Preference-based Reinforcement Learning

Riad Akrour **Marc Schoenauer** **Michèle Sebag**

TAO
CNRS − INRIA − Université Paris-Sud
FirstName.Name@inria.fr

Abstract

This paper investigates the problem of policy search based on the only expert's preferences. Whereas reinforcement learning classically relies on a reward function, or exploits the expert's demonstrations, preference-based policy learning (PPL) iteratively builds and optimizes a policy return estimate as follows: The learning agent demonstrates a few policies, is informed of the expert's preferences about the demonstrated policies, constructs a utility function compatible with all expert preferences, uses it in a self-training phase, and demonstrates in the next iteration the policy maximizing the current utility function.

PPL actually tackles an expensive optimization problem, as each policy assessment relies on the expert's feedback. The expert's preference model is used as a surrogate model of the true objective function. Compared to earlier work devoted to active preference learning (see e.g. Brochu et al., 2008), a main lesson is that the surrogate model must preferably be learned on the state $\times$ action space, as opposed to, on the policy parametric space. Empirical evidence from the cancer treatment policy domain is provided to support and discuss this claim.

1 Introduction

Since the early 2000s, significant advances in reinforcement learning have been obtained through using direct expert's input (inverse reinforcement learning [18], learning by imitation [8], learning by demonstration [16]), assuming the expert's ability to demonstrate quasi-optimal behaviors, and to provide an informed representation.

In 2011, two approaches based on preference learning have been proposed to learn directly a ranking-based policy [9] or a policy return estimate [2]. In the latter case, referred to as preference-based policy learning (PPL), the agent demonstrates a few policies, receives the expert's preferences about the demonstrated policies, constructs a utility function compatible with all expert preferences, uses it in a self-training phase, and demonstrates in the next iteration the policy maximizing the current utility function. The main merit of the PPL approach is twofold. Firstly, it sidesteps the design of the reward function at the state-action level [20]; as noted by [9], this design is critical when qualitative outcomes are considered, e.g. in the cancer treatment domain. Secondly, as opposed to inverse reinforcement learning [1, 15] PPL does not require the expert to demonstrate a quasi-optimal behavior; it does not even assume that the expert knows how to solve the task (see also [21]); the expert is only required to know whether some behavior is more able to reach the goal than some other one.

PPL relies on preference learning to build the policy return estimate, an intermediate utility function used to keep the expert's burden within reasonable limits. This utility function can be thought of as a surrogate model, supporting expensive function optimization [6]. As shown by e.g. [7], active preference learning can indeed be used for interactive optimization.

The first contribution of the paper concerns the space used to learn this preference-based surrogate model. The default option is to use the input space a.k.a. direct representation, here the policy parametric space. Another option, exploiting the RL specificities and referred to as feature space or indirect representation, has also been considered. Within the latter option, the surrogate model is a weighted sum of the overall time spent in a state-action pair (i.e. the average time the policy executes a given action in a given state). The rationale for this is the following. On the one hand, this indirect representation complies with the standard RL setting under a finite time horizon, where the policy return is defined as the cumulative reward expectation in a Markov Decision Process. On the other hand, this representation is not restricted to the MDP setting, as will be shown on the cancer treatment problem [9]. Lastly, it will be shown experimentally that the indirect representation is significantly more effective than the direct one, again on the example of the cancer treatment problem.

A second contribution regards the trade-off between exploration and exploitation, visiting new state-action pairs and exploiting the current utility function. Related approaches concerned with active optimization [14, 7, 17] proceed by generating points in the input space which maximize the expected global improvement. These approaches however do not apply when considering an indirect representation; an adaptive trade-off between the current utility function and an exploration term linked to the empirical success rate is used.

The paper is organized as follows. Section 2 gives an overview of PPL. Section 3 reports on the empirical validation of the approach on a cancer treatment problem. Section 4 discusses Preference-based Policy Learning strengths and weaknesses and presents perspectives for further research.

