Score-based Inverse Reinforcement Learning

Layla El Asri
Orange Labs & Maluuba
Montréal, Canada
layla.elasri@maluuba.com

Bilal Piot
Univ. Lille, CNRS, Centrale
Lille, Inria UMR 9189 -
CRIStAL
Lille, France
bilal.piot@univ-lille1.fr

Matthieu Geist
UMI 2958,
GeorgiaTech-CNRS,
CentraleSupélec, Université
Paris-Saclay
Metz, France
matthieu.geist@centralesupelec.fr

Romain Laroche
Orange Labs
Issy les Moulineaux, France
romain.laroche@orange.com

Olivier Pietquin
Univ. Lille, CNRS, Centrale
Lille, Inria UMR 9189 -
CRIStAL
Institut Universitaire de France
Lille, France
olivier.pietquin@univ-lille1.fr

ABSTRACT

This paper reports theoretical and empirical results obtained for the score-based Inverse Reinforcement Learning (IRL) algorithm. It relies on a non-standard setting for IRL consisting of learning a reward from a set of globally scored trajectories. This allows using any type of policy (optimal or not) to generate trajectories without prior knowledge during data collection. This way, any existing database (like logs of systems in use) can be scored *a posteriori* by an expert and used to learn a reward function. Thanks to this reward function, it is shown that a near-optimal policy can be computed. Being related to least-square regression, the algorithm (called SBIRL) comes with theoretical guarantees that are proven in this paper. SBIRL is compared to standard IRL algorithms on synthetic data showing that annotations do help under conditions on the quality of the trajectories. It is also shown to be suitable for real-world applications such as the optimisation of a spoken dialogue system.

Keywords

Reinforcement Learning, Inverse Reinforcement Learning, Markov Decision Processes, Learning from Demonstration, Spoken Dialogue Systems

1. INTRODUCTION

Reinforcement Learning (RL) has seen a growing interest this last decade both from a theoretical and a practical point of view, especially in the domain of Human-Machine Interaction (HMI) where it is now part of the state of the art [20, 13, 24]. In this paradigm, an agent steps from states to states in an environment of unknown dynamics by selecting actions. The action selection process is encoded in a so-called *policy*, which is a mapping between states and actions. Learning occurs via a reward function providing a numerical feedback to the agent after each transition from one state to another. An optimal policy, which should be the result of the learning process, leads to the highest cumulative reward in average. This paradigm is very attractive to learn optimal behaviors in a model-free manner, which allows dealing with complex systems (such as HMI where the human is in the loop and hard to model). Yet, correctly defining the reward function so that it leads to the desired behavior remains a tricky task. For this reason, Inverse Reinforcement Learning (IRL) [19] received a lot of attention recently, including in HMI [15, 5, 16], and many algorithms can now be found in the literature [1, 6, 12]. IRL consists of finding the reward optimised by an expert that provides optimal demonstrations. Yet, these algorithms require numerous expert demonstrations, some of them have to solve the direct RL problem many times [1] or also require many non-expert (and if possible random) trajectories to estimate the dynamics [6]. It is often hard and potentially impossible to comply with these requirements, especially when dealing with human experts. Collecting enough expert demonstrations is time consuming and expensive. Besides, it is also hard for a person to consciously act sub-optimally on demand and, above all, randomly so as to cover the widest range of possible behaviors. This is even more difficult when facing other persons (in the case of HMI problems).

In this paper, we study a different setting for IRL based on scored trajectories and especially an associated algorithm based on the minimisation of the distance between actual and predicted scores [7]. In this paradigm, a database of trajectories is first collected. Each trajectory can be obtained by applying any policy (optimal or not). Each trajectory is then annotated *a posteriori* in terms of global performance by an external expert, who might be error-prone. This is a very practical setting where no expert is needed during the collection of trajectories and there is no requirement on their quality. The expert's job is reduced since s/he only has to rate trajectories instead of providing optimal decisions in each state. It also applies in tasks where only subjective ratings can be obtained, which is often the case for HMI [14],

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

or where a set of unreliable data already exists (such as Mechanical Turk data collection).

Compared to preference-based reinforcement learning [2], this method doesn't require a user to explicitly compare each pair of trajectories in the database and therefore results in a lighter effort for the expert. Yet, each trajectory is implicitly compared to all the others through the scores which results in additional information. Moreover, preference-based RL learns a policy, not a reward whereas learning a reward is the purpose of this paper. Indeed, we argue that IRL is a technique which has many advantages such as the transferability to other environments [17] or the ability for the agent to learn online by reinforcement after the reward has been computed, making the policy improve with time. In addition, in the domain of HMI, collecting expert ratings is common and well-researched [14, 8]

It is important to notice that global scores cannot be used directly as rewards (for instance on the final state). Indeed, similar scores can be associated to very different trajectories, finishing on the same state, but having different lengths for instance. Scores must thus be considered as a value associated to the global trajectory. In practice, the IRL problem is cast into a special case of least-square regression and thus results in a very efficient method (renamed SBIRL for Score-based IRL instead of Distance-Minimisation IRL), robust to noisy scoring and coming with theoretical guarantees. Especially, the error propagation analysis is an original contribution of this paper which is accompanied by a finite-sample analysis relying on very recent results on linear least-square estimation. The practical properties of the algorithm are studied on synthetic problems as well as on a real-world application: the optimisation of a spoken dialogue system strategy.

2. FORMAL DESCRIPTION

A Markov Decision Process (MDP) is a tuple $\{\mathcal{S}, \mathcal{A}, P, \gamma, r\}$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the finite action space, $P(ds'|s,a)$ the Markovian transition kernel on $\mathcal{S}$, $\gamma \in (0,1)$ the discount factor and $r : \mathcal{S} \to \mathbb{R}$ the (bounded) reward function. A policy is a mapping $\pi : \mathcal{S} \to \mathcal{A}$. For a policy π, we define the transition kernel $P_\pi(ds'|s) = P(ds'|s, \pi(s))$. The quality of a policy is quantified by the related value function, that associates to each state the expected discounted sum of rewards received from starting in state s and following the policy π:

$$v_\pi^r(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(S_t) | S_0 = s, S_{t+1} \sim P_\pi(.|S_t)\right].$$

An optimal policy π_*^r (with respect to the reward r) is such that the related value function $v_{\pi_*^r}^r = v_*^r$ satisfies componentwise $v_*^r \geq v_\pi^r$, for any policy π.

