Iterative Cost Learning from Different Types of Human Feedback

Oleg Arenz[1] and Gerhard Neumann[1]

Abstract— **Human-robot collaboration in unstructured environments often involves different types of interactions. These interactions usually occur frequently during normal operation and may provide valuable information about the task to the robot. It is therefore sensible to utilize this data for lifelong robot learning. Learning from human interactions is an active field of research, e.g., Inverse Reinforcement Learning, which aims at learning from demonstrations, or Preference Learning, which aims at learning from human preferences. However, learning from a combination of different types of feedback is still little explored. In this paper, we propose a method for inferring a reward function from a combination of expert demonstrations, pairwise preferences, star ratings as well as oracle-based evaluations of the true reward function. Our method extends Maximum Entropy Inverse Reinforcement Learning in order to account for the additional types of human feedback by framing them as constraints to the original optimization problem. We demonstrate on a gridworld, that the resulting optimization problem can be solved based on the Alternating Direction Method of Multipliers (ADMM), even when confronted with a large amount of training data.**

I. Introduction

Robots that operate in unstructured environments usually have to rely on human assistance, e.g. by providing demonstrations via kinesthetic teaching or tele-operation, or by rating the actions of the robot based on an absolute scale or relative to each other. Such feedback may either be provided intentionally by the human or it may be inferred based on the human-robot interactions. For example, when teleoperating a semi-autonomous robot for sorting nuclear waste, the control signals provided by the operator can be treated as expert demonstrations. Furthermore, the semi-autonomous system might suggest several options to the operator, e.g. different items to be grasped next, and elicit preferences based on the selection of the operator. By using the human feedback for adapting the behavior of the robot, the necessary amount of human assistance can be decreased, reducing the workload on the operator. However, while a number of methods have been developed for learning from a given type of human feedback, it remains unclear how several, different types of human feedback can be combined for learning a consistent representation of the task.

Reward functions are concise task representations that are commonly applied in robotics for specifying optimal behavior. However, manually specifying a reward function for a given task is cumbersome since the induced behavior is often hard to predict, especially when the robot operates in an unstructured environment. The difficulty of specifying reward functions directly, led to the development of methods for inferring reward functions from more intuitive–but less concise–ways of teaching a task to robot. Inverse reinforcement learning (IRL) [1] infers a reward function from expert demonstrations, whereas preference learning (PL) [2] is based on human preferences.

In this paper we propose a method for learning a reward function that is consistent with several, different types of human feedback, namely expert demonstrations, pairwise preferences, star ratings and oracle-based evaluations. Our method is based on Maximum Entropy Inverse Reinforcement Learning (MaxEnt-IRL) [3], extending it to enforce consistency with the other types of human feedback.

A. Related Work

Inverse Reinforcement Learning [1] aims at finding a reward function for a Markov Decision Problem (MDP) that is consistent with observed expert demonstrations. As pointed out by [1], the inverse RL problem is ill-defined and many reward functions exists that satisfy this criterion. Ziebart et al. [4], [3] introduced a max-entropy formulation for inverse reinforcement learning which follows the maximum entropy principle for estimating distribution [5]. As our method is based on [3] we will present it in more detail in the preliminaries.

Preference Learning [2] has been applied for learning controllers directly [6] or reward functions [7], [8], [9], [10] based on preferences. To the best of our knowledge, [10], [11] are most closely related to our work. [11] propose a method for combining preferences with a known reward function. They compute two stochastic policies, where one is based on the reward function and the other one is based on the preferences, and combine them by multiplying their likelihoods. [10] learn a reward function from a combination of pairwise preferences and oracle-based evaluations. They assign likelihoods for both types of human feedback and apply Bayesian inference to compute the posterior distribution based on a Gaussian process prior.