2 Preference-based policy learning: Overview

Let $\mathcal{S}$ and $\mathcal{A}$ respectively denote the state and the action space. A policy π is a mapping from $\mathcal{S}$ onto $\mathcal{A}$.

For the sake of simplicity only the finite time horizon H will be considered, where H is the number of time steps during which each candidate policy is demonstrated, although the extension to the infinite discounted case is straightforward. A parameterized policy representation, characterizing policy π from its parameter $\theta \in \Theta \subseteq \mathbb{R}^D$ and referred to as *direct representation* will be considered. An indirect representation will also be considered, mapping policy π onto some feature space, where $\mu(\pi)$ describes the state-action frequency under π; in the discrete case,

$$\mu(\pi) \in (\mathcal{S} \times \mathcal{A} \mapsto [0,1]) \qquad \mu(\pi)(s,a) = \mathbb{E}_{a_t \sim \pi(s_t), s_{t+1} \sim P_{s_t,a_t}} \left[\frac{1}{H} \sum_{t=0}^{H-1} 1_{s,a}(s_t, a_t) \right]$$

where the expectation is taken under the probability distribution P_{s_t,a_t} over $\mathcal{S}$ when taking action a_t in state s_t after $\pi(s_t)$. The utility function J_t learned at iteration t is sought as a linear function in the feature space (see below).

After initialization of the policy and constraint archives, respectively denoted Π and $\mathcal{C}$, PPL proceeds by iterating a 3-step process[1] ; at step t,

1. A new policy π_t is demonstrated by the agent; it is added to the archive and ranked by the expert w.r.t. the other policies in the archive, enriching the set of ordering constraints $\mathcal{C}$;
2. The utility function J_t is built from all constraints in $\mathcal{C}$ (section 2.1);
3. New policies are generated; candidate policy π_{t+1} is selected using an adaptive trade-off between J_t and an empirical exploration term (section 2.2), and the process is iterated.

2.1 Learning the intermediate utility function

Let $\{\mu_1, \ldots \mu_t\}$ denote the indirect representation of all policies in the archive Π up to step t; at this point $\mathcal{C}$ contains up to $\frac{t(t-1)}{2}$ constraints.

[1] The policy archive Π is initialized to a first randomly generated policy π_1, and J_1 is set to the identically null function. The selection of candidate policy π_2 is conducted as in section 2.2. Policies π_1 and π_2 are demonstrated to the expert; the expert ranks them (say $\pi_2 \succ \pi_1$) and the constraint archive $\mathcal{C}$ is initialized accordingly ($\mathcal{C} = \{\pi_2 \succ \pi_1\}$).

Using a standard constrained convex optimization formulation [4, 13], the *policy return estimate* J_t is sought as a linear mapping $J_t(\mu) = \langle w_t, \mu \rangle$ with $w_t \in \mathbb{R}^{n_t}$ solution of (P):

$$(P) \begin{cases} \text{Minimize} & \frac{1}{2} ||\mathbf{w}||^2 + C \sum_{i,j=1, i>j}^{t} \xi_{i,j} \\ \text{subject to} & (\langle \mathbf{w}, \mu_\mathbf{i} \rangle - \langle \mathbf{w}, \mu_\mathbf{j} \rangle \geq 1 - \xi_{i,j}) \text{ and } (\xi_{i,j} \geq 0) \text{ for all } \mu_i \succ \mu_j \end{cases}$$

The utility function J_t features some good properties. Firstly, it is consistent as the weight associated to state-action pairs which have not yet been visited is set to 0. Secondly, by construction J_t is independent on the policy parameterization and can be transferred among different policy spaces. Finally, J_t can be interpreted; $\mathbf{w}_t$ provides some assessment of the state-action pairs, akin the interpretation of a feature weight as a relevance score at the root of SVM-RFE [12]; specifically, if some $(s, a)_i$ is associated a high positive weight $\mathbf{w}_t[i]$, this state-action pair is considered to significantly and positively contribute to the quality of a policy, *comparatively to the policies considered so far*. In particular $\mathbf{w}_t[i]$ might increase or decrease with t.