In this work, the true reward r is unknown and has to be estimated. To do so, we adopt a linear parameterisation of the reward and we assume the availability of a set of trajectories scored by a human expert. More formally, the reward is parameterised by a feature vector $\phi : \mathcal{S} \to \mathbb{R}^d$,

$$r_\theta(s) = \theta^\top \phi(s).$$

The available data is a set

$$\mathcal{D} = \{(h_i, v_i)_{1 \leq i \leq n}\},$$

where

$$h_i = (s_0^i, \dots, s_{T_i}^i) = (s_j^i)_{j=0}^{T_i}$$

is a trajectory of length $T_i + 1$ and v_i is the score (seen as the discounted sum of rewards of h_i) given by a human expert for this trajectory. Notice that we do not make any specific assumption about how the trajectory is generated (except that it is a trajectory obtained by applying successively T_i actions on encountered states, next states being drawn according to the dynamics). Trajectories in the dataset are assumed independently and identically distributed (which does not mean that states of a given trajectory are independent, obviously). For a given trajectory $h = (s_t)_{t=0}^T$ and a given reward r_θ, the discounted sum of rewards can be written as

$$\sum_{t=0}^{T} \gamma^t r_\theta(s_t) = \theta^\top \mu(h) \text{ with } \mu(h) = \sum_{t=0}^{T} \gamma^t \phi(s_t).$$

We interpret the scores given by the human expert as noisy estimates of the discounted sum of rewards of the associated trajectories. The expert is thus not assumed to give perfect scores but is error-prone. The problem thus consists of regressing the scores v_i on the mappings of the histories $\mu(h_i)$. Formally, we require that the underlying procedure asymptotically minimizes the risk based on the classic ℓ_2-loss:

$$\mathcal{R}(\theta) = \mathbb{E}[(V - \theta^\top \mu(H))^2]. \tag{1}$$

The joint distribution on V (scores) and $\mu(H)$ (mappings of histories) is imposed: it is the distribution used to sample the dataset $\mathcal{D}$. Given any procedure that asymptotically minimises risk (1), and denoting by θ_n the estimate computed from the dataset $\mathcal{D}$, we obtain an estimate r_{θ_n} of the reward. For example, the SBIRL algorithm estimates the reward by solving the following linear least-squares problem:

$$\begin{aligned}\theta_n &= \operatorname*{argmin}_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left(v_i - \theta^\top \mu(h_i)\right)^2 \\ &= \left(\frac{1}{n} \sum_{i=1}^{n} \mu(h_i)\mu(h_i)^\top\right)^{-1} \frac{1}{n} \sum_{i=1}^{n} \mu(h_i) v_i, \end{aligned} \tag{2}$$

assuming that the matrix $\frac{1}{n}\sum_{i=1}^n \mu(h_i)\mu(h_i)^\top$ is invertible (ordinary least-squares). If this is not the case, one can use for example ℓ_2-regularisation [22] or ℓ_1-regularisation [21], among others. Hence, given a set of trajectories scored by a human expert, we have a reward estimate r_{θ_n}. In the next section, we analyze the quality of this reward.

3. ANALYSIS

To study the estimate r_{θ_n}, we assume that the scores provided by the expert correspond to a discounted cumulative sum of rewards, up to some noise η, for a reward function that lies in the hypothesis space $\mathcal{H} = \{\theta^\top \phi(s), \theta \in \mathbb{R}^d\}$.

Assumption 1
There exists a vector parameter $\theta_* \in \mathbb{R}^d$ and a centered noise η such that for any trajectory $h = (s_t)_{t=0}^T$ and any associated score v, one has

$$v = \sum_{t=0}^{T} \gamma^t r_{\theta_*}(s_t) + \eta(h).$$

Under this assumption, the minimiser of risk (1) is

$$\theta_* = \operatorname*{argmin}_{\theta \in \mathbb{R}^d} \mathcal{R}(\theta) = A^{-1} b,$$
$$\text{with } A = \mathbb{E}[\mu(H)\mu(H)^\top] \text{ and } b = \mathbb{E}[\mu(H)V],$$

assuming that the matrix A is invertible. First, we assume that the estimator θ_n satisfies (with high probability) that

$$\mathcal{R}(\theta_n) \leq \mathcal{R}(\theta_*) + \epsilon,$$

for some error ϵ. Then, we instantiate the related results in the case of ordinary least-squares.

3.1 Propagation of errors

First, we want to control the risk $\mathbb{E}_\nu[(r_{\theta_*}(S) - r_{\theta_n}(S))^2]$ for some distribution ν over states.

Theorem 1 (Propagation of errors: rewards)
Write λ_m the minimum eigenvalue of $\mathbb{E}[\mu(H)\mu(H)^\top]$ and λ_M the maximum eigenvalue of $\mathbb{E}_\nu[\phi(S)\phi(S)^\top]$. Assume that θ_n satisfies

$$\mathcal{R}(\theta_n) \leq \mathcal{R}(\theta_*) + \epsilon,$$

then the associated reward satisfies

$$\mathbb{E}_\nu[(r_{\theta_*}(S) - r_{\theta_n}(S))^2] \leq \frac{\lambda_M}{\lambda_m}\epsilon.$$

PROOF. The proof is provided in the appendix. □

Theorem 1 shows that if ϵ is small, the estimated reward r_{θ_n} will be close to the true reward r_{θ_*}. However, we are ultimately interested in computing an optimal policy $\pi_*^{r_{\theta_n}}$ (optimal policy relatively to the reward r_{θ_n}) for the control problem at hand. The question we answer now is: how close to optimal is the policy $\pi_*^{r_{\theta_n}}$ respectively to the one of the true reward $\pi^*_{r_{\theta_*}}$? To do so, we bound the quantity

$$\mathbb{E}_{\nu'}[(v_*^{r_{\theta_*}}(S) - v^{r_{\theta_*}}_{\pi_*^{r_{\theta_n}}}(S))^2],$$

states being sampled according to an arbitrary distribution ν' of interest (possibly different from the distribution ν).

Before stating the result, we provide some notations. For a distribution ν and a function $f \in \mathbb{R}^S$, we write

$$\nu f = \mathbb{E}_\nu[f(S)].$$

For a stochastic kernel Q, we have

$$[Qf](s) = \mathbb{E}_{Q(.|s)}[f(S')].$$

Therefore, νQ is a distribution such that

$$\mathbb{E}_{\nu Q}[f] = \mathbb{E}_\nu[Qf] = \mathbb{E}_{S\sim\nu}[\mathbb{E}_{S'\sim Q(.|S)}][f(S')].$$

Theorem 2 (Propagation of errors: values)
For a policy π, define C_π as the smallest coefficient such that

$$(1-\gamma)\nu'(I - \gamma P_\pi)^{-1} \leq C_\pi \nu.$$

Assume that θ_n satisfies $\mathcal{R}(\theta_n) \leq \mathcal{R}(\theta_*) + \epsilon$, then the associated value satisfies

$$\mathbb{E}_{\nu'}[(v_*^{r_{\theta_*}}(S) - v^{r_{\theta_*}}_{\pi_*^{r_{\theta_n}}}(S))^2] \leq \frac{2\left(C_{\pi_*} + C_{\pi_*^{r_{\theta_n}}}\right)}{(1-\gamma)^2}\frac{\lambda_M}{\lambda_m}\epsilon.$$