II. Preliminaries

We assume finite-horizon Markov Decision Processes (MDPs), that can be defined by a 5-Tuple $\big(\boldsymbol{s}, \boldsymbol{a}, p_t(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a}), r_t(\boldsymbol{s},\boldsymbol{a}), T\big)$ where $\boldsymbol{s}$ denotes a vector of states, $\boldsymbol{a}$ denotes a vector of actions and T denotes the time horizon. The reward function at time step t is denoted by $r_t(\boldsymbol{s},\boldsymbol{a})$ and the system dynamics by $p_t(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a})$.

Our approach is based on features of state-action pairs that are computed based on a given vector function $\boldsymbol{\phi}(\boldsymbol{s},\boldsymbol{a})$

*This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No #645582 (RoMaNS).

[1] Computational Learning for Autonomous Systems, TU Darmstadt, 64287 Darmstadt, Germany, {oleg, geri}@robot-learning.de

and have dimensionality N_ϕ. The likelihood of choosing an action $\boldsymbol{a}$ in a given state $\boldsymbol{s}$ is given by the policy $\pi_t(\boldsymbol{a}|\boldsymbol{s})$. Our method extends MaxEnt-IRL, which aims at maximizing the conditional, differential entropy of the policy, given by

$$H(\pi_t(\boldsymbol{a}|\boldsymbol{s})) = -\int_{\boldsymbol{s}} p_t(\boldsymbol{s}) \int_{\boldsymbol{a}} \pi_t(\boldsymbol{a}|\boldsymbol{s}) \log \pi_t(\boldsymbol{a}|\boldsymbol{s}) d\boldsymbol{a} d\boldsymbol{s},$$

where $p_t(\boldsymbol{s})$ denotes the state distribution at time step t.

A. Maximum Entropy IRL

Maximum Entropy IRL [3] aims at maximizing the entropy of the policy while matching its expected feature count $\tilde{\boldsymbol{\phi}}_t$ with the empirical feature counts of the expert

$$\hat{\boldsymbol{\phi}}_t = \frac{1}{N_D} \sum_{i=1}^{N_D} \boldsymbol{\phi}_t(\boldsymbol{s}_{i,t}, \boldsymbol{a}_{i,t}), \tag{1}$$

where $\boldsymbol{\phi}_t(\boldsymbol{s}_{i,t}, \boldsymbol{a}_{i,t})$ denote the features for the state and action given at demonstration i and time step t, and N_D denotes the number of demonstrations. The corresponding optimization problem is given by

$$\begin{aligned} \underset{\pi_t(\boldsymbol{a}|\boldsymbol{s})}{\text{maximize}} \quad & \sum_{t=1}^{T-1} H(\pi_t(\boldsymbol{a}|\boldsymbol{s})) \qquad (2) \\ \text{subject to} \quad & \forall_{t>1}: \int_{\boldsymbol{s},\boldsymbol{a}} p_t^\pi(\boldsymbol{s},\boldsymbol{a}) \boldsymbol{\phi}_t(\boldsymbol{s},\boldsymbol{a}) d\boldsymbol{s} d\boldsymbol{a} = \hat{\boldsymbol{\phi}}_t, \end{aligned}$$

where additional constraints specify $p_t^\pi(\boldsymbol{s},\boldsymbol{a})$ as the state-action distribution at time step t that is consistent with the policy $\pi(\boldsymbol{a}|\boldsymbol{s})$, the system dynamics $p(\boldsymbol{s},\boldsymbol{a})$ and the initial state distribution $p_1(\boldsymbol{s})$.

Solving the optimization problem (2) with the method of Lagrangian multipliers, the max-ent policy is found to be

$$\pi_t(\boldsymbol{a}|\boldsymbol{s}) = \exp\left(Q_t^{\text{soft}}(\boldsymbol{s},\boldsymbol{a}) - V_t^{\text{soft}}(\boldsymbol{s})\right), \tag{3}$$

where $V_t^{\text{soft}}(\boldsymbol{s})$ and $Q_t^{\text{soft}}(\boldsymbol{s},\boldsymbol{a})$ are *softened* state and state-action value functions. We refer to [12] for further details.

