Robust Behavior Cloning with Adversarial Demonstration Detection

Mostafa Hussein[1], Brendan Crowe[2], Madison Clark-Turner[1], Paul Gesel[1], Marek Petrik[3], and Momotaz Begum[1]

***Abstract*— Imitation learning (IL) frameworks in robotics typically assume that a domain expert's demonstration always contains a correct way of doing the task. Despite its theoretical convenience, this assumption has limited practical values for an IL-powered robot in real world. There are many reasons for an expert in the real world to provide demonstrations that may contain incorrect or potentially unsafe way of doing a task. In order for IL-powered robots to work in the real world, IL frameworks need to detect such adversarial demonstrations and not learn from them. This paper proposes an IL framework that can autonomously detect and remove adversarial demonstrations, if they exist in the demonstration set, as it directly learns a task policy from the expert. The proposed framework that we term Robust Maximum Entropy behavior cloning (R-MaxEnt) learns a stochastic model that maps states to actions. In doing so, R-MaxEnt solves a *minmax* problem that leverages the entropy of the model to assign weights to different demonstrations while assigning poor weights to adversarial samples. Our empirical results show that R-MaxEnt outperforms the existing IL approaches in both real and simulated robotics tasks.**

I. Introduction

The main appeal of the imitation learning (IL) paradigm in robotics, also known as learning from demonstrations or programming by demonstrations, is its promise to enable lay users (hereafter termed as domain experts) with the ability to train robots new skills from demonstrations [4], [8], [13]. IL algorithms have experienced tremendous recent progress in learning task policies from demonstrations in controlled laboratory settings [28], [35] or simulated environments [23], [25], [29]. IL algorithms, however, need the following two key characteristics to enable IL-powered robots learning realistic tasks in natural human environments from the demonstrations of domain experts:

- Dealing with Adversarial demonstrations: Domain experts may inadvertently provide one or more demonstrations, among many correct ones, which show a potentially unsafe policy. Demonstrators' fatigue or lack of knowledge of robotics/programming may also contribute to such adversarial demonstrations. A robust IL algorithm needs to detect and eliminate such adversarial samples from the demonstration set before learning the task policy.
- Sample efficiency: Having a simulator for every task to be taught to a robot by a domain expert is infeasible and, most importantly, invalidate the reason why IL appeals to domain experts. Therefore, IL algorithms need to learn only from a handful of demonstrations and without a simulator that can generate an unlimited number of training samples.

To the best of our knowledge, there are no IL methods that feature both of the characteristics above. A typical assumption in most existing IL algorithms is that all expert demonstrations are reliable and trustworthy. There exist a handful of work that can deal with sub-optimal or noisy demonstrations [22], [28], [45], [14], [44], [33]. But we are interested in tackling adversarial demonstrations. Adversarial demonstrations are those that do not follow the task definition and are structurally wrong. For example, in the context of teaching a robot how to make a cup of tea, an adversarial demonstration would be the one where water or tea-bag is not added to the cup. No existing IL algorithm addresses the issues with adversarial demonstration. Most existing IL algorithms also require many training samples, often supplied by a simulator, to learn a policy.

This paper proposes to directly learn a task policy from a handful of demonstrations while detecting and discarding adversarial samples, if any, from the demonstration set. The proposed algorithm (R-MaxEnt), learns a stochastic model of the task policy in a supervised manner through constraining feature expectation matching between the learned policy and the demonstrated policy. R-MaxEnt analyzes the entropy of different policy models in the model space, and the entropy contributed by different demonstrations to each model. It then leverages the maximum entropy principle (MEP) [3], [24] to choose the model with the maximum entropy while setting weights to different demonstrations that can help to identify adversarial samples. We demonstrate that R-MaxEnt is more robust and sample efficient than existing IL approaches in classical control tasks in the OpenAi-gym simulator [11]. We also demonstrate R-MaxEnt's ability to teach a Yumi robot an activity of daily living: *tea-making*.

