Augmenting Active Preference-Based Learning of Reward Functions with Reinforcement Learning

Siddarth Ijju

Abstract

Training complex robots is limited by the usefulness and ease of collection of the underlying data. One significant blocker for this task is the accuracy of the reward function used in reinforcement learning algorithm. The current state-of-the-art relies on human demonstrations to learn the reward functions for a given task. However, human demonstrations can be infeasible to collect and difficult to learn off of. Preference-based algorithms attempt to work around this issue by using a sequence of queries comparing various trajectories to the user in place of actual human demonstrations. We propose a new query generation algorithm that utilizes a combination of reinforcement learning and a greedy optimization objective to improve over previous query generation algorithms. We propose two frameworks for training such a query generation algorithm in a reinforcement learning environment - one based of using the query as an action, and the other based on using the query as part of the state. We perform experiments within a Gridworld environment using four dimensional reward parameters, a query size of 2 queries, and a query sequence length of 20, and design a custom Query environment for each reinforcement learning framework. In addition, we evaluate results on the alignment of the predicted reward parameters with the true reward parameters. Overall, we find that our current algorithm outperforms random query generation but fails to outperform state-of-the-art greedy query generation, leading to two potential conclusions. First, our work could provide empirical evidence for the optimality of state-of-the-art greedy query generation algorithms. Second, our work could show the feasibility of the reinforcement learning approach and open avenues for future research. We would like to perform further experiments around changing the RL objective, modifying the policy training procedure to account for task uncertainty, and generating continous queries instead of optimizing over a discrete set of queries.

1 Introduction

To deploy an advanced robot in the field, it is critical for that robot to understand how the human deployer wants it to behave. Consider a simple task such as preparing a cup of coffee for a user. The robot must consider a variety of factors related to that user - how hot they like their coffee, what method to use to strain the liquid, how to press the beans, etc. This behavior is often highly personal and dependent on the particular user and their reward function. To circumnavigate this obstacle, a robot could ask questions about the desired behavior from the user, similar to how humans often ask clarifying questions when learning a new task.

In the current state-of-the-art, robots pose and solve an optimization problem to determine what questions they should ask [1, 2, 3]. In many of these works, the questions asked are not necessarily easy to answer - for example, they may ask if the human prefers their coffee at 31 degrees or 31.1 degrees. Further work aims to solve this problem by changing the optimization objective to account for the ease of answering the question [4].

However, there is a relatively small corpus of work dedicated to determining the best order in which to ask questions to the user. State of the art algorithms currently use a greedy formulation to ask the best query at each time step according to the optimization objective. However the greedy questioning approach may not be optimal because at each timestep we lose information about what questions we have already asked. In extreme cases, it could be possible for a robot to ask the same question multiple times because that particular question is significantly higher than the others based on the optimization objective.

In this paper, we focus on finding the most optimal sequence of questions while also accounting for the human's ability to answer them and the relative cost of computing this answer. Considering the optimal query sequence can improve the robot's learning by strategically asking relevant questions based off the previously asked questions and leveraging past questions and answers in its selection of the next question. Based on this observation, we develop an approach that actively learns the best sequence of questions to determine the human's reward function.

2 Related Work

2.1 Reward Learning

There is an extensive body of research into learning correct human reward function from human feedback. Inverse reinforcement learning (IRL) recovers the reward from expert demonstrations by humans the show the robot how to perform the task [5, 6, 7, 8, 9]. However, in many cases providing demonstrations is a difficult task, especially when the robot is complicated [10, 11].

Preference-based learning attempts to address this issue by providing a human-friendly alternative: the human is shown a few potential trajectories and then asked to select the best one (or rank each of them) [12, 13, 14]. In this paper, we are interested in leveraging preference-based learning while choosing the queries in an intelligent manner, so that we can improve data efficiency and user experience.

2.2 Active Learning

There is also an extensive body of research into selecting the most informative queries in learning settings so that the robot learns as much as possible from only a few questions [15]. The current state-of-the-art applies active learning to preference-based learning using various optimization objectives to select the queries [1, 13, 16, 17]. One previously popular approach was to maximize the volume removed from the hypothesis space (i.e., volume removal) [1, 3, 14]. There have since been advances made for a new objective that maximizes the information gained from a question[4].

3 Problem Formulation

3.1 Model

In our experiments, we use a deterministic, fully-observable dynamical system, specifically a grid-world. We use $s_t \in S$ to denote the state and $a_t \in A$ to denote the action, for timestep t. A trajectory $\xi \in \Xi$, is a finite sequence of states and actions. i.e., $\xi = ((s_t, a_t))_{t=0}^{T}$ where T is the horizon. We model the human with a reward function $R : \Xi \to R$ that describes how a human wants the actor in the system to behave. For simplicity, we assume R is a linear combination of selected features $\phi : \Xi \to \mathbb{R}^d$. Then, we can write $R(\xi) = \omega^T \phi(\xi)$, and now we only have to learn the d-dimensional vector ω to learn the human's reward function.