2.2 New policy generation and selection

The generation of new policies, which can be thought of in terms of self-training, relies on black-box optimization. As already mentioned the use of expected-global improvement methods [14, 7] or even gradient methods (e.g. [19]) is forbidden as J_t is not defined on the policy parameter space $\Theta \subset \mathbb{R}^D$. New policies are thus generated using a Cross-Entropy Maximization-like method [10, 3] on Θ; a Gaussian distribution on Θ is maintained, updating at each step t the center of the distribution and the covariance matrix after the current best policy π_t.

The new policy to be demonstrated to the expert, referred to as candidate policy, is chosen in the set of new policies noted P_t. While the simplest option is to select the one optimizing J_t, such a greedy selection harms the PPL process as it lacks any incentive to explore the state-action space (e.g. the discovery of new state-action pairs is worthless according to J_t). For this reason, candidate policy π_{t+1} is selected in P_t by maximizing the sum of J_t and a weighted exploration term E_t, measuring the diversity Δ of the policy w.r.t the policy archive Π:

$$\pi_{t+1} = \text{arg max} \{J_t(\mu(\pi)) + \alpha_t E_t(\mu(\pi)),\ \pi \in P_t\} \qquad \alpha_t > 0$$

With Exploration term

$$E_t(\mu) = \min \{\Delta(\mu, \mu(\pi_u)),\ \pi_u \in \Pi\}$$
$$\Delta(\mu, \mu') = \frac{||\mu - \mu'||^2}{||\mu||^2 ||\mu'||^2}$$

Dynamic Exploration Exploitation trade-off

$$\alpha_t = \begin{cases} c.\alpha_{t-1} & \text{if } \pi_t \text{ improves on } \Pi_{t-1} \quad c > 1 \\ \frac{1}{c^{1/p}} \alpha_{t-1} & \text{otherwise} \end{cases}$$

Parameter α_t dynamically controls the **exploration vs exploitation trade-off**, accounting for the fact that both the policy distribution and the objective function J_t are non stationary. Therefore, α_t is adjusted by comparing the empirical success rate[2] with the expected success rate of a reference function (usually the sphere function [3]). When the empirical success rate is above (respectively below) the reference one, the amplitude of the perturbations is increased (resp. decreased). Parameter p, empirically adjusted, is used to guarantee that p failures cancel out one success and bring α_t back to its original value.

3 Experimental Validation

This section presents the experimental validation of the PPL approach on a cancer treatment problem. Given an initial state of a patient, defined by its tumor size and toxicity, the goal is to adjust the medicine dosage for reducing tumor size without reaching a too high toxicity level. The experimental setting is same as the one used in [9] where the transition model is provided. The only differences are the time horizon $H = 12$ (the treatment duration is 12 months as opposed to 6 months), and the use of continuous actions (the dosage level is a real value in $(0, 1)$, as opposed to 4 discrete values

[2] That is, the number of times π_t improves on all policies in the archive in a given time window.

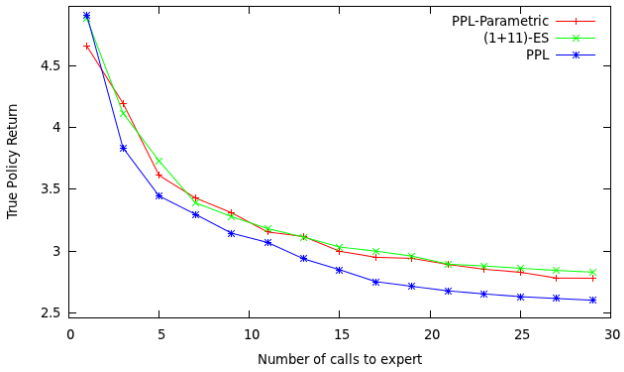

Figure 1: Performance, displayed as the max of tumor size and toxicity over the time horizon, versus the number of calls to the expert; the performance is averaged over 41 independent runs.

in [9]). The policy is implemented as 1-hidden-layer feed-forward neural network with 3 inputs (the current tumor size and toxicity and a bias), 10 neurons in the hidden layer and 1 output being the dosage. J_t is learned using SVM^{rank} [13] with linear kernel and default parameters. Two policy representations are considered, the direct or parametric one ($\Theta \subset \mathbb{R}^{41}$) and the indirect one based on state-action features (these features correspond to partitioning the state and action spaces, with interval width 1 on the state space and .1 on the action space, amounting to circa 500 features).

