Improving Reward Learning by Estimating Annotator Expertise

Pavel Czempin[1], Rachel Freedman[2], Ellen Novoseller[3], Vernon J. Lawhern[3], Cameron Allen[2], Erdem Bıyık[1]
[1] Thomas Lord Department of Computer Science, University of Southern California
[2] Center for Human-Compatible Artificial Intelligence, UC Berkeley
[3] DEVCOM Army Research Laboratory
Correspondence: {czempin, biyik}@usc.edu

***Abstract*—Reinforcement learning from human feedback (RLHF) has been used successfully to teach robots tasks that are difficult to specify procedurally. However, feedback from human annotators can be suboptimal and noisy, decreasing accuracy and leading to potentially unsafe behavior. Furthermore, different human annotators may have varying context-dependent expertise. In this work, we study the feasibility of learning annotator expertise jointly with a reward model based on annotator feedback. As opposed to prior works that assume human annotators are perfect or that their expertise levels are known, our method performs RLHF training without these assumptions by estimating the expertise of every annotator given information about annotator identities in the data. We show that if annotators exhibit varying degrees of expertise, estimating annotator expertise improves the ranking accuracy of the learned reward functions. When the annotator's expertise depends on the *context*, our method shows limited success.**

I. Introduction

For many tasks, such as robotics and natural language processing, it is infeasible to procedurally specify an objective function that can be optimized by reinforcement learning (RL). A promising alternative to specifying an objective function is to *learn* a reward function from pairwise human preference comparisons, which is often called reinforcement learning from human feedback (RLHF). RLHF has achieved considerable success in domains such as natural language processing [28, 37] and control [15, 32, 7, 24].

Current RLHF approaches, for example, those that utilize the Bradley-Terry model [15], frequently assume that human annotators are noisily optimal and that the noise parameters are known and uniform across annotators. In practice, data collection is often outsourced and collected from a diverse set of annotators with potentially varying expertise levels. We study the feasibility of *jointly* learning a reward model in conjunction with a model of an *annotator's expertise*.

In particular, we posit that the quality of human evaluations varies not only *between annotators*, but also depends on the *context* of a preference query. For example, a human annotator who evaluates a cooking task might not be proficient in certain recipes or ingredients. As another example, if comparative labels are collected to improve a coding agent, a human's expertise might differ depending on the programming language or the task at hand. Learning these variations in expertise and incorporating them into the training process allows the appropriate use of cheap crowd-sourced data.

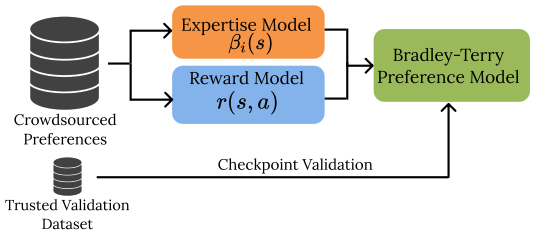


Fig. 1: **Method Overview.** We jointly learn a **reward model** and an **expertise model**, which depend on a) the annotator and b) a *context* derived from the state. The primary RLHF training is done on a large-scale dataset collected from multiple annotators with varying levels of expertise. To help distinguish between noise caused by a lack of expertise and noise that is inherent to the ground-truth reward function, we use a validation dataset of trusted human labels that makes up 10% of our data to select model checkpoints and avoid overfitting.

In this work, we study estimating annotator expertise by expressing an expertise model as a parameterized function and adding the learnable parameters to regular RLHF training. Our method learns an expertise model in an unsupervised way and uses a small validation set of trusted human data to aid in model selection. Our contributions are as follows.

- We propose a method (see Fig. 1) that improves the accuracy of a learned reward function trained with noisy annotators that have varying levels of expertise.
- We evaluate the algorithm's performance in grid-world environments relative to a baseline that does not model annotator expertise, and demonstrate that modeling expertise improves performance when annotators have differing expertise from one another, with some success when the expertise is also context-dependent.

II. Related Work

To reduce risk to expensive hardware and improve the feasibility of learning control policies under an unknown objective function, algorithms often use human demonstrations, e.g., in imitation learning [IL; 30, 2, 21, 23, 1] and inverse reinforcement learning [IRL; 27]. When the human demonstrations include suboptimal data, prior work has shown

that modeling human demonstrator suboptimality improves learning performance [12, 38, 42, 6, 25, 36, 9, 5].

These methods can usually be improved upon by explicitly asking annotators to rate the quality of the demonstrations. Consequently, offline preference learning is a popular approach for control tasks [20, 19, 22]. Confidence-aware imitation learning [42] bridges preference learning and IL by modeling suboptimality of demonstrations as opposed to learning a full reward function. However, while modeling varying levels of quality in demonstrations, these approaches treat human preference rankings as coming from the same rationality model. In our work, while we also focus on an offline preference learning setting, instead, we explicitly consider a setting where the expertise among evaluators varies in addition to the demonstrators.

