), March 2025.
- Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. Personalizing Reinforcement Learning from Human Feedback with Variational Preference Learning, August 2024.
- Dorsa Sadigh, Anca Dragan, Shankar Sastry, and Sanjit Seshia. Active Preference-Based Learning of Reward Functions. In *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation, July 2017. ISBN 978-0-9923747-3-0. doi: 10.15607/RSS.2017.XIII.053.
- Simo Särkkä and Lennart Svensson. *Bayesian Filtering and Smoothing*. Cambridge University Press, June 2023. ISBN 978-1-108-92664-5.
- Burr Settles. Active Learning Literature Survey. Technical Report, University of Wisconsin-Madison Department of Computer Sciences, 2009.
- Yuesong Shen, Nico Daheim, Bai Cong, Peter Nickl, Gian Maria Marconi, Bazan Clement Emile Marcel Raoul, Rio Yokota, Iryna Gurevych, Daniel Cremers, Mohammad Emtiyaz Khan, and Thomas Möllenhoff. Variational Learning is Effective for Large Deep Networks. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 44665–44686. PMLR, July 2024.
- Daniel Shin, Anca Dragan, and Daniel S. Brown. Benchmarks and Algorithms for Offline Preference-Based Reward Learning. *Transactions on Machine Learning Research*, September 2022. ISSN 2835-8856.
- Sharad Singhal and Lance Wu. Training Multilayer Perceptrons with the Extended Kalman Algorithm. In *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988.
- Anand Siththaranjan, Cassidy Laidlaw, and Dylan Hadfield-Menell. Distributional Preference Learning: Understanding and Accounting for Hidden Context in RLHF. In *The Twelfth International Conference on Learning Representations*, October 2023.

- Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable Bayesian Optimization Using Deep Neural Networks. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 2171–2180. PMLR, June 2015.
 - Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. ISSN 1533-7928.
 - Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, August 2005. ISBN 978-0-262-20162-9.
 - Kevin Tran, Willie Neiswanger, Junwoong Yoon, Qingyang Zhang, Eric Xing, and Zachary W Ulissi. Methods for comparing uncertainty quantifications for material property predictions. *Machine Learning: Science and Technology*, 1(2):025006, May 2020. ISSN 2632-2153. doi: 10.1088/2632-2153/ab7e1a.
 - Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A Survey of Preference-Based Reinforcement Learning Methods. *The Journal of Machine Learning Research*, 18(1): 4945–4990, January 2017. ISSN 1532-4435.
 - Adam X. Yang, Maxime Robeyns, Thomas Coste, Zhengyan Shi, Jun Wang, Haitham Bou-Ammar, and Laurence Aitchison. Bayesian Reward Models for LLM Alignment, July 2024.
 - Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative Verifiers: Reward Modeling as Next-Token Prediction. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*, October 2024.

A TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

We use the Dynamax package (Linderman et al., 2025) for implementations of extended Kalman filtering (EKF) (Thrun et al., 2005; Särkkä & Svensson, 2023) for reward learning, and the Unifloral package (Jackson et al., 2025) for implementation of implicit Q-learning (IQL) (Kostrikov et al., 2021) for offline reinforcement learning. Unless stated otherwise, all experiments are done on a single node with 8 NVIDIA RTX A6000 GPUs via SLURM sharding.

A.1 PREFERENCE-BASED REWARD LEARNING

Implementation Details. Unless otherwise stated, all reward learning experiments are done using subspace dimensionality |z|=200, query budget B=60, and partial trajectory of length 50. All neural networks reward model are represented using multi-layer perceptrons (MLP) with two hidden layers of 64 units. We apply normalization to all input features. PreferenceEKF and Dropout uses M=100 model parameter samples to compute the acquisition function and posterior predictive distribution, while DeepEnsemble trains M=5 independent networks, each with different weight initialization and randomness for minibatch shuffling.

All tasks use a pool of 150K pairwise partial trajectory queries drawn from the trajectory dataset to perform random or active querying over, and 3000 test queries for log-likelihood evaluation. For generation of noisy-optimal synthetic labels, we apply trajectory return normalization before passing trajectory pairs through the BT model (Eq. 1) to compute the likelihood $p_{\theta}(\tau_a \succ \tau_b)$. We use temperature parameter of $\beta=7$, resulting in roughly 5-15% mistaken preference labels per task.