We set the initial state at 1.3 tumor size and 0 toxicity. During each self-training phase, eleven new policies are generated for 20 rounds, perturbing the current best policy π_t using Gaussian noise $\mathcal{N}(0, \sigma)$, where σ is initialized to 1, with a multiplicative increase (respectively decrease) factor $c = 1.5$ (resp $c^{1/4}$) when the candidate policy does (resp. does not) improve on the former ones. The exploration factor α_t (section 2.2) is initialized to 1, with a multiplicative increase factor $c = 1.5$ and a decrease factor of $\frac{1}{\sqrt{c}}$.

For the sake of reproducibility the expert preferences are emulated, favoring policies that reach the lowest maximum between tumor size and toxicity amongst all months of the treatment. The presented results are averaged over 41 independent runs. The PPL performance is assessed comparatively to two baselines. The first one, referred to as *(1+11)-ES*, uses the same optimization algorithm as in PPL but does not involve any self-training phase (it does not learn J_t). The second one, *PPL-Parametric*, only differs from PPL as J_t is learned on the Θ space (neural network weights) using a Gaussian Kernel; the reported results correspond to the best kernel parameter $\sigma = 10^{-3}$.

As can be seen from Fig. 1, PPL significantly outperforms both baselines, although the performance gap is smaller than in former experiments [2]. This smaller performance gap is explained from the small scale of the problem, resulting in a fast convergence of all methods. The experiments however confirm that the proposed feature-space representation of policies is more effective than the parametric one. Not only *PPL-Parametric* is more computationally demanding as J_t is learned using a Gaussian Kernel (as opposed to a linear kernel in PPL); more importantly, *PPL-Parametric* does not improve on *(1+11)-ES*. This latter fact suggests that the parametric J_t does not provide any information about the most promising policies; the learned utility function is classifying policies at random.

The main limitation of the approach is that PPL happens to converge toward a local optimum. Two such sub-optimal policies are depicted in Fig. 2. This limitation is intrinsically related to the fact that J_t is not learned on the policy space, and the underlying optimization problem is non-convex[3].

[3]In practice, this drawback is alleviated by using random restarts of the stochastic optimization algorithm.

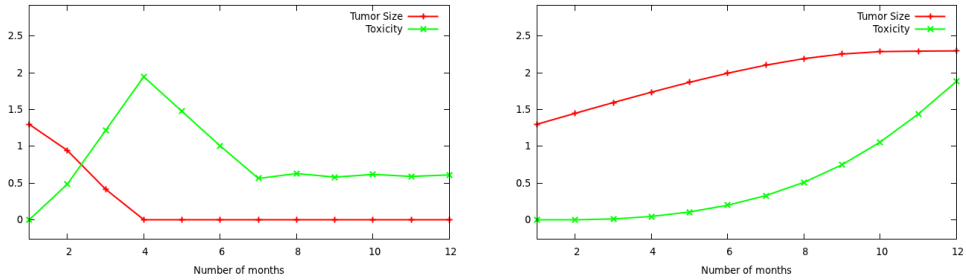

Figure 2: PPL might get stuck to sub-optimal policies. In the left one, the tumor is cured fast causing a peak in toxicity; in the right one the tumor size is contained and the toxicity remains low in the initial stages.

4 Discussion and Perspectives

Preference-based policy learning addresses some limitations of reinforcement learning and inverse reinforcement learning, as it does not require the expert to provide an appropriate reward function, and it does not require the expert to demonstrate a quasi-optimal policy; instead, the expert provides feedback as to whether the current policy improves on the previous ones. In this setting, likened to expensive function optimization, preference learning is used to learn a surrogate model of the objective function.