ILEED [6] is an IL approach to learn from suboptimal demonstrations and its successor IRLEED [5] is a similar approach for inverse reinforcement learning. The authors show that it is possible to *jointly* optimize a policy and estimate demonstrators' expertise in an unsupervised way. Inspired by ILEED, we propose a method of jointly estimating expertise that instead focuses on the RLHF setting.

Several prior works extend traditional RLHF by accounting for multiple annotators with varying rationalities. Daniels-Koch and Freedman [16], Barnett et al. [4], and Freedman et al. [17] focus on leveraging annotator expertise to select annotators in a beneficial manner. These methods either assume that the true expertise parameters are known or require access to exact preference probabilities instead of binary preference labels. In contrast, our work estimates the expertise parameters from binary preference labels during reward model training.

Various methods focus on explicitly learning differing human expertise levels. Crowd-PrefRL [14] learns the rationality values for a crowd of annotators. Their approach assumes that each annotator in the crowd labels every example from the dataset and utilizes techniques from unsupervised crowdsourcing to learn annotator reliabilities. This incurs significant human labeling costs and assumes multiple (not necessarily all) annotators are giving feedback noisily according to the same objective. Similarly, Zhang and Kashima [41] estimate user reliabilities, but require multiple annotators to label the same preference query. In contrast, our method works in settings where each sample is labeled by a single annotator, which is common in practice as it incurs a lower annotator burden. Singhal et al. [33] also consider unreliable preference feedback. Unlike our approach, their method relies on external data to estimate expertise, e.g., asking an LLM about the difficulty of a given query. Yamagata et al. [39] also learn annotator rationality values jointly with rewards. Their setting is limited to annotators with constant rationality, whereas rationality can vary across different contexts in reality.

III. Problem Setting

We consider an RLHF setting in which a reward function must be learned from a large offline dataset of pairwise preference comparisons. As in many RLHF settings, the reward function aims to capture an intended behavior, and the preference comparisons are assumed to be given with respect to this underlying reward. Such an approach is commonly used, for example, to fine-tune large language models [44, 35, 28, 37], or to train robotic behavior that is difficult to procedurally formulate as an objective function [15, 32, 7, 24]. Labels over binary choices enable learning an objective function with a low burden on the human annotator. We consider crowd-sourced data that is noisy, such that not all labels perfectly align with the intended behavior, and furthermore, that varies across both annotators and queries.

As in most RLHF setups, the preference labels are collected from multiple annotators, for example, by crowd-sourcing. Additionally, we collect an ID for every annotator during this process, to identify which annotator labeled each preference query. Importantly, while our approach requires annotator IDs at training time, they are not needed at inference time, as at inference time, we leverage the learned reward model without modeling noise in the reward predictions.

Whereas this large dataset of pairwise preferences comes from untrusted sources, we assume access to an additional smaller validation dataset that is labeled by experts we trust (see Fig. 1). For example, a company might train a robot to perform challenging household tasks, such as cooking. To cheaply collect data for these tasks, the company might make use of crowd-sourcing. Instead of manually verifying the collected labels, the company could collect a small trusted dataset to facilitate safe training with the crowd-sourced one.

Though we assume annotators to be noisy with respect to a single ground-truth function, in this work, we assume that they are not explicitly adversarial. In practice, annotators could provide labels maliciously [11] and collude [10]. We leave further study of these settings for future work.

IV. Estimating Annotator Expertise for Offline Reward Learning

Following the established approach for RLHF, we model the annotators' choices using the Bradley-Terry model [8]. We learn a reward function $r(s, a)$ in a contextual bandit setting with preferences over pairs of actions a^1 and a^2 given a state s. For example, states can be sensor inputs and proprioceptive state of a robot, and actions control signals to the robot's actuators. The annotator's probability of preferring a^1 over a^2 is modeled as:

$$P[a^1 \succ a^2 \mid s] = \frac{\exp(\beta \cdot r(s, a^1))}{\exp(\beta \cdot r(s, a^1)) + \exp(\beta \cdot r(s, a^2))},$$

where β is a hyperparameter modeling rationality in the human feedback and is often set to $\beta = 1$ for convenience. We parameterize the reward function as a neural network and learn its parameters using a standard maximum-likelihood approach.

Instead of assuming a constant β, Daniels-Koch and Freedman [16] model β as a function over arbitrary preference queries, $\beta_i(s, a^1, a^2)$. In this way, the inverse temperature parameter β can now be interpreted as an annotator-specific expertise. Unlike Daniels-Koch and Freedman, we hypothesize

that in most cases, rather than varying as an arbitrary function of the preference query, the expertise varies only with respect to the annotator and a relevant *context*. In other words, for a given context, the expertise may differ between each annotator, but does not generally vary for different action choices in the *same* context. We further hypothesize that this "context" is a function of the current state s only. For example, the context could represent different tasks in which annotators might have different expertise, or, in a question-answering setting, the context might be the question topic.