II. Related Work

There are two main types of IL algorithms: Behavior Cloning (BC) and Inverse Reinforcement Learning (IRL). BC methods learn a mapping from states to actions as a supervised learning problem [34]; BC is considered conceptually simple and theoretically sound [42]. The main criticism of BC in its current state is the covariance shift [36], [37], where small inaccuracies of the learned model compound over time, and can lead to states that are very different from

[1] Cognitive Assistive Robotics Lab, University of New Hampshire, {mhussein, mbc2004, pac48, mbegum}@cs.unh.edu; [2] Department of Statistics, University of New Hampshire, bjc1041@wildcats.unh.edu; [3] Department of Computer Science, University of New Hampshire, mpetrik@cs.unh.edu

the ones encountered during training. In recent years, BC-based end-to-end IL realized through deep neural networks achieved success in autonomous driving [10]. There is, however, no known general framework that can be applied to learn different robotics tasks. In addition to that, learning neural network-based task-policies has known issues such as lack of convergence guarantee and sample inefficiency. There is no BC-based IL method in the current literature that can deal with adversarial demonstrations.

Classical IRL-based IL methods first learn the demonstrator's reward function and then use it to learn a policy that maximizes the total reward [2], [31], [38], [41], [45]. Through connecting IRL with generative adversarial networks (GANs) [16], [20], several recent papers [17], [18], [23], [25] achieve higher expected return than the classical IRL based IL methods in simulated environments.

Despite their elegance and the capability of dealing with sub-optimal demonstrations, IRL-based IL methods have rarely been used in realistic robotics task [17]. The main reason behind this is, IRL algorithms require millions of samples during training to converge even for the simplest control tasks [27]. The alternative is to have a full transition probability knowledge [45].

The default assumption in these methods is that all demonstrations are correct and should be considered for policy learning [17], [18], [23], [25]. Even though these methods use adversarial networks (i.e. GANs) for policy learning, the definition of adversaries in GANs is markedly different from the adversaries in demonstrations that we are interested in exploring in this paper. For example, GANs train a generative model G to estimate the distribution of the expert data, and a discriminator D that tries to differentiate between the actual expert data coming from the demonstration set and the data coming from the generator G. At the convergence point, the generator will produce data close enough to the expert data that the discriminator will not differentiate from the expert demonstrations. In IRL methods, the generator G typically represents the policy, and the D represents the reward function. However these recent approaches [17], [18], [23], [25] use the given demonstrations as a reference to discriminate the data coming from the generator to come closer to the optimal policy. These approaches do not have any mechanism to deal with the present of adversarial demonstration in the dataset.

A recent line of research in [29], [43], [44] considers a limited amount of random noise in a demonstration set and proposes a a various approaches to detect such noisy demonstrations. However, limited random noise in demonstrations is different from adversarial demonstrations that always lead to a potentially unsafe policy. Beside they still leverage the advantage of using more samples through interaction with the simulator

A few other IRL based approaches [21], [40] used "failed" demonstration, in addition to correct ones, to train the model. However, they assume that these failed demonstrations are labeled a priori and do not detect them autonomously.

III. Feature Expectation Matching

As the basic framework, we assume the Markov decision process $(\mathcal{S}, \mathcal{A}, P, r, \rho_0)$ with the stochastic shortest path objective (assuming some terminal states) [7], where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and $\rho_0 \in \Delta^{\mathcal{S}}$ represents the distribution over the initial state. Here, $\Delta^{\mathcal{S}}$ denotes the probability simplex overt the set $\mathcal{S}$. The unknown transition probabilities are $P : \mathcal{S} \times \mathcal{A} \to \Delta^{\mathcal{S}}$ and the unknown rewards are $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

Our goal is to compute a randomized policy $\pi : \mathcal{S} \to \Delta^{\mathcal{A}}$ that matches the expert's policy $\tilde{\pi} : \mathcal{S} \to \Delta^{\mathcal{A}}$ of the expert. The policy π is unknown and, instead, must be estimated from a set of demonstrations. The demonstrations $\mathcal{D} = (s_i, a_i)_{i=1,\dots,Q}$ consist of Q states and expert's actions for these states such that $a_i \sim \tilde{\pi}(\cdot, s_i)$. We also assume that there is a distribution $\tilde{p} \in \Delta^S$ over the states that represents the expert's probability of visiting the state. In practice, we assume that this distribution is uniform over the states in $\mathcal{D}$.