Before the sequential learning phase starting on Line 13, all algorithms receive a small dataset consisting of $\tau=8$ query-response pairs for belief initialization, i.e., all algorithms observe a total of $\tau+B=8+60=68$ samples. All algorithms run variants of gradient descent (GD) on the warm-up dataset for 420 optimizer steps. While PreferenceEKF uses SGD with learning rate of 1e-4, momentum of 0.9, and batch size of 1, DeepEnsemble and Dropout uses Adam (Kingma & Ba, 2014) with learning rate of 1e-4 along with default hyperparamters from Optax (DeepMind et al., 2020), and batch size of 8. Previous works have found that SGD with a high constant learning is crucial to producing GD iterates with enough variance to construct a subspace effective for optimization and inference (Fort et al., 2020), hence the different choice of optimizer for PreferenceEKF.

PreferenceEKF constructs the subspace by running SVD on the GD iterates obtained from running SGD on the warmup dataset. We throw away the first 20 out of the 420 GD iterates and keep only every other remaining iterate, for a total of (420-20)/2=200 iterates. Thus, SVD takes in a model parameter array of shape $(200\times|\pmb{\theta}|)$, and return a projection matrix $\bf A$ of shape $(200\times|\pmb{z}|)$ by keeping only the top $|\pmb{z}|=200$ principal components. The final GD iterate is used as the full space parameter offset $\pmb{\theta}_*$, which, along with projection matrix $\bf A$, is used to transform from the subspace back up to the full space for, e.g. computing predictive distributions as described in Section 4. Finally, PreferenceEKF performs belief initialization (Line 12) in the subspace using a zero-mean isotropic Gaussian of dimension $|\pmb{z}|=200$.

On the belief update step (Line 16), PreferenceEKF learns from only the most recent query-label pair, while DeepEnsemble and Dropout learns learns from all data seen so far over 3 epochs. Note that the specific filtering algorithm we use is the iterated EKF (Bell & Cathey, 1993), which repeatedly re-linearize the measurement model around the estimated posterior. Empirically, we observed better log-likelihood evaluation performance in exchange for marginally extra runtime. We refer to the number of such re-linearization steps on every new sample as $n_{\rm linearize}$. For further details on iterated EKF, refer to Section 8.3.2.2 of (Murphy, 2023). We use $n_{\rm linearize} = 5$, prior noise of 0.07, systems noise of 1e-3, and measurement noise of 0.07 for all of our PreferenceEKF experiments.

A.1.1 ACQUISITION FUNCTIONS

The InfoGain acquisition function introduced in Eq. 2a was developed by Bıyık et al. (2020) for active reward learning using linear reward models. To motivate its origin, we first express the InfoGain objective in three equivalent forms below due to symmetry of mutual information. We refer to Section 5 of Bıyık et al. (2020) for further interpretations of the objective, and Appendix 9.1 of their

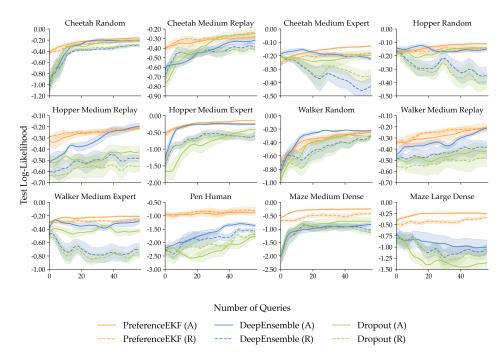


Figure A.1: Comparison of the (R)andom and (A)ctive variants of the algorithms using the InfoGain acquisition function, across 12 D4RL tasks for preference-based reward modeling (mean±s.e. over 5 seeds). In all tasks, active PreferenceEKF either performs on par with or outperforms other algorithms in terms of sample-efficiency and final log-likelihood.

work for derivation of the sampling-based approximation shown in Eq. 3.

$$Q_i^* = \underset{Q_i}{\operatorname{arg\,max}} I\left(\boldsymbol{\theta}; y_i \mid Q_i, \boldsymbol{b}^{i-1}\right)$$
 (5a)

$$= \operatorname*{arg\,max}_{Q_i} H\left(\boldsymbol{\theta} \mid Q_i, \boldsymbol{b}^{i-1}\right) - \mathbb{E}_{y_i} \left[H(\boldsymbol{\theta} \mid y_i, Q_i, \boldsymbol{b}^{i-1})\right]$$
 (5b)

$$= \underset{Q_i}{\operatorname{arg\,max}} H\left(y_i \mid Q_i, \boldsymbol{b}^{i-1}\right) - \mathbb{E}_{\boldsymbol{\theta}} \left[H(y_i \mid \boldsymbol{\theta}, Q_i)\right]$$
 (5c)