Consequently, we define the expertise as $\beta_i(s)$, where i is the annotator for the current query. This formulation is similar to the state-dependent expertise for imitation learning used in ILEED [6]. The full formulation now becomes,

$$P(a_1 \succ a_2 \mid i, s) = \frac{\exp(\beta_i(s) \cdot r(s, a^1))}{\exp(\beta_i(s) \cdot r(s, a^1)) + \exp(\beta_i(s) \cdot r(s, a^2))}. \quad (1)$$

This formulation generalizes previous approaches [14, 33, 39] that assume a *constant* annotator-dependent β_i.

A. Learning the Annotator Expertise

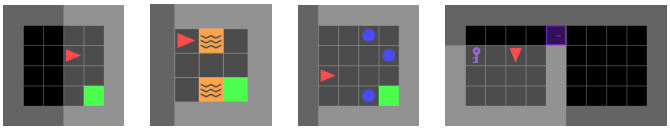
In our formulation, the annotator- and context-dependent expertise is a function of an anonymous annotator ID and the state s. We use two approaches to estimate the expertise values.

First, similar to ILEED [6], we consider a normalized dot product of a learned annotator embedding ω_i and a flattened representation of the state s. We also add a bias b_i to allow the model to learn a constant expertise for all inputs, and normalize the expertise values over the number of annotators, N. The function to determine the expertise is therefore,

$$\beta_i(s) = \frac{1}{N}\left(\langle s, \omega_i \rangle + b_i\right).$$

Secondly, we consider a neural network-based approach to estimate the expertise, where $\beta_i(s) = f_\theta(i, s)$ is a network parameterized by θ. Since we assume that no annotator gives adversarial feedback, it makes sense to enforce non-negative learned expertise values, $\beta_i(s) \geq 0$. We compare the application of various activation functions to the output of $\beta_i(s)$, as a way to ensure this. To learn either the linear or neural network-based expertise models, we incorporate their learnable parameters into the maximum likelihood formulation of standard reward learning.

Since it can be challenging to disentangle the varying annotator noise and query difficulty, we found in preliminary experiments that access to a trusted validation set is useful (see Fig. 1). This smaller dataset is labeled by noiseless annotators, i.e., experts whose labels we trust. We use this validation dataset to select the best checkpoint for inference for both our method and the baseline. Additional experiments ablating whether the validation set contains trusted labels can be found in Table VII of Appendix C.



(a) `Empty` (b) `Lava` (c) `Obstacles` (d) `Unlock`

Fig. 2: For our experiments, we use the MiniGrid [13] environments `Empty`, `Lava`, `Obstacles`, and `Unlock`. All environments have partially observable states, and the goal for the agent is to reach the green square. In `Lava`, if the agent touches a lava cell, it receives a large negative reward and the episode ends. In `Obstacles`, if the agent touches one of several blue obstacles that randomly move around the grid, it receives a large penalty and the episode ends. In `Unlock`, the agent must collect a key to open a door.

V. Experiments

We design experiments to answer the following questions:

(Q1) Can we learn the annotator expertise from preferences in an ***unsupervised*** manner?

(Q2) Does jointly learning the annotator expertise and reward function help us to learn ***better reward functions***?

(Q3) What ***architectural decisions*** affect the expertise model?

A. Data Collection

Our proposed training method requires preferences over binary queries in addition to annotator identifiers. Because existing preference datasets do not contain annotator identifiers, we simulate noisy annotators on newly-collected MiniGrid [13] environments. While existing preference datasets do not contain annotator identifiers, we emphasize that it would be easy to collect them for new preference datasets.

1) Environments: Mirroring the setup of Beliaev et al. [6], we perform experiments in four MiniGrid [13] environments illustrated in Figure 2. In all environments, states are only partially observable, and initial states are randomized. Reward functions are learned from partial observations in place of states, which is common in RLHF.

2) Annotator Labels: We mirror the data collection setup of Beliaev et al. [6]. First, we train an expert "oracle" policy using RL on the MiniGrid environment rewards. Details of the RL training are given in Appendix A. For each environment, we use the trained RL policy as an oracle.

We simulate annotators of varying expertise levels who provide trajectory demonstrations and subsequently provide pairwise preferences in which they prefer their own demonstrated actions over any other actions. We assign a probability p_i to each annotator, representing that annotator's probability of taking the oracle's action as opposed to a randomly sampled action. Then, we collect preferences using a bandit-like approach in which at every step, we compare a randomly sampled action with the oracle's action. The ground-truth preference corresponds to the oracle's action; the *annotator's preference* is whichever action the annotator took at that state. Each preference query for training the reward model consists

of a state, two alternative actions, the annotator's preference, and an integer ID identifying the annotator.

Depending on whether we generate data from an annotator with constant or context-dependent expertise, we either model a constant probability p_i of that annotator taking the correct action or have multiple p_i that vary depending on the context.

We note that, despite labels being binary, there are more than two possible actions in any given state. Thus, a very noisy annotator, such as one with $p = 0.01$, will have less than 50% chance of preferring the "oracle" action. This means that the reward function's ranking accuracy is likely to be less than 50% if it does not account for annotator expertise or learns an incorrect expertise level.

