CAPI: Generalized Classification-based Approximate Policy Iteration

Amir-massoud Farahmand
School of Computer Science
McGill University
Montreal, Quebec, Canada

Doina Precup
School of Computer Science
McGill University
Montreal, Quebec, Canada

André M.S. Barreto
School of Computer Science
McGill University
Montreal, Quebec, Canada

Mohammad Ghavamzadeh
INRIA Lille - Team SequeL
Lille, France

Abstract

Efficient methods for tackling large reinforcement learning problems usually exploit regularities, or intrinsic structures, of the problem in hand. Most current methods benefit from the regularities of either value function or policy, but not both. In this paper, we introduce a general classification-based approximate policy iteration (CAPI) framework, which can benefit from both types of regularities. This framework has two main components: a generic user-specified value function estimator and a weighted classifier that learns a policy based on the estimated value function. The result is a flexible and sample-efficient class of algorithms.

We also use a particular instantiation of CAPI to design an adaptive treatment strategy for HIV-infected patients. Comparison with a state-of-the-art purely value-based reinforcement learning algorithm, Tree-based Fitted Q-Iteration, shows that benefitting from the regularity of both policy and value function can lead to better performance.

Keywords: Reinforcement Learning, Approximate Policy Iteration, Classification-based Approximate Policy Iteration, HIV Drug Scheduling

Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council (NSERC).

1 Introduction

Efficient methods for tackling large reinforcement learning (RL) problems [1] usually exploit special regularities (or structures) of the problem. For example, value-based methods may exploit the smoothness properties of the value function. Many methods have focused on exploiting structure in value function representation and learning, e.g., Farahmand et al. [2], Taylor and Parr [3], Ghavamzadeh et al. [4]. Nonetheless, useful structure can also arise in the policy space. For instance, in many control problems simple policies such as a bang-bang or PID controller can perform quite well if tuned properly. The same is true for many other decision-making problems. In addition to the direct policy search algorithms (e.g., various policy gradient algorithms [5, 6]), one class of methods that try to benefit from the regularities of the policy is the conventional classification-based RL algorithms, e.g., Lagoudakis and Parr [7], Fern et al. [8], Li et al. [9], Lazaric et al. [10]. These methods, however, do not benefit from the regularities of the value function. Thus, they might not be as effective in solving RL problems as one would hope. The goal of this paper is to present a class of algorithms, which we call Classification-based Approximate Policy Iteration (CAPI), that can potentially benefit from the regularities of both value function and policy. This class has been introduced by Farahmand et al. [11] and its theoretical properties have been analyzed there. Here we briefly present the algorithm and describe the application of a particular instantiation of CAPI on the problem of designing an adaptive treatment strategy for HIV-infected patients [12].

Conventional classification-based approaches can be interpreted as a variant of Approximate Policy Iteration (API) that uses rollouts to estimate the action-value function at several points (policy evaluation step) and *projects* the greedy policy obtained at those points onto the predefined space of policies (policy improvement step). In many problems, this approach is helpful because of three main reasons. First, good policies are sometimes simpler to represent and learn than good value functions. Second, even a rough estimate of the value function is often sufficient to separate the best action from the rest, especially when the gap between the value of the greedy action and the rest is large. And finally, even if the best action estimates are noisy (perhaps due to value function imprecision), one can take advantage of powerful classification methods to smooth out the noise. This classification-based approach might particularly be helpful in problems where the classes of "good" policies (parametric or nonparametric) are known in advance, and one simply wants to find good parameters within the class.

Nevertheless, the conventional classification-based RL algorithms suffer from a couple of drawbacks. Most previous work (with the exception of [9, 10, 13, 14] and the related Conservative Policy Iteration approach [15]), uses the $0/1$-loss for training a classifier. This penalizes all mistakes equally and does not consider the relative importance of different regions of the state space. This may lead to surprisingly bad policies [11]. Moreover, the rollout-based estimate of the action-value function does not allow generalization over the state-action space and is quite data-inefficient. This is a big concern in real-world problems where new samples are very expensive or impossible to generate, e.g., adaptive treatment strategy or user dialogue systems. In addition, one cannot easily use rollouts when we only have access to a batch of data, and a generative model or simulator of the environment is not available.