3.2 Query

We define a query $Q = \{\xi_1, \xi_2, \ldots, \xi_K\}$ as a set of K trajectories. The human responds to a query by picking what the believe is the best trajectory from this set. In our formulation, we will attempt to learn the human's reward parameter ω by choosing a sequence of queries for the human to respond to. To minimize the number of queries we need to ask, we want to find the sequence of queries that maximizes the information that we learn about the human's reward parameter ω. This problem is in general, quite difficult - there is no closed form solution to determine the best sequence of queries at the beginning. We can however, proceed in a greedy fashion - we initialize a starting distribution over ω, and then alternate between generating a query Q and using the human's response to Q to update the distribution over ω.
First, we can outline the methodology for the second step: updating the belief distribution over ω. Define $\mathbb{P}(q \mid Q, \omega)$ as the probability that the human chooses trajectory q from the query Q assuming a reward parameter of ω. In our experiments, since we do not have access to a live human (and it is inefficient to assume a live human for training), we can use the Boltzmann Rationality model, which has been used extensively in literature [18, 19, 20]. We have

$$\mathbb{P}(q = \xi_i \mid Q, \omega) = \frac{\exp \tau R(\xi_i)}{\sum_{\xi \in Q} \tau R(\xi)}$$

where τ is a tunable hyperparameter known as the temperature. Assuming a prior $\mathbb{P}(\omega)$ and a response q to a query Q, we can then compute the posterior over ω:

$$\mathbb{P}(\omega \mid Q, q) \propto \mathbb{P}(q = \xi_i \mid Q, \omega)\mathbb{P}(\omega) = \frac{\exp \tau R(\xi_i)}{\sum_{\xi \in Q} \tau R(\xi)}\mathbb{P}(\omega)$$

In general this is intractable to compute, since the distribution $\mathbb{P}(\omega)$ can get increasingly complex. We use Monte Carlo sampling to approximate $\mathbb{P}(\omega)$.

3.3 Query Generation

Now, we can outline the methodology for the first step from the previous section - generating a query Q given a belief distribution $\mathbb{P}(\omega)$. For our experiments, we optimize over a large discrete set of queries π, such that we select potential queries $Q \in \pi$. In general, the continuous case for query generation [1] is possible, but significantly more difficult - we experiment over the discrete

case first to determine feasibility. We define a new environment model where we select $s \in \Omega$, the set of possible beliefs ω, $a \in \pi$. We define s' for a state action pair (s, a) as the mean of the updated belief distribution $\mathbb{P}(s' \mid Q, q)$ where q is simulated via the Boltzmann Rational model from the previous section with Q and s. We then utilize the information gain formulation used in literature [4] as the reward r for the pair (s, a) - we have

$$r = H(\omega \mid Q) - \mathbb{E}_{q_i} H(\omega \mid q_i, Q)$$

Again, we use sampling [4] over M samples of $\omega \in \Omega$ to estimate the reward as

$$r = \frac{1}{M} \sum_{q_i \in Q} \sum_{\omega \in \Omega} \mathbb{P}(q_i \mid Q, \omega) \log_2 \left(\frac{M \cdot \mathbb{P}(q_i \mid Q, \omega)}{\sum_{\omega' \in \Omega} \mathbb{P}(q_i \mid Q, \omega')} \right)$$

where the query Q is the action a, the belief ω is the state s. This expression is convenient because it only relies on s and a to compute r, while simultaneously factoring in s' because it is directly derived from $H(\omega \mid Q) = H(s' \mid a)$. This formulation has an additional experimental benefit - we can utilize the samples from approximating the distribution $\mathbb{P}(\omega \mid q, Q)$ within the calculation of the reward for a particular pair (s, a).

3.4 RL World Setup

With the previous setup for query generation in the RL case, we outline two different methods for conducting RL experiments. First, we proceed as outlined before, using queries as actions a and beliefs as states s. However, this method suffers from a linear action space relative to the size of the dataset, which can make learning over this space difficult.

In our experiment, we propose another setup where we define the state $s = (\omega, Q) \in (\Omega, \pi)$ and the action $a \in (0, 1)$. In this setup, the action represents whether to ask the query Q in the state s or not. If we choose $a = 0$, we define $s' = s$ and the reward $r = -H(\omega \mid Q) + \mathbb{E}_{q_i} H(\omega \mid q_i, Q)$ (i.e. the regret from not asking the query Q). If we choose $a = 1$, we sample a new belief ω from $\mathbb{P}(\omega \mid Q, q)$ and define $s' = (\omega, Q')$ where Q' is randomly selected from π.