PROOF. The proof is provided in the appendix. □

Theorem 2 shows that, if the error ϵ is small, the optimal policy with respect to the learned reward function will be close to the optimal policy with respect to the unknown reward (closeness being measured in terms of value functions). There is an additional multiplicative term compared to Th. 1. The concentrability coefficient C_π measures the dissimilarity between the distribution of data ν and the distribution $(1-\gamma)\nu'(I-\gamma P_\pi)^{-1}$, the γ-weighted occupancy measure induced by policy π when the initial state is sampled from ν' (the distribution of interest for controlling the value function) and the coefficient $(1-\gamma)^{-1}$ is the average optimisation horizon. Both terms are standard when bounding value functions.

3.2 A finite sample analysis

In this section, we provide a finite sample analysis in the case where $\theta_n \in \mathbb{R}^d$ is the ordinary linear least-squares estimate of Eq. (2), based on the bound of [11]. For this, we need some technical assumptions.

Assumption 2
The noise is subgaussian: there exists $\sigma \geq 0$ such that, almost surely, for all $\lambda \in \mathbb{R}$,

$$\mathbb{E}[e^{\lambda\eta(H)}|H] \leq e^{\frac{\lambda^2\sigma^2}{2}}.$$

Moreover, there exists a finite $\rho \geq 1$ such that, almost surely,

$$\frac{\|A^{-\frac{1}{2}}\mu(H)\|_2}{\sqrt{d}} \leq \rho.$$

The subgaussian assumption is easily satisfied if the scores provided by the expert and the basis functions are bounded (in this case, the noise is bounded, and a bounded random variable is subgaussian [10]). The other assumption corresponds to condition 1 of [11]. For example, if for any $s \in \mathcal{S}$ we have

$$\|\phi(s)\|_\infty \leq \phi_{\max},$$

then almost surely we have

$$\|\mu(H)\|_2 \leq \frac{\sqrt{d}}{1-\gamma}\phi_{\max}$$

and the assumption holds for

$$\rho \geq \frac{\phi_{\max}}{(1-\gamma)\sqrt{\lambda_m}}.$$

Corollary 1
Let δ be such that

$$\ln\frac{9}{\delta} > \max(0, 2.6 - \ln d).$$

If assumptions 1 and-2 hold and if

$$n \geq 6\rho^2 d \ln\frac{9d}{\delta},$$

then with probability at least $1-\delta$ we have

$$\mathbb{E}_\nu[(r_{\theta_*}(S) - r_{\theta_n}(S))^2] \leq$$
$$\frac{\lambda_M}{\lambda_m}\frac{\sigma^2(d + 2\sqrt{d\ln\frac{9}{\delta}} + 2\ln\frac{9}{\delta})}{n} + o\left(\frac{1}{n}\right).$$

Under the same assumptions, w.p. at least $1-\delta$*:*

$$\mathbb{E}_{\nu'}[(v_*^{r_{\theta_*}}(S) - v_{\pi_*^{r_{\theta_n}}}^{r_{\theta_*}}(S))^2] \leq$$

$$\frac{2\left(C_{\pi_*} + C_{\pi_*^{r_{\theta_n}}}\right)}{(1-\gamma)^2}\frac{\lambda_M}{\lambda_m}\frac{\sigma^2(d + 2\sqrt{d\ln\frac{9}{\delta}} + 2\ln\frac{9}{\delta})}{n} + o\left(\frac{1}{n}\right).$$

PROOF. The proof is provided in the appendix. □

This result mainly tells that in the case of ordinary least-squares, we have $\epsilon = O(\frac{d}{n})$, and thus the rates for the errors of the reward and of the value (of the optimal policy) are the same, up to the additional constants arising because of error propagation. Since d is the number of features and n the number of trajectories, this gives an idea of the amount of data required to attain a given accuracy (however, notice that the concentrability coefficients can be hardly estimated, a standard problem in reinforcement learning). A wiser (but much more difficult) analysis would take into account the number of transitions instead of the number of trajectories, yet this is beyond the scope of this paper.

4. EXPERIMENTS

Experiments are organised in two sets. The first set is executed on synthetic data, *i.e.* randomly-constructed finite MDPs called Garnets [3, 17, 18]. We lead three main canonical experiments aiming at supporting the theoretical results. First, we verify the improvement in performance of SBIRL when the number of trajectories grows. Second, we evaluate the sensitivity to the noise η. Third, we determine if sampling the trajectories with an expert or a random policy influences the performance of SBIRL and compare with standard IRL methods provided with the same data. In a second set of experiments, we investigate the suitability of SBIRL for real-world problems by applying it to the optimisation of spoken dialogue system management.

4.1 Experiments on Garnets

In this section, experiments are done on a set of artificially built MDPs so as to support the theoretical findings. Testing SBIRL requires three key elements: first, to build a finite MDP; second, to generate trajectories of random lengths in this MDP; and finally, to score these trajectories with a given noise η. Garnets [3] are an abstract class of finite MDPs, easy to build. Here, we consider a special case of Garnets specified by three parameters: $(N_{\mathcal{S}}, N_{\mathcal{A}}, N_B)$. Parameters $N_{\mathcal{S}}$ and $N_{\mathcal{A}}$ are respectively the number of states and of actions. Thus, $\mathcal{S} = (s_i)_{i=1}^{N_{\mathcal{S}}}$ and $\mathcal{A} = (a_i)_{i=1}^{N_{\mathcal{A}}}$ are, respectively, the state and action spaces. The parameter N_B ($N_B \leq N_{\mathcal{S}}$), called the branching factor, defines for each pair (s, a) the number of next states. N_B states are drawn uniformly and without replacement from $\mathcal{S}$ and form the set of next states of (s, a) noted $\mathcal{S}_{s,a} = (s'_i)_{i=1}^{N_B}$. Then, to define the Markovian kernel P, for each state-action (s,a), we draw randomly and uniformly in $[0, 1]$ $N_B - 1$ cutting points. Let us note $(p_i)_{i=1}^{N_B-1}$ this set of cutting points sorted in increasing order, $p_0 = 0$ and $p_{N_B} = 1$. To completely define the dynamics, one assigns $P(s'_i|s, a)$ according to the following rule: $\forall i \in \{1, \dots, N_B\}$, $P(s'_i|s, a) = p_i - p_{i-1}$. Finally, for each state $s \in \mathcal{S}$, the reward $r(s)$ is drawn randomly and uniformly in $[0, 1]$, and $\gamma = 0.9$. This choice imposes that the non-perturbed score of a trajectory $h = (s_t)_{t=0}^T$ is bounded as follows

$$0 \leq \sum_{t=0}^{T}\gamma^t r(s_t) \leq \frac{1}{1-\gamma} = 10.$$

As we choose finite MDPs, a canonical choice of features ϕ is the tabular basis $\phi : \mathcal{S} \to \mathbb{R}^{N_{\mathcal{S}}}$ where $\phi(s) \in \mathbb{R}^{N_{\mathcal{S}}}$ is a vector which is null excepted in s where it is equal to 1.