The flexible CAPI framework addresses both issues. The error function used for the classification step in CAPI is weighted according to the difference between the value of the greedy action and those of the other actions. This ensures that the resulting policy closely follows the greedy policy in regions of the state space where the difference between the best action and the rest is considerable (so choosing the wrong action is costly), but pays less attention to regions where all actions are almost the same. Moreover, CAPI allows using any policy evaluation method including, but not restricted to, rollout-based estimates (as in previous work [7, 10]), LSTD [16], policy evaluation version of Fitted Q-Iteration [17, 18], and their regularized variants [2, 19]. This is a significant generalization of existing classification-based RL algorithms, which become special cases of CAPI. This simple change allows us to benefit from the regularities of both the policy and value function, whenever the value and policy spaces are chosen properly. Our theoretical results (reported in [11]) indicate that this extension is indeed sound.

2 Definitions

We consider a *continuous-state, finite-action discounted MDP* $(\mathcal{X}, \mathcal{A}, P, \mathcal{R}, \gamma)$, where $\mathcal{X}$ is a measurable state space (e.g., a subset of $\mathbb{R}^d$), $\mathcal{A}$ is a finite set of actions, P is the transition model, $\mathcal{R}$ is the reward model, and $\gamma \in [0, 1)$ is a discount factor. The mapping $\pi : \mathcal{X} \to \mathcal{A}$ is called a (deterministic) *policy*. As usual, V^π and Q^π denote the value and action-value function for π, while V^* and Q^* denote the corresponding value functions for the optimal policy π^*. A policy π is *greedy* with respect to (w.r.t.) an action-value function Q, denoted by $\pi = \hat{\pi}(\cdot; Q)$, if $\pi(x) = \operatorname{argmax}_{a \in \mathcal{A}} Q(x, a)$ holds for all $x \in \mathcal{X}$.

A recently introduced characterization of the complexity of an RL problem is its *action-gap* regularity [20]. For simplicity, we discuss the action-gap for MDPs with only two actions, i.e., $|\mathcal{A}| = 2$. For any $Q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$, the action-gap function is defined as $\mathrm{g}_Q(x) \triangleq |Q(x, 1) - Q(x, 2)|$ for all $x \in \mathcal{X}$, that is, the difference between the best and second-best action at each state. To understand why the action-gap function is informative, suppose we have an estimate $\hat{Q}^\pi$ of Q^π and we want to perform policy improvement based on $\hat{Q}^\pi$. The greedy policy w.r.t. $\hat{Q}$, i.e., $\hat{\pi}(\cdot; \hat{Q}^\pi)$, should ideally be close

Algorithm 1 CAPI(Π, ν, K)

Input: Policy space Π, State distribution ν, Number of iterations K
Initialize: Let $\pi_{(0)} \in \Pi$ be an arbitrary policy
for $k = 0, 1, \ldots, K-1$ **do**
  Construct a dataset $\mathcal{D}_n^{(k)} = \{X_i\}_{i=1}^n,\ X_i \overset{\text{i.i.d.}}{\sim} \nu$
  $\hat{Q}^{\pi_{(k)}} \leftarrow \text{PolicyEval}(\pi_{(k)})$
  $\pi_{(k+1)} \leftarrow \text{argmin}_{\pi \in \Pi} \hat{L}_n^{\pi_{(k)}}(\pi)$ (action-gap-weighted classification)
end for

to the greedy policy w.r.t. Q^π, i.e., $\hat{\pi}(\cdot; Q^\pi)$. If the action-gap $\mathbf{g}_{Q^\pi}(x)$ is large for some state x, the regret of choosing an action different from $\hat{\pi}(x; Q^\pi)$, roughly speaking, is large; however, confusing the best action with the other one is also less likely. If the action-gap is small, a confusion is more likely to arise, but the regret stemming from the wrong choice will be small. In the next section, we see that the action-gap function is directly used in the formulation of the algorithm. One can prove that the behaviour of the action-gap function for an RL problem influences the difficulty of solving it. This is true for both purely value-based approaches [20] as well as CAPI [11].

3 CAPI Framework

CAPI is an approximate policy iteration framework that takes a policy space Π, a distribution over states $\nu \in \mathcal{M}(\mathcal{X})$, and the number of iterations K as inputs, and returns a policy whose performance should be close to the best policy in Π. Its outline is presented in Algorithm 1. PolicyEval can be any algorithm that computes an estimate $\hat{Q}^\pi$ of Q^π, including: rollout-based estimation (in which case CAPI becomes a nonparametric extension of the DPI algorithm [10]), LSTD-Q [16] and Fitted Q-Iteration [17], or a combination of rollouts and function approximation (in which case CAPI becomes a nonparametric extension of the DPI-Critic algorithm [13] as well as a nonparametric extension of a variant of Classification-based Approximate Modified Policy Iteration of Scherrer et al. [14]).