A main originality of the presented approach compared to related work [14, 7] is to consider an indirect representation of the search space, in order to better capture the expert's preferences; these preferences are more easily related to the actual policy behavior than to its parametric description. The use of such an indirect representation implies the use of derivative-free optimization methods; it further requires new heuristics to address the exploration vs exploitation dilemma. Such a heuristics has been proposed and empirically investigated. Further work will investigate its consistency.

Another perspective is to reconsider preference-based policy learning in a multiple-instance perspective [5]. Typically in a robotic learning context, the expert might assess the robot policy depending on whether some sub-behaviors are appropriate/risky.

A question for further research is then whether one can take advantage simultaneously of both the direct and indirect representation, to propose hybrid policies, using the preference-based model to combine policy fragments in a modular way.

References

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

[2] Riad Akrour, Marc Schoenauer, and Michèle Sebag. Preference-based policy learning. In Gunopulos et al. [11], pages 12–27.

[3] Anne Auger. Convergence results for the (1, lambda)-sa-es using the theory of phi-irreducible markov chains. *Theor. Comput. Sci.*, 334(1-3):35–69, 2005.

[4] G. Bakir, T. Hofmann, B. Scholkopf, A.J. Smola, B. Taskar, and S.V.N. Vishwanathan. *Machine Learning with Structured Outputs*. MIT Press, 2006.

[5] Charles Bergeron, Jed Zaretzki, Curt M. Breneman, and Kristin P. Bennett. Multiple instance ranking. In *Proc. ICML*, pages 48–55, 2008.

[6] Andrew Booker, J. E. Dennis, Paul D. Frank, David B. Serafini, Virginia Torczon, and Michael W. Trosset. A rigorous framework for optimization of expensive functions by surrogates, 1998.

[7] E. Brochu, N. de Freitas, and A. Ghosh. Active preference learning with discrete choice data. In *Advances in Neural Information Processing Systems 20*, pages 409–416, 2008.

[8] S. Calinon, F. Guenter, and A. Billard. On Learning, Representing and Generalizing a Task in a Humanoid Robot. *IEEE transactions on systems, man and cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, 37(2):286–298, 2007.

[9] Weiwei Cheng, Johannes Fürnkranz, Eyke Hüllermeier, and Sang-Hyeun Park. Preference-based policy iteration: Leveraging preference learning for reinforcement learning. In Gunopulos et al. [11], pages 312–327.

[10] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals OR*, 134(1):19–67, 2005.

[11] Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors. *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I*, volume 6911 of *Lecture Notes in Computer Science*. Springer, 2011.

[12] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[13] Thorsten Joachims. Training linear svms in linear time. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *KDD*, pages 217–226. ACM, 2006.

[14] D.R. Jones, M. Schonlau, and W.J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

[15] J. Zico Kolter, Pieter Abbeel, and Andrew Y. Ng. Hierarchical apprenticeship learning with application to quadruped locomotion. In *NIPS*. MIT Press, 2007.

[16] G. Konidaris, S. Kuindersma, A. Barto, and R. Grupen. Constructing skill trees for reinforcement learning agents from demonstration trajectories. In *NIPS*, pages 1162–1170. 2010.

[17] Rémi Munos and Andrew W. Moore. Rates of convergence for variable resolution schemes in optimal control. In Pat Langley, editor, *ICML*, pages 647–654. Morgan Kaufmann, 2000.

[18] A.Y. Ng and S. Russell. Algorithms for inverse reinforcement learning. In P. Langley, editor, *Proc. of the Seventeenth International Conference on Machine Learning (ICML-00)*, pages 663–670. Morgan Kaufmann, 2000.

[19] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

[20] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, 1998.

[21] U. Syed and R. Schapire. A game-theoretic approach to apprenticeship learning. In *NIPS*, pages 1449–1456, 2008.