B. Experiment Setup

1) Algorithmic Comparisons: To study the effect of learning annotator expertise, we compare several algorithms across two sets of experiments. First, in Section V-C, we study a simulated MiniGrid setting in which each annotator has a predefined constant expertise level. Second, in Section V-D we study preference learning with context-dependent annotator expertise in MiniGrid, such that the annotators' expertise values vary depending on the context.

In Section V-C, with constant annotator expertise, we evaluate the following algorithms:

- **Baseline**: The baseline performs preference learning without learning expertise, rather assuming a constant $\beta = 1$.
- **Ours**: We consider four different conditions which vary the final activation function on the user reliability model, specifically no activation (No Ac.), ReLU, sigmoid (Sigm.), and exponential (Exp).

In Section V-D, all comparisons use a ReLU activation. We consider the same baseline, and evaluate three variations of learning the expertise (described in Section IV-A): We compare estimating the expertise model using a dot product (**LearnDot**) and a neural network (**LearnNN**), as well as a simple model that assumes the expertise is constant across all contexts and only learns a single β parameter per annotator (**LearnConst**). Finally, we compare to a model that receives privileged information (**OracleContext**) in the form of a context vector that correctly identifies the context as used to generate the data.

2) Evaluation Metric: To evaluate the performance of our method, we compare the labeling accuracy on unseen ground-truth labels, which are obtained by querying a noiseless annotator ($\beta \to \infty$). The learned models' labels are chosen by selecting the action with the highest reward assigned by the learned reward model, i.e., generating preference labels without noise.

C. Learning Constant Annotator Expertise

Our first set of experiments evaluates our method's performance in settings where the algorithm learns to perform a single task and where different annotators can have varying expertise levels, although each annotator's expertise is constant across all states. For example, if humans give feedback on shooting a soccer ball, then different annotators might have varying qualifications to assess the robot's shooting technique.

Similar to Beliaev et al. [6], we simulate 10 annotators for three sets of annotator expertise levels, P-1, P-5, and P-unif:

- P-1 $= \{p_1 = 0.99, p_{2:10} = 0.01\}$,
- P-5 $= \{p_{1:5} = 0.99, p_{6:10} = 0.01\}$,
- P-unif $= \{p_i = 0.05 + 0.1(i-1)\}$,

where p_i is the ith annotator's probability of taking the oracle's action instead of a uniformly random one at each time step. Note that we omit settings where most annotators are close to noiseless, since we expect modeling annotator expertise to be most useful when annotators exhibit varying noise.

We evaluate the effect of learning annotator expertise over varying expertise combinations; e.g., P-1 has a relatively low number of accurate annotators. We train separate reward models for each of the four environments described in Figure 2. Because the expertise is constant across all states in these experiments, we learn a constant β_i parameter for each annotator. These experiments serve to evaluate the approach of learning expertise models via simple maximum likelihood optimization in a simplified setting.

We collect 10,000 preference comparisons, equally distributed across the ten annotators for every setting, and calculate average ranking accuracy across 20 seeds. Additional details regarding our training setup can be found in Appendix B.

Method	P-1 (%)	P-5 (%)	P-unif (%)
Baseline	58.0 ± 11.0	53.8 ± 2.6	46.4 ± 5.7
Ours (ReLU)	$\mathbf{72.3 \pm 8.8}$	54.4 ± 10.7	51.2 ± 5.7
Ours (No Ac.)	69.9 ± 12.9	52.7 ± 2.1	$\mathbf{52.1 \pm 2.0}$
Ours (Sigm.)	63.1 ± 15.4	$\mathbf{56.3 \pm 7.7}$	49.0 ± 6.9
Ours (Exp)	59.7 ± 12.4	51.0 ± 7.3	50.8 ± 6.9

TABLE I: **Preference prediction accuracy in `Obstacles`.** Mean and standard deviation for different noise settings, calculated over 20 runs. We find that learning the expertise benefits the accuracy on unseen data.

1) Results: Table I reports the results of these experiments. Due to the large number of noisy annotators, the reward model predictions have a high variance, both with and without estimating the expertise. While we find that, on average, the models that estimate expertise outperform the baseline that does not account for varying annotator expertise, for most methods, the effect is fairly small. We observe the largest average improvement with the noisiest annotators (P-1). In this setting, despite many labels being chosen uniformly at random, our reward model manages to achieve a preference prediction accuracy of over 70% on held-out preference pairs.

Since we assume that annotators are not actively adversarial (see Section III), it is safe to assume that β is non-negative. Thus, in the same table, we ablate the choice of activation function applied to the learned β values, and also consider the identity activation function (No Ac.), which could potentially learn negative β values. Among the activation functions evaluated, there is not a clear preferred choice; however, in the

highest-noise setting (P-1), where learning the expertise yields the greatest benefit, we find that ReLU activations perform best.