At each iteration k, the approximate policy improvement step of CAPI is performed by minimizing the following empirical loss function in policy space Π:

$$\hat{L}_n^{\pi_{(k)}}(\pi) \triangleq \int_{\mathcal{X}} \mathbf{g}_{\hat{Q}^{\pi_{(k)}}}(x) \mathbb{I}\{\pi(x) \neq \underset{a \in \mathcal{A}}{\text{argmax}}\, \hat{Q}^{\pi_{(k)}}(x, a)\}\, \mathrm{d}\nu_n, \tag{1}$$

where ν_n is the empirical distribution induced by the samples in $\mathcal{D}_n^{(k)} = \{X_i\}_{i=1}^n$ with $X_i \sim \nu$. The solution to the optimization problem $\pi_{(k+1)} \leftarrow \text{argmin}_{\pi \in \Pi} \hat{L}_n^{\pi_{(k)}}(\pi)$ is the projection of the greedy policy $\hat{\pi}(\cdot; \hat{Q}^{\pi_{(k)}})$, defined only at points $\mathcal{D}_n^{(k)}$, onto policy space Π when the distance measure is weighted according to the estimated action-gap function $\mathbf{g}_{\hat{Q}^{\pi_{(k)}}}$. This should be contrasted with the conventional classification-based approaches [7], which use a uniform weight in all state space, i.e., they minimize $\int_{\mathcal{X}} \mathbb{I}\{\pi(x) \neq \text{argmax}_{a \in \mathcal{A}} \hat{Q}^{\pi_{(k)}}(x, a)\} \mathrm{d}\nu_n$. The statistical properties of CAPI are discussed by Farahmand et al. [11].

As a practical note, the minimization problem $\pi_{(k+1)} \leftarrow \text{argmin}_{\pi \in \Pi} \hat{L}_n^{\pi_{(k)}}(\pi)$ might be difficult to solve for a general policy space Π. The problem is similar (but not identical) to minimizing the $0/1$-loss function in binary classification. A common approach to the non-convex $0/1$-loss optimization is to use a convex surrogate loss, such as action-gap-weighted hinge or exponential loss. This convex relaxation is possible for CAPI too, but we do not formulate it here. Instead, we propose an easy-to-implement solution for this step based on the decision-tree model of classification in the next section.

4 HIV Drug Scheduling

Most of the anti-HIV drugs currently available fall into one of two categories: reverse transcriptase inhibitors (RTI) and protease inhibitors (PI). RTI and PI drugs act differently on the organism, and typical HIV treatments use drug cocktails containing both types of medication. Despite the success of drug cocktails in maintaining low viral loads, there are several complications associated with their long-term use. This has attracted the interest of the scientific community to the problem of optimizing drug-scheduling strategies. Among them, a strategy that has been receiving a lot of attention recently is structured treatment interruption (STI), in which patients undergo alternate cycles with and without the drugs. The scheduling of STI treatments can be seen as a sequential decision-making problem, in which the actions correspond to the types of cocktail that should be administered to a patient [12]. To simplify the problem formulation, it is assumed that RTI and PI drugs are administered at fixed amounts, reducing the available actions to the four possible combinations of drugs. The goal is to minimize the HIV viral load using as little drug as possible.

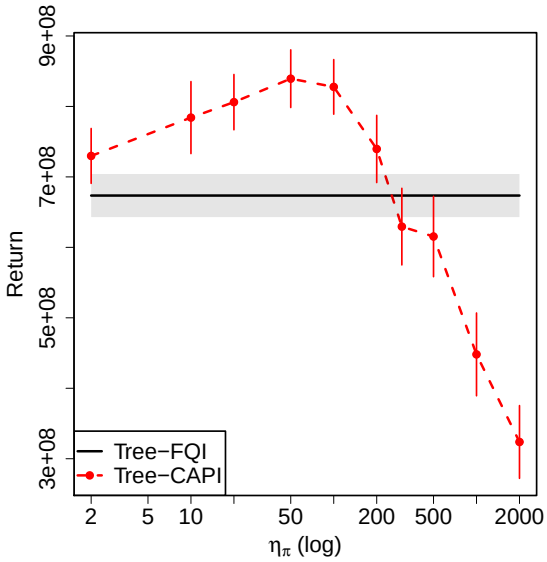

Figure 1: (HIV) Comparing the expected return of Tree-based CAPI vs. Tree-based Fitted Q-Iteration as a function of the complexity of policy space η_π. The policies (in this case STI treatments) were evaluated for $5,000$ days starting from an "unhealthy" state with a high viral load. Error bars and shadowed region show one standard error over 50 runs.