The reward function $r_t(\boldsymbol{s},\boldsymbol{a})$ is linear in the features, i.e.

$$r_t(\boldsymbol{s},\boldsymbol{a}) = \boldsymbol{\theta}_t^\top \boldsymbol{\phi_t}(\boldsymbol{s},\boldsymbol{a}),$$

where $\boldsymbol{\theta}_t$ are the Lagrangian multipliers for the feature matching constraints and found by minimizing the dual

$$\mathcal{G}(\boldsymbol{\theta}) = E_{p_1(\boldsymbol{s})}\left[V_1^{\text{soft}}(\boldsymbol{s})\right] - \sum_t \theta_t^\top \hat{\boldsymbol{\phi}}_t \tag{4}$$

based on the gradient

$$\frac{\partial \mathcal{G}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_t} = \tilde{\boldsymbol{\phi}}_t - \hat{\boldsymbol{\phi}}_t.$$

The dual function (4), which corresponds to the negative likelihood of the expert demonstrations based on the policy (3), serves as the starting point of our approach.

B. Types of Human Feedback

Our method assumes that the reward function is linear in features $\boldsymbol{\phi}(\boldsymbol{s},\boldsymbol{a})$ and we model the different types of human feedback with respect to their respective feature counts. Feature counts can correspond to the features of a single state-action pair, or to the sum over features of several state action pairs. Hence, we can express relations between partial trajectories of arbitrary–and potentially different–lengths.

1) Expert Demonstrations: Expert demonstrations are assumed to be nearly optimal with respect to an unknown reward function. They are modeled as empirical feature counts $\hat{\boldsymbol{\phi}}_t$ that are provided to the algorithm directly or computed based on state-action pairs according to (1).

2) Pairwise Preferences: Pairwise preferences are given as a set of pairs of feature counts

$$\mathcal{D}_P = \{(\boldsymbol{\phi}_{P,1}, \boldsymbol{\phi}_{P,2})_1, \ldots, (\boldsymbol{\phi}_{P,1}, \boldsymbol{\phi}_{P,2})_{N_P}\},$$

where for each preference i, the first feature count $\boldsymbol{\phi}_{P,1}^{(\text{i})}$ is assumed to produce at least as much reward as the second feature count $\boldsymbol{\phi}_{P,2}^{(\text{i})}$.

3) Star Ratings: For star ratings, we assume several feature counts to be rated on a finite, discrete scale from 1 to N_S such that feature counts with higher rating produce at least as much reward as feature counts with lower rating. The set of star ratings is given by

$$\mathcal{D}_S = \{\boldsymbol{\Phi}_S^{(1)}, \ldots, \boldsymbol{\Phi}_S^{(N_S)}\},$$

where $\boldsymbol{\Phi}_S^{(\text{i})}$ is a matrix of size $N_S^{(\text{i})} \times N_\phi$ that contains one row for each feature count with rating i.

4) Oracle-Based Evaluations: For oracle-based evaluations, we assume that the true reward is known for a given set of feature counts. This can be useful, for example when the reward is assumed to be given by a physical quantity that is hard to measure is practice, like force-based grasp quality metrics. The oracle-based evaluations are given by the set

$$\mathcal{D}_R = \{(\boldsymbol{\phi}_R, r)_1, \ldots, (\boldsymbol{\phi}_R, r)_{N_R}\},$$

where for each oracle-based evaluation i, $r^{(\text{i})}$ indicates the reward associated with feature count $\boldsymbol{\phi}_R^{(\text{i})}$.

III. Learning from Different Types of Human Feedback

Our method aims at maximizing the likelihood (4) of the expert demonstrations under policy (3) subject to the constraint of being consistent with the additional types of human feedback. We thereby assume existence of at least one expert demonstration; the other types of human feedback, however, are optional.

A. Constraint Formulations

The different types of human feedback can be treated as linear constraints on the reward function.