The strategy chosen to generate a trajectory h of random length in a Garnet following a policy π consists of first choosing randomly and uniformly a starting state $s_0 \in \mathcal{S}$. Then, starting from s_0 we apply, with probability $(1-p)$ ($p \in (0, 1)$), the policy π to the current state s_i in order to go to the next state s_{i+1} and with probability p we stop the trajectory. Doing so, we obtain a trajectory $h = (s_t)_{t=0}^T$ where the length of the trajectory $T + 1$ is a geometrically distributed random variable: for all $k \in \mathbb{N}^*$,

$$Pr(T = k) = (1-p)^{k-1}p, \mathbb{E}[T] = \frac{1}{p} \text{ and } Var[T] = \frac{1-p}{p^2}.$$

We choose this strategy because it is easy to implement and allows us to control the mean of the length of the trajectories.

Finally, to give a score to a trajectory $h = (s_t)_{t=0}^T$, we proceed as follows. Let η be a Gaussian distributed and real valued random variable with mean $\nu = 0$ and variance σ^2 ($\eta \sim \mathcal{N}(0, \sigma^2)$), the score of the trajectory h is

$$v(h) = \lceil\sum_{t=0}^{T}\gamma^t r(s_t) + \eta\rfloor = \sum_{t=0}^{T}\gamma^t r(s_t) + \eta + \eta_q,$$

where $\lceil x\rfloor$ is the nearest integer from x and η_q is a quantification noise which is supposed to be a uniformly distributed random variable of mean $\mu = 0$ and variance $\sigma^2 = \frac{1}{12}$. We deliberately add a quantification noise in order to model the fact that experts are often asked to use an integer scale. More precisely, we have seen that the choice of the reward r and the discount factor γ imposes, for a trajectory $h = (s_t)_{t=0}^T$, that the non-perturbed score is bounded: $0 \leq \sum_{t=0}^{T}\gamma^t r(s_t) \leq \frac{1}{1-\gamma} = 10$. Thus, we have constructed the score $v(h)$ such that it models the choice of a score in the set of integers $\{0, 1, \dots, 10\}$.

4.1.1 Performance evaluation of SBIRL

So as to evaluate the SBIRL performance, we want first to measure the mean performance of this algorithm over several Garnets when the number of trajectories, N_T, grows. To do so, we create $(G_q)_{q=1}^{N_G}$ Garnets of size ($N_{\mathcal{S}} = 100, N_{\mathcal{A}} = 5, N_B = 10$) where we compute an optimal policy π_q^e w.r.t the real reward r_q of G_q via the policy iteration algorithm. For each G_q and each iteration $j \in \{1, \dots, N_{It}\}$ ($N_{It} = 10$), we generate N_T trajectories $(h_i)_{i=1}^{N_T}$ according to a random policy π_r (at each state s, the probability to choose action a is $\frac{1}{N_{\mathcal{A}}}$) with $p = 0.01$ such that the mean length is around 100. We compute, for each trajectory the score $v(h_i)$ with $\eta \sim \mathcal{N}(0, \sigma^2 = 1)$. Here, the standard deviation of the noise represents 10% of the maximum non perturbed-score achievable by a trajectory. Thus, for each G_q, each iteration j and a given number of trajectories N_T, we obtain a set $(h_i, v_i = v(h_i))_{i=1}^{N_T}$ that we use as an input for SBIRL which outputs the reward $r_{q,j}$. If $\pi_*^{r_{q,j}}$ is the optimal policy w.r.t $r_{q,j}$, then the performance of SBIRL is measured via the normalised error between the value functions (w.r.t r_q)

of π^e_q and $\pi^{r_{q,j}}_*$:

$$T_{q,j}(N_T) = \|v^{r_q}_{\pi^e_q} - v^{r_q}_{\pi^{r_{q,j}}_*}\|_2 \|v^{r_q}_{\pi^e_q}\|_2^{-1},$$

where $\|.\|_2$ is the Euclidean norm. The lower the error, the better the performance. Finally, the mean error $T(N_T)$ is the mean of the

$$(T_{q,j}(N_T))^{1 \leq j \leq 10}_{1 \leq q \leq 50}.$$

In Fig. 1, we plot the SBIRL error, $T(N_T)$, where N_T varies from 100 to 1000 and the error corresponding to the random policy. We observe, as predicted by the analysis, that the error converges to zero as N_T grows.

Another interesting aspect to evaluate is the tolerance of our method to the noise $\eta \sim \mathcal{N}(0, \sigma)$ over a number of Garnets. In order to evaluate this aspect, we realise exactly the same experiment as before except that N_T is fixed to 500 and σ (standard deviation of the noise) is now the varying parameter. For each G_q, each iteration j and for a given σ, SBIRL outputs the reward $r_{q,j}$ and the error $T_{q,j}(\sigma)$ of SBIRL is defined as before. The mean error $T(\sigma)$ is the mean of the

$$(T_{q,j}(\sigma))^{1 \leq j \leq 10}_{1 \leq q \leq 50}.$$

In Fig. 2, we plot the SBIRL error, $T(\sigma)$, where σ varies from 0 to 5 which represents 50% of the maximum non-perturbed score given to a trajectory. We observe that the performance is good when the noise is low and deteriorates as the noise gets bigger which is also observed in the bound of Corollary 1. We also remark that the standard deviation of the error, which appears as a shade in Fig. 2, gets bigger as the noise increases.