In most IL algorithms, we try to represent the task using a set of n features $f_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}, i = 1, 2, \dots, n$ defined for state-action pairs that contain enough information to enable generalization from the demonstrations to the entire state space. The important question in IL is *"How can the learner match the expert's demonstrations?"* Many approaches have been introduced and studied in the IL community. Some of the most-successful methods have been based on *feature expectation matching (FEM)* [13], [33], [45] where the goal is compute a policy π that satisfies the following equality:

$$\mathbb{E}_{\tilde{\pi}}[f_i] = \mathbb{E}_{\pi}[f_i],\ i = 1, 2, \dots, n\ . \tag{1}$$

The feature expectations are computed with respect to the policy indicated by the subscript and the state probabilities as follows:

$$\mathbb{E}_{\tilde{\pi}}[f_i] = \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \tilde{p}(s)\tilde{\pi}(a|s) f_i(s, a)$$
$$\mathbb{E}_{\pi}[f_i] = \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \tilde{p}(s)\pi(a|s) f_i(s, a)\ .$$

Note that we assume we can compute a policy that matches the expert's state distribution $\tilde{p}$ sufficiently well.

The FEM problem in (1) is ill-defined because it can be satisfied by many policies [45]. To select a policy π that is most-likely to generalize beyond the demonstrations, we employ the principle of maximum entropy [3], [24] and solve the following optimization problem to compute the policy π:

$$\begin{aligned} \max_{\pi \in \mathbb{R}^{\mathcal{S} \times \mathcal{A}}} \quad & H(\pi) \equiv -\sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \tilde{p}(s)\pi(a|s) \log \pi(a|s) \\ \text{s.t.} \quad & \mathbb{E}_{\tilde{\pi}}[f_i] - \mathbb{E}_{\pi}[f_i] = 0 \quad i = 1, \dots, n \\ & \sum_{a \in \mathcal{A}} \pi(a|s) - 1 = 0 \quad \forall\ s \in \mathcal{S} \end{aligned} \tag{2}$$

Here, $H(\pi)$ denotes the causal entropy of the policy π. Using the standard convex duality arguments, we can see that the optimal solution $\pi^\star$ to (2) must satisfy, for some $\lambda \in \mathbb{R}^N$,

that [3], [6], [15]:

$$\pi^\star(a|s) = (z_\lambda(s))^{-1} \cdot \exp\left(\sum_{i=1}^{N} \lambda_i f_i(s,a)\right) \ , \qquad (3)$$

where $z_\lambda(s) = \sum_{a\in\mathcal{A}} \exp\left(\sum_{i=1}^{N} \lambda_i f_i(s,a)\right)$ is a normalization constant. Equation (3) indicates that it may be possible to match the expert's policy even with a limited number of demonstrations because the policy $\pi^\star$ depends only on a small number of features.

The FEM algorithm in (2) can be derived independently from the maximum likelihood principle [6], [15]. In fact, one can readily show that the FEM optimization in (2) is the dual formulation of the maximal likelihood solution to the multinomial logistic regression [9]. The derivation, which we omit due to the lack of space, follows the derivation for other related methods [15].

As mentioned above, our goal is to design an IL algorithm that is suitable for real-life robotics applications where we do not have a simulation for each task and can only generate a limited number of demonstrations. The FEM algorithm in (2) does not need to access the simulator. However, the algorithm must be robust enough not to fail even when some of the demonstrations are incorrect or even adversarial. In the next section, we discuss an algorithm that is statistically robust to such errors in demonstrations.

A. Connection to Max-Ent IRL Approach

Now, we discuss the relationship between FEM, described in Section III, and the popular Max-Ent IRL [45] which also matches feature expectations while maximizing entropy. Although FEM is a pure behavioral cloning technique while Max-Ent IRL is an inverse reinforcement learning technique, these two methods are closely related. FEM can be seen as a special case of Max-Ent IRL with a simplifying assumption that the state distribution $\tilde{p}(s,d)$ for the computed policy π is the same as for expert's policy $\tilde{\pi}$. This assumption is likely to be satisfied when the agent can match the expert's policy closely, and the transition probabilities are mostly deterministic.