1) Pairwise Preferences: For each pairwise preference, the first feature count should result in higher reward than the second feature count, and hence

$$\forall_{i \in [1,N_P]}: \quad \boldsymbol{\theta}^\top \boldsymbol{\phi}_{P,1}^{(\text{i})} \geq \boldsymbol{\theta}^\top \boldsymbol{\phi}_{P,2}^{(\text{i})},$$

or slightly more concise $\boldsymbol{\Phi}_P \boldsymbol{\theta} \geq \boldsymbol{0}$, where $\boldsymbol{\Phi}_P$ is a $N_P \times N_\phi$ matrix, such that each row i is given by the transpose of its respective feature count difference $\phi_{P,1}^{(\text{i})} - \phi_{P,2}^{(\text{i})}$.

2) Star Ratings: Expressing star ratings with pair-wise preferences results in an exponentially growing number of constraints; expressing them based on the maximum and minimum reward within each rating involves subgradient-based optimization which can become slow in practice.

Instead, we framed the optimization problem by demanding that for each pair of consecutive ratings i and $i+1$, there exists a reward level $\eta^{\text{(i)}}$, that is lesser or equal than the reward of each feature count with rating $i+1$ and larger or equal than the reward of each feature count with rating i, i.e.

$$\begin{aligned} \forall_{i<N_g}, \forall_{j\in[1,N_S^{\text{(i)}}]}: &\quad \eta^{\text{(i)}} \geq \mathbf{\Phi}_{S,j}^{\text{(i)}}\theta \\ \forall_{i>1}, \forall_{j\in[1,N_S^{\text{(i+1)}}]}: &\quad -\eta^{\text{(i)}} \geq -\mathbf{\Phi}_{S,j}^{\text{(i+1)}}\theta, \end{aligned}$$

where the subscript j refers to the jth row of the respective matrix. Using this formulation, the number of constraints grows only linearly with the number of feature counts for each rating and no subgradient-based optimization is required. The reward levels η are auxiliary variables that have to be optimized. However, we will later see, that their optimal value can be computed in closed form.

3) Oracle-Based Evaluations: The constraints for the oracle-based evaluations can be framed similarly to the constraints for the pair-wise preferences and are given by

$$\mathbf{\Phi}_R\boldsymbol{\theta} = \boldsymbol{r},$$

where $\mathbf{\Phi}_R$ is a $N_R \times N_\phi$ matrix, such that each row i is given by $\phi_R^{\text{(i)}}$ and $\boldsymbol{r}$ is a vector containing the respective reward.

B. ADMM-Based Optimization

The resulting optimization problem is given by

$$\begin{aligned} \underset{\boldsymbol{\theta},\boldsymbol{\eta}}{\text{minimize}} \quad & \mathcal{G}(\theta) \\ \text{subject to} \quad & \mathbf{\Phi}_P\boldsymbol{\theta} \geq \mathbf{0}, \\ \forall_{i<N_S}, \forall_{j\in[1,N_S^{\text{(i)}}]}: \quad & \eta^{\text{(i)}} \geq \mathbf{\Phi}_{S,j}^{\text{(i)}}\theta, \\ \forall_{i>1}, \forall_{j\in[1,N_S^{\text{(i+1)}}]}: \quad & -\eta^{\text{(i)}} \geq -\mathbf{\Phi}_{S,j}^{\text{(i+1)}}\theta, \\ & \mathbf{\Phi}_R\boldsymbol{\theta} = \boldsymbol{r}. \end{aligned} \tag{5}$$

The MaxEnt-IRL dual function $\mathcal{G}(\theta)$ cannot be given in closed form as it depends on the softened state value function $V_1^{\text{soft}}(\boldsymbol{s})$. However, based on the assumption, that the isolated MaxEnt-IRL problem can be solved, we can optimize (5) using the alternating direction method of multipliers (ADMM) [13]. ADMM minimizes the augmented Lagrangian function, given by