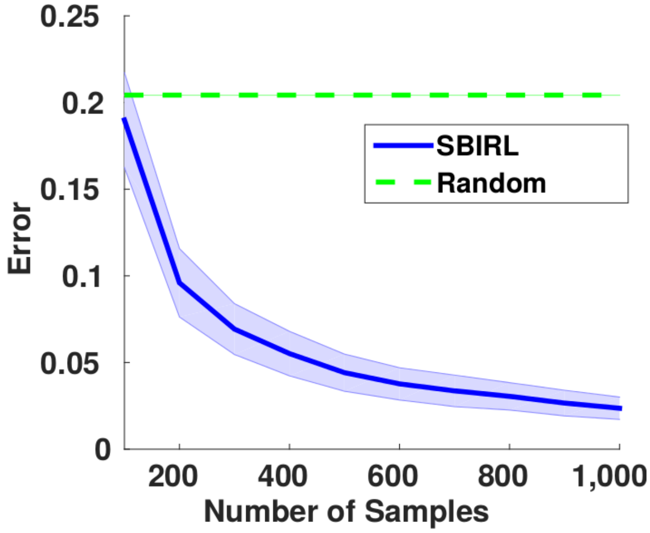


Figure 1: SBIRL error as N_T grows

4.1.2 Comparison to IRL

This experiment is designed to provide insights about how unreliable (w.r.t an optimal policy) the data should be to require annotation, preventing the use of standard IRL. To compare IRL to SBIRL, we need to generate near-optimal trajectories to simulate a wide range of expertise levels. To do so, we consider a noisy expert where the noise is controlled by a parameter β such that the generated policy could range from optimal to random policies. More precisely, the trajectories $(h_i)^{N_T}_{i=1}$ are sampled according to a noisy-expert policy $\pi^n(\beta)$ where $\beta \in [0, 1]$ and given as an input to SBIRL and two IRL algorithms which are Projection [1] and SCIRL [12]. To do so, $\lceil \beta N_{\mathcal{S}} \rfloor$ states are drawn randomly without replacement from $\mathcal{S}$. On those states, $\pi^n(\beta)$ is random and on the others, $\pi^n(\beta)$ is optimal. We perform the same experiment as before except that N_T is fixed to 300 and 500, $N_{It} = 3$ and the sampling policy, for each G_q, is $\pi^n_q(\beta)$. For each G_q, each iteration j and for a given β, each algorithm A outputs the reward $r^A_{q,j}$ with an error $T^A_{q,j}(\beta)$. The mean error $T^A(\beta)$ is the mean of the

$$(T^A_{q,j}(\beta))^{1 \leq j \leq 3}_{1 \leq q \leq 50}.$$

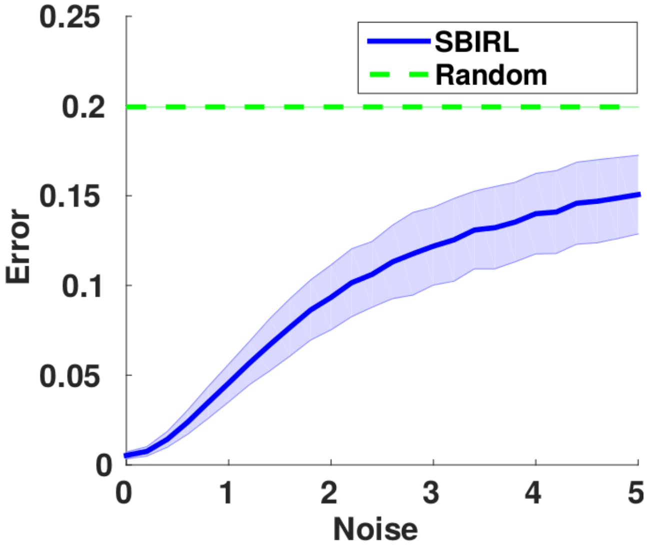


Figure 2: SBIRL error as σ grows

In Fig. 3(a) and Fig. 3(b), errors are plot w.r.t β (proportion of random *vs* optimal policy) varying from 0 to 1 for $N_T = 300$ and $N_T = 500$ respectively. When $N_T = 300$, the collected trajectories must be optimal on approximately 60% ($\beta = 0.4$) of states for Projection and 40% ($\beta = 0.6$) for SCIRL to achieve the same result as SBIRL. And, when $N_T = 500$, the collected trajectories must be optimal on approximately 80% ($\beta = 0.2$) of states for Projection and 55% ($\beta = 0.45$) for SCIRL to achieve the same result as SBIRL. This shows that when N_T grows, the data should be of high quality to reach the level of performance of SBIRL. So, according to the amount of available data, their reliability has to be assessed so as to decide whether directly using IRL or spending time for annotations and use SBIRL.

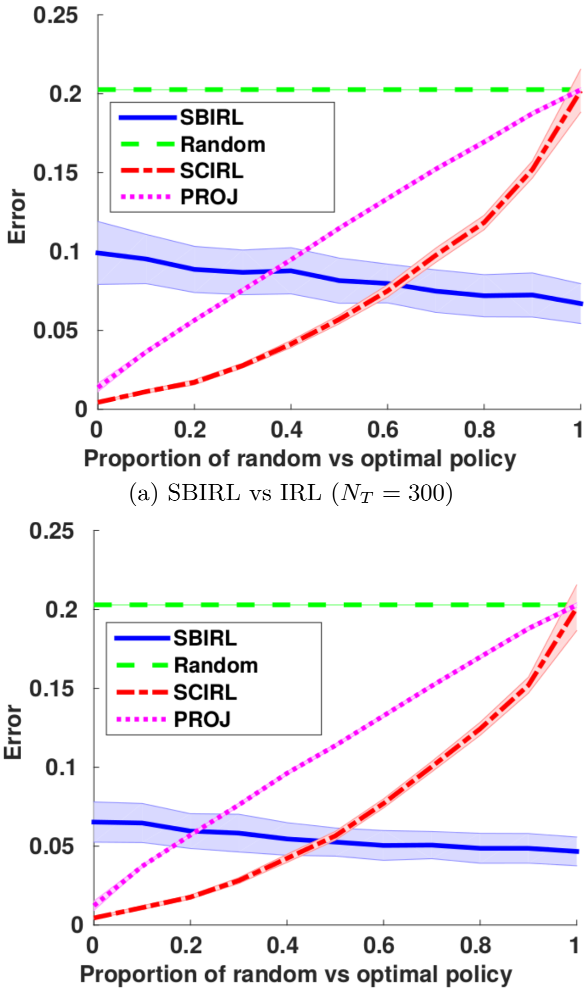
4.2 Application to Spoken Dialogue Systems

Here, SBIRL is applied to the optimization of a spoken dialogue system (SDS) implementing an appointment scheduling task. This experiment is mainly proposed to show the suitability of SBIRL for use in a real-world problem. Of course we do not have information about the quality of the scores or the noise and we do not study the sensitivity to these variables here.

An SDS is an HMI which converses in spoken natural language with users. Speech is transcribed to the machine by Automatic Speech Recognition (ASR). A dialogue is mod-

Policy	Score	#turns	#ASR rejections	Task Completion
Handcrafted	8.266 ± 0.008	6.547 ± 0.036	0.73 ± 0.008	0.878 ± 0.006
SBIRL 500	8.373 ± 0.006	5.568 ± 0.035	0.541 ± 0.006	0.796 ± 0.001
SBIRL 1000	8.424 ± 0.006	4.938 ± 0.027	0.469 ± 0.005	0.791 ± 0.001
SBIRL 1500	8.446 ± 0.005	4.776 ± 0.025	0.44 ± 0.005	0.79 ± 0.001

Table 1: Performance of the handcrafted and SBIRL policies in the appointment scheduling simulations.