Our algorithms are based on FEM instead of Max-Ent IRL for two main reasons. First, the behavioral cloning technique is easier to apply to robotics domains because it does not require accessing a simulator. Second, FEM involves a simpler optimization problem than Max-Ent IRL, making it more convenient to develop and analyze methods that can detect adversarial demonstrations. However, it is important to note that the approach that we present in this paper can be generalized to Max-Ent IRL and other related BC and IRL algorithms.

IV. Robust Maximum Entropy Behavior Cloning (R-MaxEnt)

In this section, we propose methods that add robustness to our model and detect any adversarial or incorrect demonstration. We assume that, the dataset $\mathcal{D} = ((s_{di}, a_{di})_{i=1,\ldots,Q_d})_{j=1,\ldots,D}$ comprises D individual demonstrations, or trajectories. Each one of these trajectories is either considered to be generated by the actual expert, or is adversarial and should be discarded.

To reduce the sensitivity to incorrect demonstrations, we introduce a variable $w \in [0,1]^D$ which assigns an importance weight to each demonstration. The goal is to give the adversarial demonstration the minimum possible weight and to give the correct demonstration a higher weight automatically through our model. We assume that $M = \sum_{d=1}^{D} w_d$ represents the assumed minimum number of demonstrations that we can trust and is known.

We achieve statistical robustness by relying on the maximum entropy principle [3], [24]. In particular, we exclude any demonstrations with high entropy with respect to other demonstrations. The entropy is, in other words, used to measure the consistency of the expert demonstrations. An adversarial demonstration adds incorrect (or "random") information to the model, which increases its entropy. We limit the impact of this noise by assigning a lower weight w_d to a demonstration that significantly increases the learned entropy. The goal of the formulation is to ensure that the optimal $w^\star$ is $w_d^\star = 0$ when the demonstration d is adversarial and $w_d^\star = 1$ when the demonstration is correct.

To incorporate the weights w into the FEM algorithm, we need to assume that the expert's policy $\tilde{\pi} : \mathcal{S}\times\{1,\ldots,D\} \to \Delta^{\mathcal{A}}$ and the state distribution $\tilde{p} : \{1,\ldots,D\} \to \Delta^{\mathcal{S}}$ are parameterized by the demonstration. This will make it possible to essentially ignore some of the demonstrations. Next, we replace $\tilde{\pi}$ and $\tilde{p}$ in (2) by their weighted versions defined as follows:

$$\tilde{\pi}_w(a|s) = \frac{1}{M}\sum_{d=1}^{D} w_d \cdot \tilde{\pi}(a|s,d)$$

$$\tilde{p}_w(s) = \frac{1}{M}\sum_{d=1}^{D} w_d \cdot \tilde{p}(s,d) \ .$$

In essence, we allow the expert policies to differ among demonstrations and aim to find the one policy that is most consistent among them.

Then, optimizing for weights w that minimize the entropy in (2), leads to the saddle point problem in Fig. 1. We call the method R-MaxEnt. Note that the inner maximization problem is convex (concave objective function) and, therefore, it can be replaced by its dual minimization problem. The result is the non-convex quadratic problem depicted in Fig. 2.

We solve the non-convex quadratic problem in Fig. 2 using the Sequential Quadratic Programming (SQP) algorithm [1]; see, for example, Chapter 18 of Nocedal and Wright [32]. The SQP algorithms generalizes the Newton's method to constrained optimization problems. In each iteration, the Hessian of the Lagrangian function is approximated in a quasi-Newton style. The algorithm then solves the resulting

[1]Implementation: `https://www.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html\#bsgppl4`

$$
\begin{aligned}
\min_{w\in\mathbb{R}^D}\max_{\pi\in\mathbb{R}^{\mathcal{S}\times\mathcal{A}}} \quad & -\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\pi(a|s)\log\pi(a|s)\sum_{d=1}^{D} w_d\cdot\tilde{p}(s,d) \\
\text{s. t.} \quad & \sum_{d=1}^{D} w_d \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} f_i(s,a)\tilde{p}(s,d)\Big(\pi(a|s)-\tilde{\pi}(a|s,d)\Big)=0, \quad i=1,\dots,N \qquad [\pi] \\
& \sum_{a\in\mathcal{A}}\pi(a|s)-1=0, \quad \forall s\in\mathcal{S} \qquad [\pi] \\
& \sum_{d=1}^{D} w_d = M, \quad w_d\geq 0, \quad \forall d\in\mathcal{D}, \quad w_d\leq 1 \quad \forall d=1,\dots,D \qquad [w]
\end{aligned}
\tag{4}
$$