This formulation has a significantly smaller action space at the cost of a larger state space. Additionally, assuming a finite horizon H for the first RL environment described above (i.e. a limit to the number of queries we can ask) and that the actions are equally distributed between 0 and 1, the second formulation will take on average $2H$ actions to complete the same number of queries. We conduct experiments over both RL setups below, labeling experiments done under the second setup as feed experiments (we are feeding a stream of random queries to the environment and asking whether or not we should ask that query given a particular belief).

4 Experiments

4.1 Implementation

We perform experiments in a Gridworld setting with two dimensional states and one dimensional actions. We featurized the trajectories using the average distance to each corner of the Gridworld

across the entire trajectory, meaning that our ω is four dimensional. We normalize trajectories to zero mean and unit variance across the training dataset, and use the mean and variance statistics pre-normalization to normalize the test dataset of trajectories. We set $K = 2$ (i.e. two trajectories per query Q), $M = 100$, and $H = 20$. As mentioned previously, we generate datasets of queries beforehand of size 1000, generating separate datasets for training and testing. To judge convergence of the predicted reward parameters with the true reward parameters, we use the alignment metric $\frac{1}{M}\sum_{\hat{\omega}\in\Omega}\frac{\omega^T\hat{\omega}}{||\omega||_2||\hat{\omega}||_2}$, where we desire a score of 1 (perfect alignment). We average results over a batch of 100 true reward parameters ω. We also experiment with the noisiness of the human response model $\mathbb{P}(q \mid Q, \omega)$ by changing the Boltzmann temperature τ. For both setups we train an advantage actor-critic model (A2C) with learning rate .0007.

4.2 Results

As seen in Figure 1, the reinforcement learning approaches fail to outperform the greedy approaches in both cases. When the user is noisy, the reinforcement learning approach using the query as an action (RL) outperforms random while the reinforcement learning approach using the query as part of the state (RL-feed) performs roughly equally to random query generation. However, when the user is optimal, RL-feed outperforms random and the RL approach performs roughly equally to random. We have some theories as to why the RL approaches are unable to outperform the greedy baseline, which we will discuss in the next section.

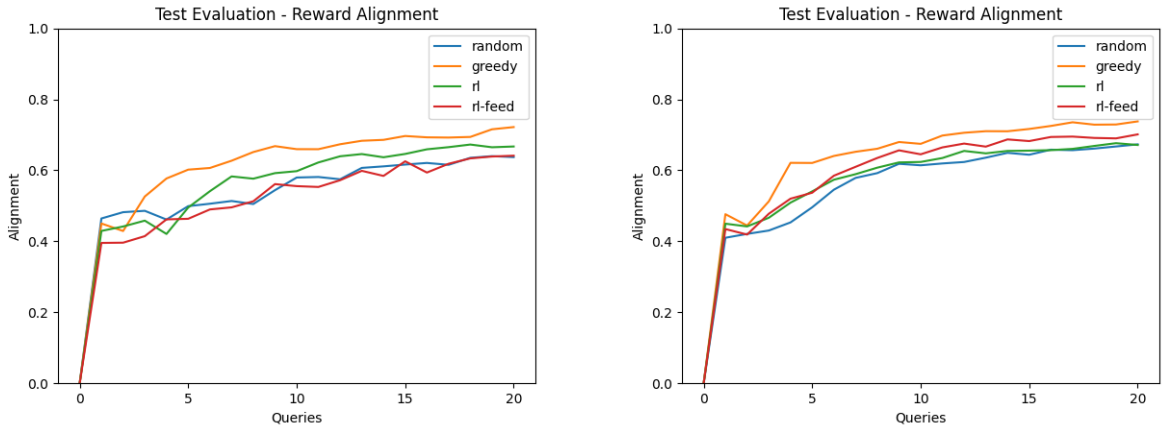

Figure 1: 10000 iterations. Left: Noisy user response (90%). Right: Optimal user response.

Specifically looking at the results for the noisy user run of RL-feed (Figure 2), we notice that as the model improves, the average number of queries tends to decrease. This is unexpected - one would assume that the model would become more selective with the queries the model selects over time, resulting in longer sequences. One possible explanation for this is that the model first learns to approximate the random query sequence performance and then will approach the performance of the greedy query sequence with enough iterations.