(a) SBIRL vs IRL ($N_T = 300$)

(b) SBIRL vs IRL ($N_T = 500$)

Figure 3: SBIRL vs IRL

eled as a sequence of turns, a turn starting at the beginning of each user utterance. The dialogue manager is the component of the SDS that chooses what the machine should say after each user utterance. To do so, it keeps track of the dialogue context. This process is modeled as an MDP where a state s encodes the current dialogue context and an action a represents a System Dialogue Act (SDA). An SDA describes the meaning of an utterance at the level of illocutionary force [4].

Unlike [7] which simulated dialogues, this experiment relies on interactions collected using an actual SDS. Scores are computed thanks to annotations given by real users. The collected corpus, DINASTI [8], contains 1734 dialogues. For each dialogue, the user had a two-week calendar and was asked to set an appointment at one of the available slots on the calendar. There was always only one slot available for both the system and the user so each dialogue needed several turns of negotiation to find the convenient appointment. After each dialogue, users were asked to fill in a questionnaire and rate the dialogue on a scale of 1 to 10. We trained an ordinal logistic regression model on these ratings and computed the score of new dialogues based on the numbers of dialogue turns and ASR rejections, the average ASR confidence score and task completion.

The system `Sys` and the user `Usr` have 30 time slots in common. Each slot is composed of a day of the week, the day number and an hour. `Usr` and `Sys` have different preferences concerning these slots. This is modeled by uniformly drawing in $[0, 1]$ two values, $v^{\mathrm{Usr}}(\mathrm{sl}_i)$ for `Usr` and $v^{\mathrm{Sys}}(\mathrm{sl}_i)$ for `Sys`, for each slot sl_i. These values are drawn at the beginning of each dialogue and `Usr` is set to only accept the 5 slots with the highest values. If `Sys` proposes one of these slots, `Usr` accepts it, otherwise s/he suggests one. `Sys` might ask `Usr` to confirm the last utterance it understood. For an implicit confirmation, `Sys` repeats the utterance and then moves on with the dialogue. If the utterance is correct, `Usr` does not say anything. Otherwise, s/he barges in before `Sys` has moved on and repeats the correct utterance. For an explicit confirmation, `Sys` asks `Usr` whether the understood utterance is correct or not. `Usr` then either confirms or repeats the correct utterance. `Usr` also repeats the last utterance if `Sys` informs her/him that the utterance was not understood (ASR rejection) or if one of the three elements of the slot was not understood. Each dialogue begins by a proposition by `Sys`. Then, during the dialogue, when there is no ASR problem, `Sys` can choose between 4 actions: accept a slot, refuse and propose another slot, ask for an implicit confirmation and ask for an explicit confirmation. A dialogue ends when one participant has accepted the other's proposition. `Sys` proposes slots in decreasing order of $v^{\mathrm{Sys}}(sl_i)$. ASR confidence scores were simulated with a word error rate of 10%. The score was computed by drawing from a normal distribution $\mathcal{N}(-1, 1)$ in the case of a misrecognition ($\mathcal{N}(1, 1)$ otherwise) and by then applying the sigmoid function to this value. The threshold for ASR rejection was set to 0.5.

For evaluation purposes, SBIRL was applied to simulated dialogues with a uniformly random policy. Given sl_i the selected slot, task completion was defined as $v^{\mathrm{Usr}}(sl_i) + 0.5 \times v^{\mathrm{Sys}}(\mathrm{sl}_i)$. The state space was composed of the latest User Dialogue Act (UDA, for instance ACCEPT) as well as the numbers of dialogue turns and ASR rejections, the average ASR confidence score and the value $v^{\mathrm{Sys}}(\mathrm{sl}_i)$ of the slot sl_i being currently discussed. The state space was built by applying entropy-based discretisation [9] with these parameters on the random corpus and then by aggregating the resulting intervals. We performed 100 runs on 500, 1000 and 1500 dialogues. For each run, we computed the state space, ran SBIRL and then learnt a policy with the new reward function. On average, the state space had respectively 48.5, 71.8 and 79.5 states. The learning algorithm was SARSA with ϵ-greedy exploration, where ϵ was set to 0.01. The learning factor α was set to 0.1. We learnt a policy on 5000 dialogues and tested the learnt policy on 2000 dialogues. We compare learning with a handcrafted policy tested on 100000 dialogues which always asks the user for an implicit confirmation and then accepts a slot sl_i if $v^{\text{Sys}}(\text{sl}_i) \geq 0.5$ or if the number of dialogue turns is superior or equal to 15. Table 1 contains 95% confidence intervals for the estimated user satisfaction score and dialogue statistics with the handcrafted policy and with the policies learnt with SBIRL. The policy learnt with SBIRL improves the performance score by finding a better compromise between the number of dialogue turns and task completion. The policy learnt with SBIRL produces dialogues with, on average, 2 fewer dialogue turns than the ones resulting of the handcrafted policy and with sacrificing less than 10% of the task completion value. This is an important result since the average dialogue duration is well-known to highly influence user's satisfaction [23].

5. CONCLUSION

This paper studied the SBIRL algorithm which applies to a non-standard setting for Inverse Reinforcement Learning (IRL) relying on annotated trajectories. Trajectories can be collected using policies that are not assumed to be generated by an expert but are globally scored *a posteriori* by such an expert. This setting is much more suitable for many applications where datasets are already available but with unreliable quality. Expert's annotations are supposed to be noisy and robustness to noise was assessed both theoretically and practically. The theoretical analysis shows that the algorithm produces a policy that approximates the optimal one at a rate that increases linearly with the number of scored trajectories. Results of synthetic experiments were in accordance with theory and also showed that the noise in the scores is not too influential. These experiments also provided insights about how unreliable the data should be to require annotation, preventing the use of standard IRL methods. An application to spoken dialogue systems, using real data collected with an actual system, showed that this method is applicable in practice to learn more efficient interaction policies in a realistic domain (appointment scheduling). In future work, this methodology will be applied to large scale systems such as human-robot interaction.

APPENDIX

This appendix recalls the stated theoretical results and proves them. First, we recall the required assumptions.