$$\begin{aligned} \mathcal{A}(\boldsymbol{\theta},\boldsymbol{\eta},\boldsymbol{\lambda}) = G(\boldsymbol{\theta}) &+ \sum_{i=1}^{N_P}\left[\lambda_{P,i}g_{P,i}^{+}(\theta) + \frac{\rho}{2}\left(g_{P,i}^{+}(\theta)\right)^2\right] \\ &+ \sum_{i=1}^{N_S-1}\left[\sum_{j=1}^{N_S^{\text{(i)}}}\left[\overline{\lambda}_{S,j}^{(i)}\overline{g}_{S,i,j}^{+}(\theta,\eta_i) + \frac{\rho}{2}\left(\overline{g}_{S,i,j}^{+}(\theta,\eta_i)\right)^2\right]\right. \\ &\left.+ \sum_{j=1}^{N_S^{\text{(i+1)}}}\left[\underline{\lambda}_{S,j}^{(i)}\underline{g}_{S,i,j}^{+}(\theta,\eta_i) + \frac{\rho}{2}\left(\underline{g}_{S,i,j}^{+}(\theta,\eta_i)\right)^2\right]\right] \\ &+ \sum_{i=1}^{N_R}\left[\lambda_{R,i}g_{R,i}(\theta) + \frac{\rho}{2}\left(g_{R,i}(\theta)\right)^2\right], \end{aligned}$$

where $\lambda_{P,i}$, $\overline{\lambda}_{S,j}^{(i)}$, $\underline{\lambda}_{S,j}^{(i)}$ and $\lambda_{R,i}$ are Lagrangian multipliers and ρ is a penalty coefficient, penalizing the squared constraint violations. The constraint violations are given by

$$g_{P,i}^{+}(\theta) = \max(-\mathbf{\Phi}_P^{\text{(i)}}\boldsymbol{\theta}, -\rho^{-1}\lambda_P^{\text{(i)}}) \tag{6}$$

$$\overline{g}_{S,i,j}^{+}(\theta,\eta_i) = \max(\mathbf{\Phi}_{S,j}^{\text{(i)}}\boldsymbol{\theta} - \eta^{\text{(i)}}, -\rho^{-1}\overline{\lambda}_{S,j}^{(i)}) \tag{7}$$

$$\underline{g}_{S,i,j}^{+}(\theta,\eta_i) = \max(\eta^{\text{(i)}} - \mathbf{\Phi}_{S,j}^{\text{(i+1)}}\boldsymbol{\theta}, -\rho^{-1}\underline{\lambda}_{S,j}^{(i)}) \tag{8}$$

$$g_{R,i}(\theta) = r^{\text{(i)}} - \mathbf{\Phi}_R^{\text{(i)}}\boldsymbol{\theta},$$

where we refer to [14] for the derivation of the constrained violations for inequality constraints.

ADMM iteratively

1) minimizes $\mathcal{A}$ with respect to the weights $\boldsymbol{\theta}$,
$$\boldsymbol{\theta}^{\text{k+1}} = \underset{\boldsymbol{\theta}}{\arg\min} \quad \mathcal{A}(\boldsymbol{\theta},\boldsymbol{\eta}^{\text{k}},\boldsymbol{\lambda}^{\text{k}}).$$
2) minimizes $\mathcal{A}$ with respect to the reward levels $\boldsymbol{\eta}$,
$$\boldsymbol{\eta}^{\text{k+1}} = \underset{\boldsymbol{\eta}}{\arg\min} \quad \mathcal{A}(\boldsymbol{\theta}^{\text{k+1}},\boldsymbol{\eta},\boldsymbol{\lambda}^{\text{k}}).$$
3) updates the Lagrangian multipliers $\boldsymbol{\lambda}$,
$$\boldsymbol{\lambda}^{\text{k+1}} = \boldsymbol{\lambda}^{\text{k}} + \rho\boldsymbol{g}(\boldsymbol{\theta}^{\text{k+1}},\boldsymbol{\eta}^{\text{k+1}}),$$

where $\boldsymbol{\lambda}$ is a vector containing all Lagrangian multipliers and $\boldsymbol{g}(\boldsymbol{\theta},\boldsymbol{\eta})$ is vector function that returns their corresponding constraint violations.