We repeat these experiments for the noisy set of annotators P-1 with various activations on the three other environments: `Empty`, `Lava`, and `Unlock`. The results are reported in Table II. Again, variances are high. We observe the clearest benefit from learning the expertise in the `Unlock` environment. Similar to Table I, we find that the ReLU activation function is beneficial in both `Unlock` and `Empty`. In future work, we plan to extend these results to the more typical setting in which preferences are collected over trajectories, which may ameliorate this problem. Overall, we note that there is some positive signal for estimating the expertise and for ensuring that β is positive by applying ReLU activations.

Environment	`Empty` (%)	`Lava` (%)	`Unlock` (%)
Baseline	39.8 ± 32.1	28.1 ± 16.6	28.8 ± 3.1
Ours (ReLU)	$\mathbf{60.6 \pm 15.4}$	57.8 ± 11.5	$\mathbf{62.7 \pm 14.1}$
Ours (No Ac.)	54.0 ± 13.0	$\mathbf{58.2 \pm 15.9}$	52.7 ± 7.9
Ours (Sigm.)	27.3 ± 20.5	25.0 ± 6.8	32.3 ± 10.9
Ours (Exp)	38.6 ± 29.2	27.6 ± 13.4	29.9 ± 5.8

TABLE II: **Preference prediction accuracy in three additional environments.** Mean and standard deviation, calculated over 20 runs, with the P-1 noise setting. We use the ReLU activation function, based on the results in Table I.

D. Learning Context-Dependent Annotator Expertise

In our second set of experiments, we study whether our method can learn from annotators whose expertise varies depending on the context. The different minigrid environments in these experiments simulate "sub-tasks," or "skills." For example, a soccer robot might be trained to play in all positions, but the humans providing feedback could be specialized.

As before, we follow an experimental procedure similar to ILEED [6], but transferred to the setting of "learning from binary preferences". Expertise varies across different *contexts* by combining the three MiniGrid environments `Empty`, `Lava`, and `Unlock`. In the resulting merged environment, state-action pairs can come from any of the three environments with equal probability.

Each of the three annotators is an expert in a different "sub-environment," performing the oracle's action with probability 1 in that environment, while in the other two environments, the annotator has probability p_i of selecting the oracle action and probability $1 - p_i$ of choosing an action at random. We evaluate the three noise sets P-0.01, P-0.10, and P-0.50, in which $p_i = 0.01$, $p_i = 0.10$, $p_i = 0.50$, respectively. We report implementation details in Appendix B.

We learn reward models with context-dependent annotator expertise to estimate the reward for state-action pairs from all three environments. Our expertise models utilize the ReLU activation function due to its strong performance in Section V-C1.

Algorithm comparisons are as outlined in Section V-B1, where the **OracleContext** comparison's expertise model receives privileged information consisting of a one-hot encoding representing the environment to which a state-action pair belongs, instead of the state. In this oracle comparison case, the algorithm still needs to learn a reward function and the annotator's expertise, but its expertise model does not need to infer context information from the state. This method lets us study whether learning the context is a challenge in our setting.

Expertise Set	**P-0.01 (%)**	**P-0.10 (%)**	**P-0.50 (%)**
Baseline	66.9 ± 2.4	63.9 ± 4.6	$\mathbf{73.1 \pm 5.3}$
LearnConst	67.4 ± 2.7	61.3 ± 3.0	51.3 ± 11.6
LearnDot	$\mathbf{69.4 \pm 3.3}$	$\mathbf{65.2 \pm 3.7}$	47.2 ± 9.8
LearnNN	63.4 ± 7.3	63.1 ± 5.1	51.0 ± 15.8
OracleContext	68.2 ± 3.8	64.6 ± 5.7	49.7 ± 7.5

TABLE III: **Context Dependent Experiments.** Results for three sets of noisy annotators when annotator expertise depends on context (mean and standard deviation over 20 runs). Modeling annotator expertise yields a slight improvement when learning the expertise from relatively-noisy users; however there is a high variance in the results. Results indicate that the dot-product approach may be the best approach to learn context-dependent expertise.

1) Results: Table III shows the results of these experiments. Again, the resulting accuracies have relatively high variance compared to the observed average improvements. As in the constant expertise case, we find that, on average, the benefit of learning the annotator expertise model decreases as the annotators become less noisy. In fact, when half of the annotators give correct responses with a probability of 0.5 in the states in which they are not skilled (P-0.50), learning an expertise estimate with our method seems to *decrease* accuracy compared to the baseline. We hypothesize that accurately learning the additional expertise parameter is difficult, given the low amount of signal about an annotator's expertise in the data. When all annotators' expertise is constant, every data sample labeled by an annotator provides information about that annotator's expertise. In the context-dependent expertise case, a datapoint only provides useful information about other datapoints of the same context. This is further reflected by the fact that the neural network-based method **LearnNN** performs worse than the less parameter-intensive **LearnDot**.

Surprisingly, when we provide "oracle contexts" to the expertise method, we do not observe an improvement in the accuracy. This suggests that identifying the context is not a significant impediment to our expertise estimation method. Likely, even when the correct context is known, it is challenging to determine the correct context-dependent expertise from only preference labels.