The idea of mutual information-based acquisition functions is rooted in the concept of expected information gain studied in Bayesian optimal experiment design and active data selection (MacKay, 1992; Lindley, 1956). It was later extended to Bayesian optimization using Gaussian process models under the methods Bayesian active learning by disagreement (BALD) (Houlsby et al., 2011), entropy search (ES) (Hennig & Schuler, 2012), and predictive entropy search (PES) (Hernández-Lobato et al., 2014). In particular, the mutual information objective function in Eq. 5a is expressed in its ES form in Eq. 5b, and expressed in its equivalent but computationally efficient PES form in Eq. 5c. Our PreferenceEKF method focuses on efficient sampling of high-dimensional neural network model parameters to approximate the predictive distribution for optimizing Eq. 5c.

Although our main experiments all use the InfoGain acquisition function to showcase the advantage of being able to sample from high-dimensional neural network parameter distributions, the PreferenceEKF method is agnostic to the acquisition function used for active learning. While Fig. 1a and Fig. A.1 showcase the aggregate and per-task log-likelihood results for active preference-based reward learning experiments using InfoGain, here we show additional results using the widely used disagreement acquisition function, which selects the query Q_i for which the predicted preference label $\mathbbm{1}(\tau_a^i \succ \tau_b^i)$ has the highest variance across the ensemble or sampled models. Fig. A.2 and Fig. A.3 show the aggregate and per-task log-likelihood results, while Fig. A.2 show the calibration results. Overall, we see that InfoGain led to superior active reward learning performance compared to disagreement.

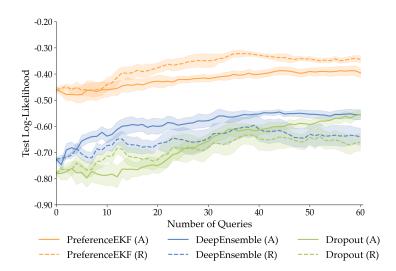


Figure A.2: Comparison of the (R)andom and (A)ctive variants of the algorithms for preference-based reward modeling using the disagreement acquisition function, aggregated over 12 D4RL tasks (mean±s.e. over 5 seeds). While PreferenceEKF outperforms all other methods, its active variant did not outperform its random variant.

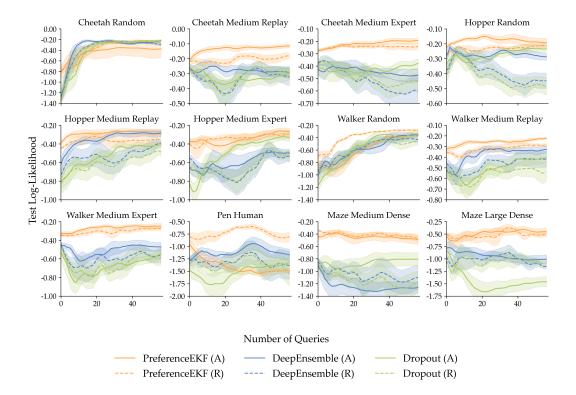


Figure A.3: Comparison of the (R) andom and (A) ctive variants of the algorithms using the disagreement acquisition function, across 12 D4RL tasks for preference-based reward modeling (mean±s.e. over 5 seeds). In most tasks, active PreferenceEKF either performs on par with or outperforms other algorithms in terms of sample-efficiency and final log-likelihood. Pen Human is a particular outlier case where active PreferenceEKF severely underperforms, which explains why the aggregate results in Fig. A.2 look unfavorably for PreferenceEKF.

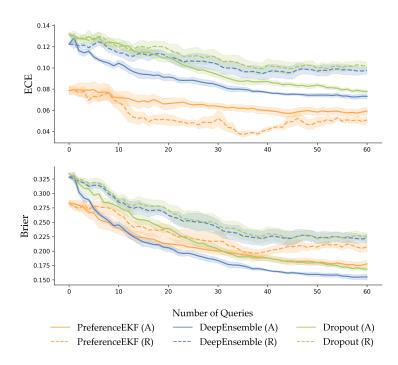


Figure A.4: Comparison of the (R)andom and (A)ctive variants of the algorithms in model calibration, as evaluated by expected calibration error and Brier score on a test dataset (lower is better for both metrics). While PreferenceEKF dominates in ECE, DeepEnsemble outperformed all other methods in Brier score.