Minimizing the augmented Lagrangian $\mathcal{A}$ with respect to $\boldsymbol{\theta}$ corresponds to minimizing the dual $\mathcal{G}$ after augmenting it with a quadratic function of $\boldsymbol{\theta}$. This is similar to solving the isolated MaxEnt-IRL problem with ℓ_2-regularization, which is commonly done in practice, and typically feasible when the unregularized optimization is feasible.

Minimizing the augmented Lagrangian $\mathcal{A}$ with respect to the reward levels $\boldsymbol{\eta}$ can be performed in closed form. The optimal reward levels are given by

$$\begin{aligned} \eta^{\text{(i)}} = &\frac{\rho\left(\sum_{j=1}^{N_S^{\text{(i)}}}\underline{\lambda}_{S,j}^{(i)}\underline{\delta}_{i,j} - \sum_{j=1}^{N_S^{\text{(i+1)}}}\overline{\lambda}_{S,j}^{(i)}\overline{\delta}_{i,j}\right)}{\sum_{j=1}^{N_S^{\text{(i)}}}\underline{\delta}_{i,j} + \sum_{j=1}^{N_S^{\text{(i+1)}}}\overline{\delta}_{i,j}} \\ &+ \frac{\left(\sum_{j=1}^{N_S^{\text{(i)}}}\mathbf{\Phi}_{S,j}^{(i)}\underline{\delta}_{i,j} + \sum_{j=1}^{N_S^{\text{(i+1)}}}\mathbf{\Phi}_{S,j}^{(i)}\overline{\delta}_{i,j}\right)\boldsymbol{\theta}}{\sum_{j=1}^{N_S^{\text{(i)}}}\underline{\delta}_{i,j} + \sum_{j=1}^{N_S^{\text{(i+1)}}}\overline{\delta}_{i,j}}, \end{aligned}$$

where $\underline{\delta}_{i,j}$ and $\overline{\delta}_{i,j}$ are indicator functions that equal one, if the corresponding constraint is active and zero otherwise. Inequality constraints are considered active, if the maximum operator for their constraint violations (6)-(8) returns its first argument.

IV. Experiments

We performed preliminary experiments on an n-by-n grid-world, where the agent can choose between the five actions *up*, *down*, *left*, *right* and *stay*. The time horizon was given by $T = 20$ and the system dynamics were stochastic such that the agent performed the chosen action with probability 0.8 and a uniformly chosen, different action otherwise. The true reward function was constructed by assigning to each state a reward that was uniformly sampled in the interval $[0, 100]$.

In the first set of experiments we tested, how the individual types of feedback can improve the result of MaxEnt-IRL. For these experiment we chose $n = 8$ and presented two expert demonstrations to the algorithm and iteratively presented additional training data of the given type. We evaluated the learned reward functions by comparing the policy that is computed according to (3) based on the learned reward function with the one that is computed based on the true reward function. After each iteration, we compute the miss-prediction probability, which is the probability of predicting a different probability than the optimal policy averaged over all states. Since even the optimal policy would have non-zero miss-prediction probability due to its stochasticity, we evaluate the learned reward function based on the miss-prediction *error*, which is the difference between the miss-prediction probability based on the learned reward function and the miss-prediction probability based on the true reward function.

We performed separate experiments for evaluating the effect of pair-wise preferences, oracle-based evaluations and star ratings. For evaluating the effect of oracle-based evaluations, we iteratively added five additional oracle-based evaluation for a sampled trajectory. For evaluating pair-wise preferences we added 100 additional pair-wise preferences at each iteration, that were produced by sampling two trajectories and comparing them on the true reward function. For evaluating star ratings we added 20 sampled trajectories at each iteration and rated them between 1 and 10 stars, such that the number of feature counts with each rating was approximately equal. For all experiments we sampled based on a uniform policy and solved the optimization problem from scratch at each iteration. Figure 1 shows the missprediction errors with 2σ-confidence for each experiment. The true reward could be closely recovered with each individual type of feedback. The amount of required training data, however, was quite large, indicating that the generated training data was not very informative. This is also reflected by the learned Lagrangian multipliers. For example, the ten thousand pair-wise preferences that we presented at the last iteration of the respective experiments, always resulted in only approximately 60 non-zero Lagrangian multipliers.