To address the limited but positive success of learning from annotators with context-dependent expertise, we plan to evaluate the data distribution and modeling assumptions in future work. Potentially, other architectures for the expertise

model could alleviate current problems, or the learned expertise values could be grounded with additional external data.

VI. Conclusion

We find that modeling annotator expertise can benefit reward learning in RLHF when annotators exhibit a range of degrees of noise in their feedback. Comparing against a baseline that does not model differences between annotators, we find that the benefit of our method becomes more pronounced as the noisiness of the annotators increases. We also find that in our setting, reward model accuracy is improved by ensuring non-negativity of the learned expertise via an activation function. For learning context-dependent expertise, the results are mixed and require further research.

VII. Limitations and Future Work

In this work, we collect preference feedback in a contextual bandit setting, in which annotators give pairwise preferences over actions given a state. In practice, with human annotators, it is usually more feasible to collect preference feedback over full trajectories. An interesting research question is whether expertise should be modeled as constant across the trajectory or whether it should be allowed to vary within a trajectory. Additionally, we plan to extend the current experiments to more complicated control settings such as MetaWorld [40] and CALVIN [26], as well as real-world robotics settings.

In this work, we evaluate the performance of reward models by their ranking accuracy with respect to the ground truth reward. A common approach in preference learning for control is to instead use the learned reward function to generate reward values to train either offline or online RL algorithms [24, 19, 20, 22]. An evaluation based on policies instead of reward function accuracies gives a more accurate picture of the *downstream* performance of the reward learning method.

Because we model the annotator's expertise as a temperature parameter in the Bradley-Terry model, our method is compatible with any Bradley-Terry-based method that learns from binary preferences. In future work, we aim to evaluate our method with *interactive* RLHF approaches, where the reward model is used to iteratively improve the policy and humans are actively queried [15, 32, 7], in addition to methods that *directly* learn policies from preference data, such as DPO [31].

Currently, we explicitly assume that for every task, there is a single ground-truth reward function and that annotators give noisy preference feedback with respect to this reward function. Other works have considered cases where human feedback depends on multiple reward functions and where rankings can *subjectively* vary across humans. Considering the pluralistic nature of rankings might help to align performance with different underlying human values [43, 3, 34] or to achieve personalization [29].

Additionally, when learning expertise in an unsupervised manner, we assume that the majority opinion is correct. In reality, this might not be true, for example, if human preferences are swayed by common misconceptions. It is conceivable that combining a large set of untrusted data with a smaller set of *trusted* evaluators could be useful. Potentially, unsupervised clustering of trusted evaluators with unknown evaluators could help to ensure the correctness of the learned expertise and ameliorate this problem. Additional avenues to consider include further integration with crowd-sourcing techniques to infer reliable evaluators in unsupervised settings.

Lastly, while we perform offline estimation of annotator expertise, prior work has shown that when annotator expertise is known, it is possible to improve the efficiency of the learning process by actively selecting annotators to query [16, 4, 17]. As our method learns annotator expertise jointly with the reward, it remains an open question how to best navigate the tradeoff between querying annotators to learn better expertise estimates and exploiting current expertise estimates.

Acknowledgments

This work was sponsored in part by DEVCOM Army Research Laboratory under Cooperative Agreement Number W911NF-19-S-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Government. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

[1] P. Abbeel, A. Coates, and A. Y. Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13): 1608–1639, 2010. doi: 10.1177/0278364910371999. URL https://doi.org/10.1177/0278364910371999.

[2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009. ISSN 0921-8890. doi: 10.1016/j.robot.2008.10.024. URL https://doi.org/10.1016/j.robot.2008.10.024.

[3] P. Awasthi, A. Blum, O. Sheffet, and A. Vijayaraghavan. Learning mixtures of ranking models. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/e40e5b7841b81e794e14fd9be7307c9e-Paper.pdf.

[4] P. Barnett, R. Freedman, J. Svegliato, and S. Russell. Active reward learning from multiple teachers. In *SafeAI Workshop at AAAI 2023*, 2023. doi: 10.48550/arXiv.2303.00894.

[5] M. Beliaev and R. Pedarsani. Inverse reinforcement learning by estimating expertise of demonstrators. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(15):15532–15540, 2025. doi: 10.1609/aaai.v39i15.33705. URL https://ojs.aaai.org/index.php/AAAI/article/view/33705.

[6] M. Beliaev, A. Shih, S. Ermon, D. Sadigh, and R. Pedarsani. Imitation learning by estimating expertise of demonstrators. In K. Chaudhuri, S. Jegelka, L. Song,

C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 1732–1748. PMLR, 2022. URL https://proceedings.mlr.press/v162/beliaev22a.html.

[7] E. Biyik and D. Sadigh. Batch active preference-based learning of reward functions. In A. Billard, A. Dragan, J. Peters, and J. Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 519–528. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/biyik18a.html.

[8] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 00063444. URL http://www.jstor.org/stable/2334029.

[9] D. Brown, W. Goo, P. Nagarajan, and S. Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 783–792. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/brown19a.html.