Fig. 1. R-MaxEnt saddle point problem. The gray label in square brackets indicates which variable the constraint applies to.

$$
\begin{aligned}
\min_{w\in\mathbb{R}^D,\lambda\in\mathbb{R}^N} \quad & \Lambda(\lambda,w)\equiv -\frac{1}{M}\sum_{d=1}^{D} w_d\Big(-\sum_{s\in\mathcal{S}}\tilde{p}(s,d)\log z_\lambda(s)+\sum_{i=1}^{N}\lambda_i\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\tilde{\pi}(a|s,d)f(s,a)\Big) \\
\text{s. t.} \quad & \sum_{d=1}^{D} w_d = M, \quad w_d\geq 0 \quad \forall d\in\mathcal{D}, \quad w_d\leq 1 \quad \forall d\in\mathcal{D}
\end{aligned}
\tag{5}
$$

Fig. 2. R-MaxEnt minimization problem equivalent to Fig. 1.

quadratic program finds the next iteration using the line search procedure. This algorithm may not converge to the global minimum, but we find it to perform very well experimentally.

V. Simulated Experiments and Results

A. Experiments with OpenAI-Gym Simulator

In order to demonstrate R-MaxEnt's ability to detect adversarial demonstrations, we now experiment with several simulated tasks.

Although R-MaxEnt is a BC approach of policy learning that does not learn any implicit/explicit reward function, we compared its performance against three recent IRL approaches: Generative Adversarial Imitation Learning (GAIL) [22] and [23] with two different objective functions; (1) Linear cost function from [2] (FEM); (2) Game-theoretic apprenticeship learning (GTAL): the algorithm of [23] using the cost function from [41]. The reason for this is that, similarly to R-MaxEnt, these IRL approaches also use/build on maximum entropy and FEM concepts. However, note that all of these existing IRL approaches require a simulator to improve the learned policy.

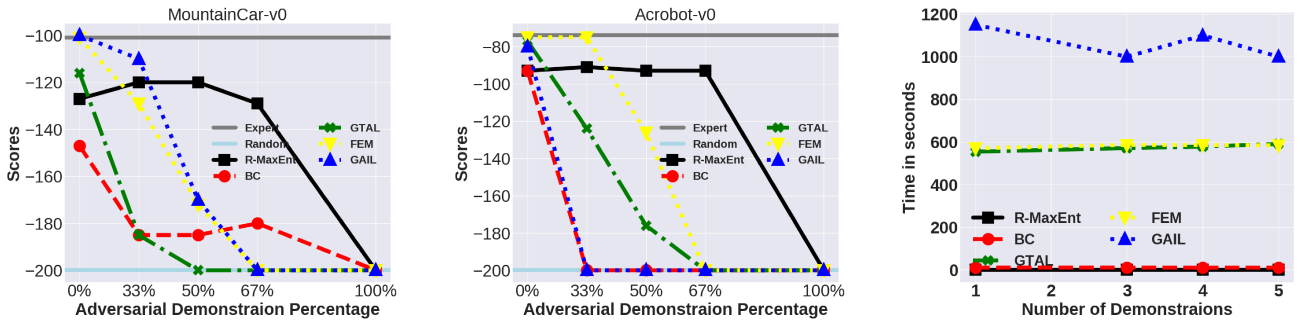
We run our algorithm on the classical control tasks *Mountain-Car* [30] and *Acrobot* [19] in the OpenAi-Gym simulator [11]. Both tasks have a continuous state space and discrete actions. As the baseline, we compare to the established BC method [5], which models π_{BC} using a neural network with parameters θ_{BC}. We find these parameters using maximum-likelihood estimation: $\theta_{BC} = \arg\max_\theta \prod_{(s,a)\in D}\pi_{BC}(a|s)$. With a given dataset of state-action pairs, we split the dataset as 70% for training and 30% validation data. We train the policy with supervised learning using ADAM [26], until the validation error stops decreasing. As we mentioned earlier, the dual of maximizing the entropy is maximizing the likelihood of the given dataset. Keep in mind that the IRL approaches has the advantage of using the simulator in the training phase to generate more samples to improve the accuracy (it used exactly 5000 samples in each of the 300 iterations, a total of $1,500,000$ samples) . We leveraged the openly available source code[2] for conducting these experiments.