We also evaluated the performance of the optimization for large number of training data. For this experiment, we increased the gridworld to $n = 32$ and added ten thousand pair-wise preferences and the same amount of star ratings at each iteration. We did not present oracle-based evaluations, because they tend to reveal the true reward function quickly, which might help in fulfilling the remaining constraints. Figure 2 depicts the required computational time for each iteration, which was always about 40 to 60 times as large as for solving the isolated IRL problem. The experiment indicates that the ADMM-based optimization scales gracefully with the amount of pair-wise preferences and star ratings.

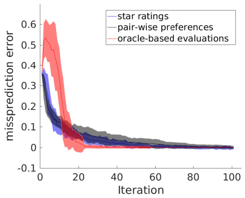

Fig. 1: Each type of human feedback could be used to substantially improve on the reward function learned from demonstration.

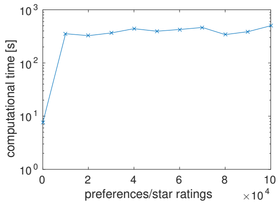

Fig. 2: The computational time scales gracefully with the amount of additional human feedback.

V. Conclusion

We presented a method for inferring reward functions from a combination of several types of training data and demonstrated its feasibility on preliminary experiments.

In future, we want to further evaluate the method on more realistic, robotic applications. Our work did not focus on generating informative training data. However, our experiments indicate, that the data generated by sampling from a uniform policy has little informative value. We therefore want to investigate, how to generate more informative training data.

References

[1] A. Y. Ng and S. J. Russell, “Algorithms for Inverse Reinforcement Learning,” in *Int. Conf. on Machine Learning (ICML)*, 2000.

[2] J. Fürnkranz and E. Hüllermeier, “Preference learning: An introduction,” in *Preference learning*. Springer, 2010, pp. 1–17.

[3] B. D. Ziebart, J. A. Bagnell, and A. K. Dey, “Modeling Interaction via the Principle of Maximum Causal Entropy,” in *Int. Conf. on Machine Learning*, 2010.

[4] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, “Maximum Entropy Inverse Reinforcement Learning,” in *AAAI Conf. on Artificial Intelligence*, 2008.

[5] E. T. Jaynes, “Information Theory and Statistical Mechanics,” *Physical review*, vol. 106, no. 4, p. 620, 1957.

[6] A. Wilson, A. Fern, and P. Tadepalli, “A bayesian approach for policy learning from trajectory preference queries,” in *Advances in Neural Information Processing Systems*, 2012.

[7] R. Akrour, M. Schoenauer, M. Sebag, and J.-C. Souplet, “Programming by feedback,” in *International Conference on Machine Learning*, no. 32. JMLR. org, 2014, pp. 1503–1511.

[8] C. Daniel, M. Viering, J. Metz, O. Kroemer, and J. Peters, “Active reward learning,” in *Proceedings of Robotics Science & Systems*, 2014.

[9] C. Wirth, J. Fürnkranz, and G. Neumann, “Model-Free Preference-Based Reinforcement Learning,” in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[10] A. Kupcsik, D. Hsu, and W. S. Lee, “Learning dynamic robot-to-human object handover from human feedback,” *arXiv preprint arXiv:1603.06390*, 2016.

[11] S. Griffith, K. Subramanian, J. Scholz, C. Isbell, and A. L. Thomaz, “Policy shaping: Integrating human feedback with reinforcement learning,” in *Advances in Neural Information Processing Systems*, 2013.

[12] B. D. Ziebart, “Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy,” Ph.D. dissertation, Machine Learning Department, Carnegie Mellon University, Dec 2010.

[13] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[14] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.