We studied the problem of optimizing STI treatments using a model of the interaction between the immune system and HIV developed by Adams et al. [21] based on real clinical data. All the parameters of the model were set as suggested by Ernst et al. [12]. The methodology adopted in the computational experiments, such as the evaluation of the decision policies and the collection of sample transitions, also followed the protocol proposed by the same authors.

To illustrate the potential benefits of controlling the complexity of the policy space, we compared the pure value-based algorithm adopted by Ernst et al. [12], Fitted Q-Iteration, with CAPI. Following the original experiments, we approximated the value function using an ensemble of 30 decision trees generated by Geurts et al.'s 2006 extra-trees algorithm (we refer to this instantiation of Fitted Q-Iteration as Tree-FQI). Tree-CAPI is similar to Tree-FQI, except that instead of using the greedy policy induced by the current value function approximation, it uses a second ensemble of 30 trees to represent the policy that is the minimizer of (1).

To build the trees representing decision policies, the extra-trees algorithm has to be slightly modified to incorporate the estimated action-gap as its loss function. Such a modification is straightforward, as we now illustrate by showing how a Tree-CAPI policy based on a single tree would perform the action selection. A decision tree defines a partition $\{\mathcal{X}_1, \mathcal{X}_2, \dots\}$ of the state space such that $\bigcup_i \mathcal{X}_i = \mathcal{X}$ and $\mathcal{X}_i \cap \mathcal{X}_j = \emptyset$ for $i \neq j$. Define an index function $I : \mathcal{X} \to \{1, 2, \dots\}$ that returns i if $x \in \mathcal{X}_i$. Thus, $\mathcal{D}^{(k)}(x) \triangleq \mathcal{D}_n^{(k)} \cap \mathcal{X}_{I(x)}$ is the set of data points that are in the same partition as x. The Tree-CAPI policy $\pi_{(k+1)}(x)$ is defined as:

$\pi_{(k+1)}(x) \leftarrow \operatorname{argmin}_{a\in\mathcal{A}} \sum_{X_i\in\mathcal{D}^{(k)}(x)} \mathrm{g}_{\hat{Q}^{\pi_k}}(X_i)\mathbb{I}\{a \neq \hat{\pi}(X_i, \hat{Q}^{\pi_{(k)}})\} \equiv \operatorname{argmin}_{a\in\mathcal{A}} \sum_{X_i\mathcal{D}^{(k)}(x)} \hat{Q}^{\pi_{(k)}}(X_i, \hat{\pi}(X_i, \hat{Q}^{\pi_{(k)}})) - \hat{Q}^{\pi_{(k)}}(X_i, a) \equiv \operatorname{argmax}_{a\in\mathcal{A}} \sum_{X_i\in\mathcal{D}^{(k)}(x)} \hat{Q}^{\pi_{(k)}}(X_i, a)$. In the last equality we used the fact that the term $\hat{Q}^{\pi_k}(X_i, \hat{\pi}(X_i, \hat{Q}^{\pi_k}))$ is not a function of a, so it does not influence the minimizer. This is a very simple rule: Pick the action that maximizes the action-value among all the data points in the same partition as x. Note that this is different from choosing the majority over the greedy actions in the partition, which would be the rule if we neglected the action gap. When we have more than a single tree, which is the case when we use extra-trees, the action to be executed is selected by voting, with ties broken randomly.

The complexity of the models built by the extra-trees algorithm can be controlled by the minimum number of points required to split a node during the construction of the trees [22]. In general, the larger this number is, the simpler the resulting models are. Here we fixed this parameter for the trees representing the value function at $\eta_v = 50$, while the corresponding parameter for the policy trees, denoted by η_π, was varied in the set $\{2, 10, 20, 50, 100, 200, 300, 500, 1000, 2000\}$. Figure 1 shows the result of such an experiment.