We generate the expert data using TRPO [39] on the optimal cost functions. For the adversarial demonstrations, we manipulate the actions of the expert data and the corresponding state. For example, in the mountain car, we had two actions: 0 and 1. The adversarial demonstrations are generated as follows. If the optimized policy takes action 0 in a given state, we replace it with an action 1, and vice versa to generate the next state. By doing this we are building an actual adversarial demonstration that does not follow the optimal policy, rather than simply a sub-optimal demonstration.

To ensure a fair comparison, we use the same experimental settings as in [22], including the exact neural network architectures for the policies, the features, and the optimizer parameters for TRPO [39] for all of the algorithms except ours, which does not use a neural network.

1) Adversarial Detection and Accuracy: The main goal for our framework is to detect those adversarial demonstrations in the dataset and give higher weights only to the "correct" demonstrations. Fig. 3(a) and 3(b) summarize the performance of different algorithms, under varying fraction of expert/adversarial demonstrations (6 demonstrations in

[2]https://github.com/openai/imitation



(a) Results of *Mountain-Car* task.

(b) Results of *Acrobot* task.

(c) Comparison of the time required for convergence.

Fig. 3. Results of simulation experiments.

total). At the first data point (we have 0 adversarial demonstrations and 6 correct demonstrations), we can see that R-MaxEnt and BC perform worse than the IRL approaches, the reason behind that is, the IRL approaches uses more samples through interaction with the simulator to calculate the policy. On the other hand, R-MaxEnt and BC can only use the given demonstrations. However, starting from the second data point (we have 2 adversarial demonstrations and 4 correct demonstrations), we can see the power of R-MaxEnt as it detects the adversarial demonstration d' in the dataset and removes it (set its $w_{d'} = 0$ while for each "correct" demonstration d, we have $w_d = 1$). Therefore, the policy π is computed with the correct demonstrations only. For the other algorithms, we see a decrease in the accuracy with more adversarial demonstrations added to the dataset. Finally, when there are only adversarial demonstrations, R-MaxEnt learns a random like policy.

2) Time complexity: We compared the time required to train each algorithm to generate the policy using different number of demonstrations. As shown in Fig 3(c), R-MaxEnt and BC require much less time to converge, as they directly compute the policy. On the other hand, the IRL approaches spend much more time in the training phase.

VI. Experiment with the Yumi Robot: Tea-Making Task

We evaluated the performance of R-MaxEnt in learning the *tea-making* task from human demonstrations. The demonstration set was generated through an IRB-approved user study with three participants. Each participant performed 12 trials with a total 36 demonstrations of the task Fig. 4. Participants were asked to make tea according to their own preferred sequence but using all the ingredients. This resulted in demonstrations with varying number and sequence of atomic actions, all of which ended up with a cup of tea.

1) Task definition: There are seven atomic actions in the task, A : { *Turn on/off the oven, Add water, Add sugar, Add milk, Add tea bag, Stir, Do nothing*}. We designed a perception module using a 3D CNN namely, I3D [12], to identify all of these atomic actions. The pre-trained I3D network in the perception module was fine-tuned using 80% split of the team-making dataset. Based on the seven atomic actions we define the task in terms of a set of $2^7 = 128$ states, as shown in Table I. Here, the role of the pre-trained I3D is to give us a clear state representation, the same way a simulator provides for the control tasks. Our objective here is to investigate whether R-MaxEnt can detect the adversarial demonstration and generate an acceptable policy.

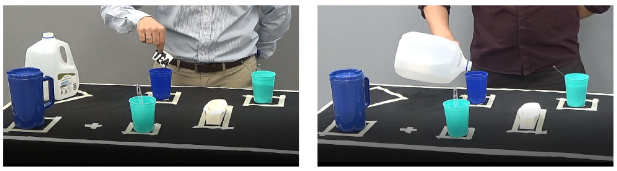


Fig. 4. Demonstration data collection: Participants making tea.