A.2 OFFLINE REINFORCEMENT LEARNING

The extent to which offline RL algorithms leverages reward information for policy optimization, i.e., whether reward-induced policy performance is a good metric for assessing learned reward models, is heavily dependent on the trajectory dataset: when ran on datasets consisting solely of expert demonstrations, offline RL algorithms will largely ignore reward information and adopt a behavioral cloning-like learning strategy. On the other hand, it is generally difficult to train a policy from a dataset consisting of purely random behavior (Kumar et al., 2021).

Following the experiment methodology of Shin et al. (2022) for our offline RL experiments, we add two reference performance scores to every task as shown in Fig. A.5: we refer to "GT" as the score from an offline RL policy trained on \mathcal{D}^{traj} labeled with ground-truth environment reward information, and "Zero" as score from a policy trained on \mathcal{D}^{traj} with reward information zeroed out. This serves to test whether an offline RL algorithm is able to effectively leverage reward information for a given trajectory dataset. For most tasks, GT and Zero serve as upper and lower performance bounds for learned policies.

All offline RL experiments were done by running implicit Q-learning (IQL) (Kostrikov et al., 2021) on trajectory transition datasets labeled with different types of rewards, e.g., ground truth environment reward, zeroed out reward, or preference-learned reward. An IQL agent consists of four neural networks: main and target Q-network, Gaussian policy network, and state-value network. All four networks have two hidden layers of 256 units each and are trained using the same optimizer configuration with cosine decay learning rate schedule. All training runs are done using 1M update steps with 5 rollouts every 50K steps for evaluation. We apply normalization to both reward and observation features, and further apply clipping for reward values exceeding 10. All hyperparameters are detailed in Table 1.

Table 1: Shared hyperparameters for IQL across all tasks. Here "Iterations" refers to the number of minibatch updates.

Name	Value
Optimizer	Adam
Learning rate	0.0003
Betas	(0.9, 0.999)
Iterations	1M
Batch size	256
Discount factor γ	0.99
Target net update step size	0.005
Expectile $ au$	0.7
Advantage temperature β	3.0
Exponential advantage clip	100

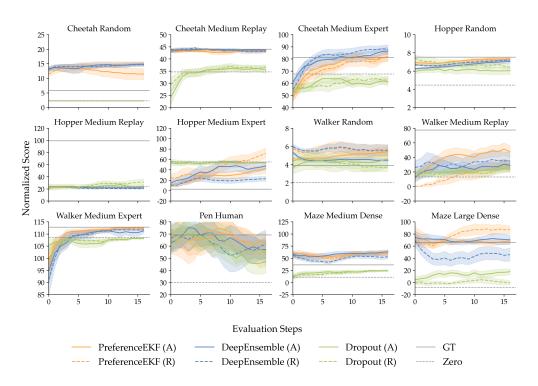


Figure A.5: Comparison of the RM learned using (R)andom and (A)ctive variants of the algorithms across 12 D4RL tasks in the offline RL setting (mean±s.e. over 5 seeds). Black solid line indicates the performance of a policy trained on ground truth reward (GT), and black dotted line for a policy trained without reward information (Zero). In most tasks, active PreferenceEKF performs on par with other algorithms in terms of rollout score.

A.3 SCALING EXPERIMENTS.

JAX offers efficient vectorization of arbitrary functions using <code>jax.vmap</code>. While we use this to parallelize ensemble model training and prediction in most experiments in Section 5, we do not use this for the scalability experiments in Section 5.3. Parallelized training and prediction of up to M=150 models with up to 2M parameters (in the case of the three layer neural networks with 1024 units each) can quickly lead to out-of-memory errors. We instead use python's native for loop to perform ensemble model training and prediction sequentially. All scalability experiments were done on CPU instead of GPU to avoid out-of-memory errors.

A.4 LLM USAGE

We used LLM exclusively for writing visualization scripts and for finding related work on subspace construction methods. We did not use LLM for paper writing or research ideation.