As shown in Figure 1, restricting the policy space can have a dramatic impact on the performance of the resulting policies. When $2 \leq \eta_\pi \leq 100$ the policies computed by Tree-CAPI perform better than those computed by Tree-FQI, leading to increases on the empirical return as high as 24%. On the other hand, an overly restricted policy space precludes the representation of the intricacies of an efficient STI treatment, resulting in poor performance. With CAPI, it is possible to adjust the complexity of the policy space to a specific context.

5 Conclusion

We proposed CAPI, a general family of classification-based reinforcement learning algorithms, that allows us to design algorithms that benefit from the regularities of both value function and policy. CAPI uses any policy evaluation method, defines an action-gap-weighted loss function, and then finds the policy minimizing this loss, from a desired policy space.

We showed how to efficiently solve the optimization problem (1) for policy spaces induced by a local method such as decision trees. Extending this to other reasonably general classes of policy spaces (e.g., policies that are defined by the sign of linear combination of basis functions) is an open question. Additionally, an interesting research direction is to extend and analyze CAPI for continuous action spaces.

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, 1998.

[2] Amir-massoud Farahmand, Mohammad Ghavamzadeh, Csaba Szepesvári, and Shie Mannor. Regularized policy iteration. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS - 21)*, pages 441–448. MIT Press, 2009.

[3] Gavin Taylor and Ronald Parr. Kernelized value function approximation for reinforcement learning. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1017–1024, New York, NY, USA, 2009. ACM.

[4] Mohammad Ghavamzadeh, Alessandro Lazaric, Rémi Munos, and Matthew Hoffman. Finite-sample analysis of lasso-TD. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 1177–1184, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5.

[5] Jonathan Baxter and Peter L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, pages 319–350, 2001.

[6] Mohammad Ghavamzadeh and Yaakov Engel. Bayesian policy gradient algorithms. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 457–464. MIT Press, Cambridge, MA, 2007.

[7] Michail G. Lagoudakis and Ronald Parr. Reinforcement learning as classification: Leveraging modern classifiers. In *ICML '03: Proceedings of the 20th international conference on Machine learning*, pages 424–431, 2003.

[8] Alan Fern, Sungwook Yoon, and Robert Givan. Approximate policy iteration with a policy language bias: Solving relational markov decision processes. *Journal of Artificial Intelligence Research*, 25:85–118, 2006.

[9] Lihong Li, Vadim Bulitko, and Russell Greiner. Focus of attention in reinforcement learning. *Journal of Universal Computer Science*, 13(9):1246–1269, 2007.

[10] Alessandro Lazaric, Mohammad Ghavamzadeh, and Rémi Munos. Analysis of a classification-based policy iteration algorithm. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 607–614. Omnipress, 2010.

[11] Amir-massoud Farahmand, Doina Precup, and Mohammad Ghavamzadeh. Generalized classification-based approximate policy iteration. In *Tenth European Workshop on Reinforcement Learning (EWRL)*, 2012.

[12] Damien Ernst, Guy-Bart Stan, Jorge Gongalves, and Louis Wehenkel. Clinical data based optimal STI strategies for HIV: a reinforcement learning approach. In *IEEE Conference on Decision and Control*, pages 667–672. IEEE, 2006.

[13] Victor Gabillon, Alessandro Lazaric, Mohammad Ghavamzadeh, and Bruno Scherrer. Classification-based policy iteration with a critic. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, Bellevue, Washington, USA,, 2011. Omnipress.

[14] Bruno Scherrer, Mohammad Ghavamzadeh, Victor Gabillon, and Matthieu Geist. Approximate modified policy iteration. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

[15] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML)*, pages 267–274, 2002.

[16] Michail G. Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

[17] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

[18] Amir-massoud Farahmand and Doina Precup. Value pursuit iteration. In *Advances in Neural Information Processing Systems (NIPS - 25)*, 2012.

[19] Amir-massoud Farahmand, Mohammad Ghavamzadeh, Csaba Szepesvári, and Shie Mannor. Regularized fitted Q-iteration for planning in continuous-space Markovian Decision Problems. In *Proceedings of American Control Conference (ACC)*, pages 725–730, June 2009.

[20] Amir-massoud Farahmand. Action-gap phenomenon in reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS - 24)*, 2011.

[21] B.M. Adams, H.T. Banks, Hee-Dae Kwon, and H.T. Tran. Dynamic multidrug therapies for HIV: optimal and STI control approaches. *Mathematical Biosciences and Engineering*, 1(2):223–41, 2004.

[22] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 36(1):3–42, 2006.