[10] T. K. Buening, J. Gan, D. Mandal, and M. Kwiatkowska. Strategyproof reinforcement learning from human feedback, 2025. URL https://arxiv.org/abs/2503.09561.

[11] X. Chen, P. Bennett, K. Collins-Thompson, and E. Horvitz. Pairwise ranking aggregation in a crowd-sourced setting. In *WSDM 2013 - Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, pages 193–202, 02 2013. doi: 10.1145/2433396.2433420.

[12] C.-A. Cheng, A. Kolobov, and A. Agarwal. Policy improvement via imitation of multiple oracles. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5587–5598. Curran Associates, Inc., 2020.

[13] M. Chevalier-Boisvert et al. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, abs/2306.13831, 2023.

[14] D. Chhan, E. Novoseller, and V. J. Lawhern. Crowd-prefrl: Preference-based reward learning from crowds. *CoRR*, abs/2401.10941, 2024.

[15] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep Reinforcement Learning from Human Preferences. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.

[16] O. Daniels-Koch and R. Freedman. The Expertise Problem: Learning from Specialized Feedback. In *SafeAI Workshop at AAAI 2022*, 2022.

[17] R. Freedman, J. Svegliato, K. Wray, and S. Russell. Active teacher selection for reinforcement learning from human feedback, 2023. URL http://arxiv.org/abs/2310.15288.

[18] A. Gleave et al. imitation: Clean imitation learning implementations. arXiv:2211.11972v1 [cs.LG], 2022. URL https://arxiv.org/abs/2211.11972.

[19] J. Hejna and D. Sadigh. Inverse Preference Learning: Preference-based RL without a Reward Function. In *Thirty-Seventh Conference on Neural Information Processing Systems*, November 2023.

[20] J. Hejna, R. Rafailov, H. Sikchi, C. Finn, S. Niekum, W. B. Knox, and D. Sadigh. Contrastive Preference Learning: Learning from Human Feedback without Reinforcement Learning. In *ICLR*, January 2024.

[21] J. Ho and S. Ermon. Generative adversarial imitation learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/cc7e2b878868cbae992d1fb743995d8f-Paper.pdf.

[22] C. Kim, J. Park, J. Shin, H. Lee, P. Abbeel, and K. Lee. Preference transformer: Modeling human preferences using transformers for RL. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Peot1SFDX0.

[23] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg. Dart: Noise injection for robust imitation learning. In *Conference on robot learning*, pages 143–156. PMLR, 2017.

[24] K. Lee, L. Smith, and P. Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In *International Conference on Machine Learning*, 2021.

[25] X. Liu, T. Yoneda, C. Wang, M. Walter, and Y. Chen. Active policy improvement from multiple black-box oracles. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 22320–22337. PMLR, 2023. URL https://proceedings.mlr.press/v202/liu23av.html.

[26] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. In *IEEE Robotics and Automation Letters*, 2022.

[27] A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, page 663–670, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1558607072.

[28] OpenAI. GPT-4 Technical Report, 2023. URL http://arxiv.org/abs/2303.08774.

[29] S. Poddar, Y. Wan, H. Ivison, A. Gupta, and N. Jaques. Personalizing Reinforcement Learning from Human Feedback with Variational Preference Learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, November 2024.

[30] D. A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991. doi: 10.1162/neco.1991.3.1.88.

[31] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Thirty-Seventh Conference on Neural Information Processing Systems*, November 2023.

[32] D. Sadigh, A. Dragan, S. Sastry, and S. Seshia. Active Preference-Based Learning of Reward Functions. In *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation, 2017. ISBN 978-0-9923747-3-0. doi: 10.15607/RSS.2017.XIII.053. URL http://www.roboticsproceedings.org/rss13/p53.pdf.

[33] S. Singhal, C. Laidlaw, and A. Dragan. Scalable oversight by accounting for unreliable feedback. In *ICML 2024 Workshop on Models of Human Feedback for AI Alignment*, 2024. URL https://openreview.net/forum?id=Noy5wbyiCS.

[34] T. Sorensen et al. Value Profiles for Encoding Human Variation, 2025. URL http://arxiv.org/abs/2503.15484.

[35] N. Stiennon et al. Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf.

[36] V. Tangkaratt, B. Han, M. E. Khan, and M. Sugiyama. Variational imitation learning with diverse-quality demonstrations. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9407–9417. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/tangkaratt20a.html.

[37] H. Touvron et al. Llama 2: Open Foundation and Fine-Tuned Chat Models, 2023. URL http://arxiv.org/abs/2307.09288.

[38] Y.-H. Wu, N. Charoenphakdee, H. Bao, V. Tangkaratt, and M. Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pages 6818–6827. PMLR, 2019.

[39] T. Yamagata, T. Oberkofler, T. Kaufmann, V. Bengs, E. Hüllermeier, and R. Santos-Rodriguez. Relatively rational: Learning utilities and rationalities jointly from pairwise preferences. In *ICML 2024 Workshop on Models of Human Feedback for AI Alignment*, 2024. URL https://openreview.net/forum?id=N9o8afNUfr.