Feature	S_0	S_1	S_{20}	S_{62}	...	S_{127}
Oven is on	0	1	0	0	...	1
Oven is off	0	0	0	1	...	1
Cup has milk	0	0	1	1	...	1
Cup has sugar	0	0	0	1	...	1
Cup has water	0	0	0	1	...	1
Stir is done	0	0	0	0	...	1

TABLE I: State space for the *Tea Making* task.

2) Feature selection: Unlike the classical control tasks where we get the state and feature information directly from the simulator, here we need to generate a set of features automatically. We used Task Precedence Graphs (TPG) (also known as directed acyclic graph, DAG) for this purpose. TPG

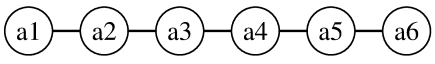


Fig. 5. Demonstration 1

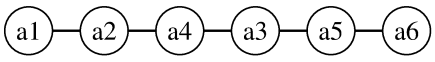


Fig. 6. Demonstration 2

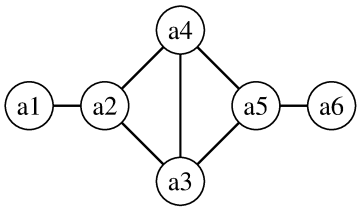


Fig. 7. TPG generated from two demonstrations.

generates a set of temporal features that can represent the sequential task [1].

For example, if we have two demonstrations with atomic action sequences shown in Figs. 5 and 6, we can form one coherent TPG, according to [1], as shown in Fig.7. Once a TPG is generated, we can collect all spatio-temporal features that it supports. For example, the TPG in Fig.7 supports the following spatio-temporal features:

- a_1, a_2 happens before a_4 and a_3
- a_5, a_6 happens after a_4 and a_3

All such features are used to train the model.

3) Adversarial Demonstration Injection: In the context of tea-making task a demonstration is considered as a correct one when i) all ingredients (i.e. tea-bag, hot water, milk and sugar) are used for making the tea, irrespective of the order they are added, and ii) *stir* is the terminal. Accordingly, an adversarial demonstration is defined as the one that misses any ingredient. We create a number of such adversarial demonstrations and include them in the dataset.

4) Results for the Tea Making Task: At first, we evaluate R-MaxEnt with only a set of correct demonstrations. Afterwards, we gradually add adversarial samples in the demonstration set. The accuracy of R-MaxEnt is calculated as the number of times, expressed in percentage, it infers the correct actions, i.e. adding one of the four ingredient or performing the *stir* action. Performance is compared with a BC algorithm described in the previous experiment. Unlike the classical control task, no comparison was performed with any IRL algorithm since they require a simulator to generate more samples during the training, which is infeasible for the realistic tea-making task.

Results are shown in Fig. 8. The figure shows that both algorithms have a 100% accuracy with no adversarial sample. As we add adversarial demonstrations, the accuracy of BC decreases while R-MaxEnt's accuracy holds steady and only decreases when adversarial demonstrations outnumber correct ones.

5) Task reproduction by a robot: We used kinesthetic teaching to teach the robot how to execute different atomic actions Fig. 9(a). The robot then autonomously executed different actions as suggested by the model to perform the complete tea-making task, as shown in Fig. 9(b) (please see the attached video).

VII. Conclusion and Future Work

We presented a novel IL method that automatically assigns weights to expert's demonstrations to excludes adversarial and noisy ones. The method leverages the model's entropy to determine which demonstrations that are inconsistent with the rest. Our algorithm achieves superior performance, time complexity, and sample efficiency compared to other BC and IRL approaches when adversarial demonstrations are present. Finally, we note that our general approach is simple and can be readily combined with IL algorithms or used as a pre-processing step. Future work should extend our method to continuous actions and develop a more efficient optimization algorithm.

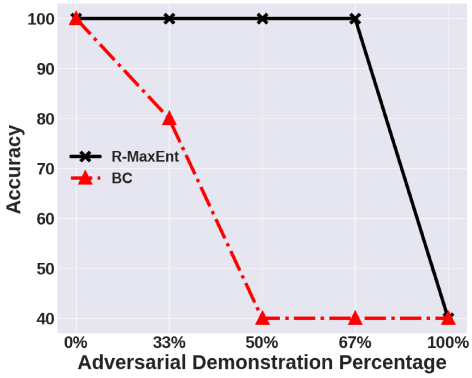
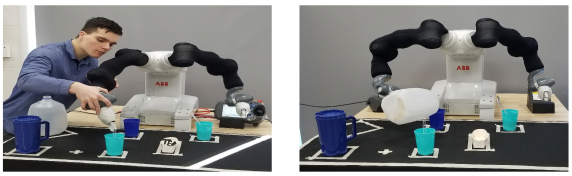


Fig. 8. Tea making task results .