Assumption 1
There exists a vector parameter $\theta_* \in \mathbb{R}^d$ and a centered noise η such that for any trajectory $h = (s_t)_{t=0}^T$ and any associated score v, one has

$$v = \sum_{t=0}^{T} \gamma^t r_{\theta_*}(s_t) + \eta(h).$$

Assumption 2
The noise is subgaussian: there exists $\sigma \geq 0$ such that, almost surely, for all $\lambda \in \mathbb{R}$,

$$\mathbb{E}[e^{\lambda\eta(H)}|H] \leq e^{\frac{\lambda^2\sigma^2}{2}}.$$

Moreover, there exists a finite $\rho \geq 1$ such that, almost surely,

$$\frac{\|A^{-\frac{1}{2}}\mu(H)\|_2}{\sqrt{d}} \leq \rho.$$

We also recall that for a distribution ν and a function $f \in \mathbb{R}^S$, we write

$$\nu f = \mathbb{E}_\nu[f(S)].$$

For a stochastic kernel Q, we have

$$[Qf](s) = \mathbb{E}_{Q(.|s)}[f(S')].$$

Therefore, νQ is a distribution such that

$$\mathbb{E}_{\nu Q}[f] = \mathbb{E}_\nu[Qf] = \mathbb{E}_{S\sim\nu}[\mathbb{E}_{S'\sim Q(.|S)}][f(S')].$$

A. PROPAGATION OF ERRORS: REWARDS

Theorem 1 (Propagation of errors: rewards)
Write λ_m the minimum eigenvalue of $\mathbb{E}[\mu(H)\mu(H)^\top]$ and λ_M the maximum eigenvalue of $\mathbb{E}_\nu[\phi(S)\phi(S)^\top]$. Assume that θ_n satisfies $\mathcal{R}(\theta_n) \leq \mathcal{R}(\theta_*) + \epsilon$, then the associated reward satisfies

$$\mathbb{E}_\nu[(r_{\theta_*}(S) - r_{\theta_n}(S))^2] \leq \frac{\lambda_M}{\lambda_m}\epsilon.$$

Proof. First, we use the fact that the risk is a quadratic form (and that $A\theta_* = b$):

$$\begin{aligned}
\mathcal{R}(\theta) &= \mathbb{E}[(v - \theta^\top\mu(H))^2] = \theta^\top A\theta - 2\theta^\top b + \mathbb{E}[V^2] \\
&= \|A\theta - b\|^2_{A^{-1}} + \mathbb{E}[V^2] - b^\top A^{-1} b \\
&= \|A\theta - b\|^2_{A^{-1}} + \mathbb{E}[V^2] - 2b^\top\theta_* + \theta_*^\top A\theta_* \\
&= \|A\theta - b\|^2_{A^{-1}} + \mathcal{R}(\theta_*) \\
&= \|\theta - \theta_*\|^2_A + \mathcal{R}(\theta_*).
\end{aligned}$$

Similarly, writing $C = \mathbb{E}_\nu[\phi(S)\phi(S)^\top]$, one can easily show that

$$\mathbb{E}_\nu[(r_{\theta_*}(S) - r_{\theta_n}(S))^2] = \|\theta - \theta_*\|^2_C.$$

Now, let $x \in \mathbb{R}^d$. For a symmetric and definite positive matrix M, write $\|x\|_M = \sqrt{x^\top Mx}$ and S_M its Cholesky decomposition. Write also $\|M\|_2$ the spectral norm of M. We have

$$\begin{aligned}
\|x\|_C &= \sqrt{x^\top S_C^\top S_C x} \\
&= \|S_C x\|_2 \\
&= \|S_C S_A^{-1} S_A x\|_2 \\
&\leq \|S_C\|_2 \|S_A^{-1}\|_2 \|S_A x\|_2 \\
&= \frac{\lambda_M}{\lambda_m}\|x\|_A.
\end{aligned}$$

Thus, we have

$$\begin{aligned}
\mathbb{E}_\nu[(r_{\theta_*}(S) - r_{\theta_n}(S))^2] &= \|\theta - \theta_*\|^2_C \\
&\leq \frac{\lambda_M}{\lambda_m}\|\theta - \theta_*\|^2_A \\
&= \frac{\lambda_M}{\lambda_m}(\mathcal{R}(\theta_n) - \mathcal{R}(\theta_*)).
\end{aligned}$$

Using the fact that $\mathcal{R}(\theta_n) - \mathcal{R}(\theta_*) \leq \epsilon$ proves the result. □

B. PROPAGATION OF ERRORS: VALUES

Theorem 2 (Propagation of errors: values) *For a policy π, define C_π as the smallest coefficient such that $(1-\gamma)\nu'(I-\gamma P_\pi)^{-1} \le C_\pi \nu$. Assume that θ_n satisfies $\mathcal{R}(\theta_n) \le \mathcal{R}(\theta_*) + \epsilon$, then the associated value satisfies*

$$\mathbb{E}_{\nu'}[(v_*^{r_{\theta_*}}(S) - v^{r_{\theta_*}}_{\pi_*^{r_{\theta_n}}}(S))^2] \le \frac{2\left(C_{\pi_*} + C_{\pi_*^{r_{\theta_n}}}\right)}{(1-\gamma)^2}\frac{\lambda_M}{\lambda_m}\epsilon.$$

Proof. To simplify the notations, write $r = r_{\theta_*}$ the unknown reward function (of associated optimal policy $\pi_* = \pi_*^{r_{\theta_n}}$) and $\hat{r} = r_{\theta_n}$ the estimated reward function (of associated optimal policy $\hat{\pi} = \pi_*^{r_{\theta_n}}$). With these notations, the quantity to be bounded is $\mathbb{E}_{\nu'}[(v^r_{\pi_*}(S) - v^r_{\hat{\pi}}(S))^2]$.

Componentwise, we have

$$v^r_{\pi_*} - v^r_{\hat{\pi}} = v^r_{\pi_*} - v^{\hat{r}}_{\pi_*} + v^{\hat{r}}_{\pi_*} - v^{\hat{r}}_{\hat{\pi}} + v^{\hat{r}}_{\hat{\pi}} - v^r_{\hat{\pi}}.$$

As $\hat{\pi}$ is optimal for $\hat{r}$, we have $v^{\hat{r}}_{\pi_*} - v^{\hat{r}}_{\hat{\pi}} \le 0$, and a value function satisfies

$$v^r_\pi = \sum_{t\ge 0} \gamma^t P^t_\pi r = (I - \gamma P_\pi)^{-1} r,$$

therefore:

$$\begin{aligned}
&v^r_{\pi_*} - v^r_{\hat{\pi}} \\
&\le v^r_{\pi_*} - v^{\hat{r}}_{\pi_*} + v^{\hat{r}}_{\hat{\pi}} - v^r_{\hat{\pi}} \\
&= (I - \gamma P_{\pi_*})^{-1}(r - \hat{r}) + (I - \gamma P_{\hat{\pi}})^{-1}(\hat{r} - r) \\
&\le \frac{2}{1-\gamma} Q|r - \hat{r}| \\
\text{with } Q &= \frac{1-\gamma}{2}((I - \gamma P_{\pi_*})^{-1} + (I - \gamma P_{\hat{\pi}})^{-1}).
\end{aligned}$$