[40] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, 2020.

[41] G. Zhang and H. Kashima. Batch reinforcement learning from crowds. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 38–51. Springer, 2022.

[42] S. Zhang, Z. Cao, D. Sadigh, and Y. Sui. Confidence-aware imitation learning from demonstrations with varying optimality. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 12340–12350. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/670e8a43b246801ca1eaca97b3e19189-Paper.pdf.

[43] X. Zhang, X. Zhang, P.-L. Loh, and Y. Liang. On the identifiability of mixtures of ranking models, 2022. URL http://arxiv.org/abs/2201.13132.

[44] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-Tuning Language Models from Human Preferences, 2019. URL http://arxiv.org/abs/1909.08593.

APPENDIX

Parameter	Value
Policy Class	MlpPolicy
Update Steps	128
Num. of Environments	8
Batch Size	4
Learning Rate	0.00025
Timesteps	200,000
Timesteps for `Unlock`	400,000

TABLE IV: Hyperparameters for expert policy training.

Environment	Expert Return
`Empty`	0.967
`Lava`	0.935
`Obstacles`	0.933
`Unlock`	0.821

TABLE V: Returns achieved by the expert policies in the MiniGrid environments, according to the true reward.

A. RL Policy Training

Because we study the same environments as Beliaev et al. [6], we mirror their hyperparameters and setup. Details of the hyperparameters are given in Table IV. Note that in this setup, we train multiple policies in parallel for each environment and use the best-performing one as our expert, including old checkpoints, if they perform better. The returns achieved by the chosen experts are reported in Table V. Note that the `Unlock` environment is the only one where the trained RL policies did not achieve satisfactory performance with the provided hyperparameters, which is why we report results for an `Unlock` expert that trained for twice the number of timesteps (400K).

B. Reward Model Training Details

Parameter	Value
Epochs	500
Batch Size	1024
Learning Rate	0.01
Num. of Preference Comparisons	10000
Validation Dataset	10%
Test Dataset	10%
Num. of Seeds	20
NN-RM Architecture	2-Layer Feed Forward
NN-RM Hidden Sizes	4

TABLE VI: Hyperparameters for reward model training.

Table VI shows various hyperparameter choices for reward model training. Multiple hyperparameters were adapted from Beliaev et al. [6], as they study a similar setting. We make use of various auxiliary methods for learning from pairwise preference comparisons from the library `imitation` [18].

Method	Trusted Validation	Untrusted Validation
Baseline	$58.0\% \pm 0.110$	$59.1\% \pm 0.146$
Ours (ReLU)	$\mathbf{72.3\% \pm 0.088}$	$\mathbf{69.8\% \pm 0.074}$
Ours (No Ac.)	$69.9\% \pm 0.129$	$54.6\% \pm 0.025$
Ours (Sigm.)	$63.1\% \pm 0.154$	$56.5\% \pm 0.114$
Ours (Exp)	$59.7\% \pm 0.124$	$55.1\% \pm 0.130$

TABLE VII: **Mean accuracies in `Obstacles`.** With the P-1 noisy annotator setting, we compare selecting checkpoints based on a trusted or untrusted validation dataset. Learning the expertise with a ReLU activation is beneficial in both settings.

C. Using a Trusted Validation Set

We ablate on the choice of what kind of validation set we use to select the final checkpoint to evaluate. These experiments are for a constant-expertise setting with significantly noisy annotators (P-1) in the `Obstacles` environment. Other experiments in this paper use a small validation set with trusted labels, and select the checkpoint that maximizes the accuracy on this validation set. It is reasonable to assume that such a dataset could be generated for many tasks, while keeping the cost low, due to the small size of the validation set ($< 10\%$ of the training dataset size).

We compare this to using an "untrusted validation set." This validation set is labeled by the same annotators as the training set and thus contains noisy labels. In this setting, we choose checkpoints based on the lowest loss on the annotator-provided labels. Intuitively, using a trusted validation set selects for checkpoints that are better at predicting the rewards, whereas using an untrusted validation set selects for checkpoints that more accurately predict the annotator's choices.

Table VII compares the performance when using the two approaches of selecting annotators. We find that for the baseline, there is little difference in what type of validation dataset we use. This is likely because the baseline does not distinguish between the label an annotator would give and the label from the current estimate of the true reward function. However, when learning the expertise while using a ReLU activation function, we observe a slight decrease in performance with the untrusted validation set. Importantly, however, this method outperforms the baseline and most other methods, *no matter which type of validation set we use*. In other words, even when a trusted validation set is not available, in the noisy annotator setting, learning the expertise is beneficial.

Additionally, we find that with other activation functions, the performance with an untrusted validation set decreases below that of the baseline. This suggests that these methods might struggle more with distinguishing annotator expertise and reward labels. The trusted validation set provides an additional source of information that grounds these methods.

We posit that further adding information regarding the annotators' expertise, or integrating known reward information into the training process directly, could serve as a promising direction to improve learning expertise.