(a) Kinesthetic teaching of the tea-making task to a Yumi robot. (b) Yumi robot performing *Adding Milk* action.

Fig. 9. Yumi robot learning and performing one of the task actions.

Acknowledgments

This work was supported in part by the National Science Foundation grants IIS-1830597, IIS-1717368, IIS-1815275.

References

[1] Tanveer Abbas and Bruce A MacDonald. Generalizing topological task graphs from multiple symbolic demonstrations in programming by demonstration (pbd) processes. In *2011 IEEE International Conference on Robotics and Automation*, pages 3816–3821. IEEE, 2011.

[2] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, page 1. ACM, 2004.

[3] Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

[4] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[5] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129, 1995.

[6] Adam Berger, Stephen A Della Pietra, and Vincent J Della Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

[7] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

[8] A Billard, S Calinon, and R Dillmann. Handbook of robotics, chapter learning from humans, 2016.

[9] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[10] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[11] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[12] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[13] Sonia Chernova and Andrea L Thomaz. Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(3):1–121, 2014.

[14] Sungjoon Choi, Kyungjae Lee, and Songhwai Oh. Robust learning from demonstration using leveraged gaussian processes and sparse-constrained optimization. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 470–475. IEEE, 2016.

[15] Miroslav Dudik. Maximum entropy density estimation and modeling geographic distributions of species, 2007.

[16] Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016.

[17] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning*, pages 49–58, 2016.

[18] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[19] Alborz Geramifard, Christoph Dann, Robert H Klein, William Dabney, and Jonathan P How. Rlpy: a value-function-based reinforcement learning framework for education and research. 2015.

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[21] Daniel H Grollman and Aude Billard. Donut as i do: Learning from failed demonstrations. In *2011 IEEE International Conference on Robotics and Automation*, pages 3804–3809. IEEE, 2011.

[22] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in neural information processing systems*, pages 4565–4573, 2016.

[23] Jonathan Ho, Jayesh Gupta, and Stefano Ermon. Model-free imitation learning with policy optimization. In *International Conference on Machine Learning*, pages 2760–2769, 2016.

[24] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

[25] Kee-Eung Kim and Hyun Soo Park. Imitation learning via kernel mean embedding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[32] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[27] Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. *arXiv preprint arXiv:1809.02925*, 2018.

[28] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[29] Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. In *Advances in Neural Information Processing Systems*, pages 3812–3822, 2017.

[30] Andrew William Moore. Efficient memory-based learning for robot control. 1990.

[31] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[33] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *arXiv preprint arXiv:1811.06711*, 2018.

[34] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

[35] Rouhollah Rahmatizadeh, Pooya Abolghasemi, Aman Behal, and Ladislau Bölöni. Learning real manipulation tasks from virtual demonstrations using lstm. *arXiv preprint arXiv:1603.03833*, 2016.

[36] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668, 2010.

[37] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

[38] Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 101–103, 1998.

[39] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

[40] Kyriacos Shiarlis, Joao Messias, and SA Whiteson. Inverse reinforcement learning from failure. 2016.

[41] Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in neural information processing systems*, pages 1449–1456, 2008.

[42] Umar Syed and Robert E Schapire. A reduction from apprenticeship learning to classification. In *Advances in neural information processing systems*, pages 2253–2261, 2010.

[43] Voot Tangkaratt, Bo Han, Mohammad Emtiyaz Khan, and Masashi Sugiyama. Variational imitation learning with diverse-quality demonstrations. In *International Conference on Machine Learning*, pages 9407–9417. PMLR, 2020.

[44] Jiangchuan Zheng, Siyuan Liu, and Lionel M Ni. Robust bayesian inverse reinforcement learning with sparse behavior noise. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[45] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Association for the Advancement of Artificial Intelligence (AAAI)*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.