One potential reason behind why the reinforcement learning approach failed to outperform the

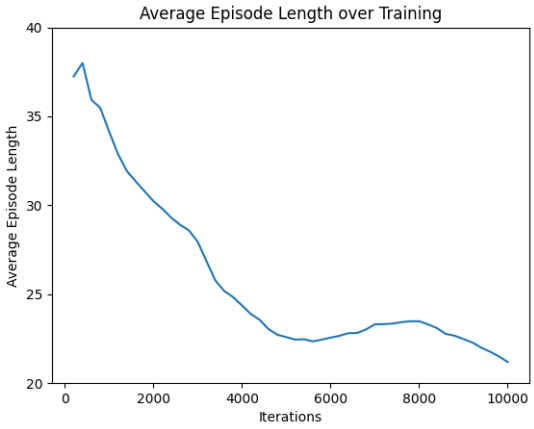

Figure 2: Average training episode length for RL-feed with default parameters

greedy approach is that the environments for training the reinforcement learning approach randomized the true human at every reset. Effectively, this ensured that the reinforcement learning algorithm only had one rollout (only up to H transitions for RL and up to $2H$ transitions for RL-feed) to attempt to learn a proper response for each environment, which is difficult.

5 Discussion

5.1 Conclusions

Although both tested models outperformed random query generation in some scenarios, neither were able to outperform greedy query generation in general. There are several possible reasons for this, which we will discuss below in relation to potential future work, but the fact that the RL algorithms were able to outperform random query generation is a promising result. With hyperparameter tuning and a modification in model architecture and setup, it may be possible for RL algorithms to beat greedy query generation. At worst, this work could provide empirical evidence that greedy query generation is the most optimal and feasible algorithm for this particular task.

5.2 Future Work

One potential problem with the current setup is the task uncertainty the model faces during training. Currently, the training procedure assumes relative alignment between the belief used as state and the true reward, but this may not always be the case. Future work may involve incorporating ideas from [21] to account for task uncertainty (i.e. the true human reward parameter) while training a policy to maximize reward in a single rollout.

Another potential problem with the current setup is the reward objective used. Currently, the reward objective used is the same as the optimization objective used in the state of the art greedy

query generation schemes. It could be possible that this objective is not well suited for reinforcement learning - further work could explore other objectives and their relative performance.

Another obvious problem with the current setup is the use of a limited discrete set of queries. Future work may improve off of work such as [1] which can generate continuous queries - this could also vastly speed up the training process and allow for better convergence.

References

[1] D. Sadigh, A. D. Dragan, S. S. Sastry, and S. A. Seshia. Active preference-based learning of reward functions. *In Proceedings of Robotics: Science and Systems (RSS)*, July 2017.

[2] Y. Cui and S. Niekum. Active reward learning from critiques. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6907–6914. IEEE, 2018.

[3] M. Palan, G. Shevchuk, N. C. Landolfi, and D. Sadigh. Learning reward functions by integrating human demonstrations and preferences. In *Proceedings of Robotics: Science and Systems (RSS)*, June 2019.

[4] Bıyık, E., Palan, M., Landolfi, N. C., Losey, D. P., and Sadigh, D. Asking easy questions: A user-friendly approach to active reward learning. In *Conference on Robot Learning (CoRL)*, 2019.

[5] 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

[6] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine Learning*, page 1. ACM, 2004.

[7] A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000

[8] P. Abbeel and A. Y. Ng. Exploration and apprenticeship learning in reinforcement learning. In *Proceedings of the 22nd International Conference on Machine learning*, pages 1–8. ACM, 2005.

[9] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[10] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[11] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan. Learning from physical human corrections, one feature at a time. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 141–149. ACM, 2018.

[12] R. Holladay, S. Javdani, A. Dragan, and S. Srinivasa. Active comparison based learning incorporating user uncertainty and noise. In *RSS Workshop on Model Learning for Human-Robot Communication*, 2016.

[13] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, 2017.

[14] E. Biyik and D. Sadigh. Batch active preference-based learning of reward functions. In *Conference on Robot Learning (CoRL)*, October 2018.

[15] B. Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

[16] C. Daniel, M. Viering, J. Metz, O. Kroemer, and J. Peters. Active reward learning. In *Robotics: Science and systems*, 2014.

[17] R. Akrour, M. Schoenauer, and M. Sebag. April: Active preference learning-based reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 116–131. Springer, 2012.

[18] E. Biyik, D. A. Lazar, D. Sadigh, and R. Pedarsani. The green choice: Learning and influencing human decisions on shared roads. In *Proceedings of the 58th IEEE Conference on Decision and Control* (CDC), December 2019.

[19] S. Guo and S. Sanner. Real-time multiattribute bayesian preference elicitation with pairwise comparison queries. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 289–296, 2010.

[20] P. Viappiani and C. Boutilier. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Advances in Neural Information Processing Systems*, pages 2352–2360, 2010.

[21] Luisa M. Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep RL via meta-learning. In *8th International Conference on Learning Representations, ICLR 2020*, Addis Ababa, Ethiopia, April 26-30, 2020.