Notice that Q is a stochastic kernel. We have

$$\begin{aligned}
&\mathbb{E}_{\nu'}[(v^r_{\pi_*}(S) - v^r_{\hat{\pi}}(S))^2] \\
&\le \frac{4}{(1-\gamma)^2}\mathbb{E}_{S\sim\nu'}[\left(\mathbb{E}_{S'\sim Q(.|S)}[|r(S') - \hat{r}(S')|]\right)^2] \\
&\le \frac{4}{(1-\gamma)^2}\mathbb{E}_{S\sim\nu'}[\mathbb{E}_{S'\sim Q(.|S)}[(r(S') - \hat{r}(S'))^2]] \\
&= \frac{4\mathbb{E}_{S\sim\nu' Q}[(r(S) - \hat{r}(S))^2]}{(1-\gamma)^2} \\
&\le \frac{4(C_{\pi_*} + C_{\hat{\pi}})}{2(1-\gamma)^2}\mathbb{E}_\nu[(r(S) - \hat{r}(S))^2] \\
&= \frac{2(C_{\pi_*} + C_{\hat{\pi}})}{(1-\gamma)^2}\mathbb{E}_\nu[(r(S) - \hat{r}(S))^2],
\end{aligned}$$

the second inequality being due to the Jensen's inequality and the third one to the definition of C_π. Using Th. 1 allows concluding. □

C. FINITE SAMPLE ANALYSIS

Corollary 1 *Let δ be such that $\ln\frac{9}{\delta} > \max(0, 2.6 - \ln d)$. If assumptions 1 and-2 hold and if $n \ge 6\rho^2 d \ln\frac{9d}{\delta}$, then with probability at least $1-\delta$ we have*

$$\mathbb{E}_\nu[(r_{\theta_*}(S) - r_{\theta_n}(S))^2] \le \frac{\lambda_M}{\lambda_m}\frac{\sigma^2(d + 2\sqrt{d\ln\frac{9}{\delta}} + 2\ln\frac{9}{\delta})}{n} + o\left(\frac{1}{n}\right).$$

Under the same assumptions, with probability at least $1-\delta$:

$$\mathbb{E}_{\nu'}[(v_*^{r_{\theta_*}}(S) - v^{r_{\theta_*}}_{\pi_*^{r_{\theta_n}}}(S))^2] \le \frac{2\left(C_{\pi_*} + C_{\pi_*^{r_{\theta_n}}}\right)}{(1-\gamma)^2}\frac{\lambda_M}{\lambda_m}\frac{\sigma^2(d + 2\sqrt{d\ln\frac{9}{\delta}} + 2\ln\frac{9}{\delta})}{n} + o\left(\frac{1}{n}\right).$$

Proof. The assumptions made allow using Theorem 1 of [11], which states that with probability at least $1-\delta$ we have

$$\mathcal{R}(\theta_n) \le \mathcal{R}(\theta_*) + \frac{\sigma^2(d + 2\sqrt{d\ln\frac{9}{\delta}} + 2\ln\frac{9}{\delta})}{n} + o\left(\frac{1}{n}\right).$$

Plugging this result in Th. 1 and 2 allows concluding. □

REFERENCES

[1] P. Abbeel and A. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proc. of ICML*, 2004.

[2] R. Akrour, M. Schoenauer, and M. Sebag. Preference-based policy learning. In *Proc. of ECML*, 2011.

[3] T. Archibald, K. McKinnon, and L. Thomas. On the generation of Markov decision processes. *Journal of the Operational Research Society*, 1995.

[4] J. L. Austin. *How to do Things with Words.* Clarendon Press, Oxford, 1962.

[5] A. Boularias, H. R. Chinaei, and B. Chaib-draa. Learning the reward model of dialogue pomdps from data. In *NIPS Workshop on Machine Learning for Assistive Techniques*, 2010.

[6] A. Boularias, J. Kober, and J. Peters. Relative entropy inverse reinforcement learning. In *Proc. of AISTATS*, 2011.

[7] L. El Asri, R. Laroche, and O. Pietquin. Reward shaping for statistical optimisation of dialogue management. In *Proc. of SLSP*, 2013.

[8] L. El Asri, R. Laroche, and O. Pietquin. DINASTI: Dialogues with a Negotiating Appointment Setting Interface. In *Proc. of LREC*, 2014.

[9] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. of UAI*, 1993.

[10] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[11] D. Hsu, S. M. Kakade, and T. Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 14(3):569–600, 2014.

[12] E. Klein, M. Geist, B. Piot, and O. Pietquin. Inverse reinforcement learning through structured classification. In *Proc. of NIPS*, December 2012.

[13] O. Lemon and O. Pietquin, editors. *Data-Driven Methods for Adaptive Spoken Dialogue Systems: Computational Learning for Conversational Interfaces.* Springer, November 2012.

[14] S. Möller, K.-P. Engelbrecht, and R. Schleicher. Predicting the quality and usability of spoken dialogue services. *Speech Communication*, 50(8):730–744, 2008.

[15] T. Paek. Reinforcement learning for spoken dialogue systems: Comparing strengths and weaknesses for

practical deployment. In *Proc. of Dialog-on-Dialog Workshop at Interspeech*, 2006.

[16] O. Pietquin. Inverse reinforcement learning for interactive systems. In *Proc. of IJCAI workshop on Machine Learning for Interactive Systems*, 2013.

[17] B. Piot, M. Geist, and O. Pietquin. Learning from demonstrations: is it worth estimating a reward function? In *Proc. of ECML*, 2013.

[18] B. Piot, M. Geist, and O. Pietquin. Boosted and reward-regularized classification for apprenticeship learning. In *Proc. of AAMAS*, 2014.

[19] S. Russell. Learning agents for uncertain environments. In *Proc. of COLT*, 1998.

[20] S. P. Singh, M. J. Kearns, D. J. Litman, and M. A. Walker. Reinforcement learning for spoken dialogue systems. In *Proc. of Nips*, 1999.

[21] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[22] A. Tikhonov. Solution of incorrectly formulated problems and the regularization method. In *Soviet Mathematics*, volume 5, pages 1035–1038, 1963.

[23] M. A. Walker, D. J. Litman, C. A. Kamm, and A. Abella. PARADISE: A framework for evaluating spoken dialogue agents. In *Proc. of EACL*, 1997.

[24] S. Young, M. Gasic, B. Thomson, and J. D. Williams. POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.