Bridging the Gap Between Value and Policy Based Reinforcement Learning

Authored by:

Dale Schuurmans
Mohammad Norouzi
Ofir Nachum
Kelvin Xu

Abstract

We establish a new connection between value and policy based reinforcement learning (RL) based on a relationship between softmax temporal value consistency and policy optimality under entropy regularization. Specifically, we show that softmax consistent action values correspond to optimal entropy regularized policy probabilities along any action sequence, regardless of provenance. From this observation, we develop a new RL algorithm, Path Consistency Learning (PCL), that minimizes a notion of soft consistency error along multi-step action sequences extracted from both on- and off-policy traces. We examine the behavior of PCL in different scenarios and show that PCL can be interpreted as generalizing both actor-critic and Q-learning algorithms. We subsequently deepen the relationship by showing how a single model can be used to represent both a policy and the corresponding softmax state values, eliminating the need for a separate critic. The experimental evaluation demonstrates that PCL significantly outperforms strong actor-critic and Q-learning baselines across several benchmarks.

1 Paper Body

Model-free RL aims to acquire an effective behavior policy through trial and error interaction with a black box environment. The goal is to optimize the quality of an agent?s behavior policy in terms of the total expected discounted reward. Model-free RL has a myriad of applications in games [22, 37], robotics [16, 17], and marketing [18, 38], to name a few. Recently, the impact of model-free RL has been expanded through the use of deep neural networks, which promise to replace manual feature engineering with end-to-end learning of value and policy representations. Unfortunately, a key challenge remains how best to combine the advantages of value and policy based RL approaches in the presence of deep function approximators, while mitigating their shortcomings.

Although recent progress has been made in combining value and policy based methods, this issue is not yet settled, and the intricacies of each perspective are exacerbated by deep models. The primary advantage of policy based approaches, such as REINFORCE [45], is that they directly optimize the quantity of interest while remaining stable under function approximation (given a sufficiently small learning rate). Their biggest drawback is sample inefficiency: since policy gradients are estimated from rollouts the variance is often extreme. Although policy updates can be improved by the use of appropriate geometry [14, 27, 32], the need for variance reduction remains paramount. Actor-critic methods have thus become popular [33, 34, 36], because they use value approximators to replace rollout estimates and reduce variance, at the cost of some bias. Nevertheless, on-policy learning remains inherently sample inefficient [10]; by estimating quantities defined by the current policy, either on-policy data must be used, or updating must be sufficiently slow to avoid significant bias. Naive importance correction is hardly able to overcome these shortcomings in practice [28, 29]. 1

Work done as a member of the Google Brain Residency program (g.co/brainresidency) An implementation of PCL can be found at https://github.com/tensorflow/models/tree/master/research/pcl_rl 2

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

By contrast, value based methods, such as Q-learning [44, 22, 30, 42, 21], can learn from any trajectory sampled from the same environment. Such ?off-policy? methods are able to exploit data from other sources, such as experts, making them inherently more sample efficient than on-policy methods [10]. Their key drawback is that off-policy learning does not stably interact with function approximation [35, Chap.11]. The practical consequence is that extensive hyperparameter tuning can be required to obtain stable behavior. Despite practical success [22], there is also little theoretical understanding of how deep Q-learning might obtain near-optimal objective values. Ideally, one would like to combine the unbiasedness and stability of on-policy training with the data efficiency of off-policy approaches. This desire has motivated substantial recent work on off-policy actor-critic methods, where the data efficiency of policy gradient is improved by training an offpolicy critic [19, 21, 10]. Although such methods have demonstrated improvements over on-policy actor-critic approaches, they have not resolved the theoretical difficulty associated with off-policy learning under function approximation. Hence, current methods remain potentially unstable and require specialized algorithmic and theoretical development as well as delicate tuning to be effective in practice [10, 41, 8]. In this paper, we exploit a relationship between policy optimization under entropy regularization and softmax value consistency to obtain a new form of stable off-policy learning. Even though entropy regularized policy optimization is a well studied topic in RL [46, 39, 40, 47, 5, 4, 6, 7]?in fact, one that has been attracting renewed interest from concurrent work [25, 11]?we contribute new observations to this study that are essential for the methods we propose: first, we identify a strong form of path consistency that relates optimal policy probabilities under entropy regulariza-

tion to softmax consistent state values for any action sequence; second, we use this result to formulate a novel optimization objective that allows for a stable form of off-policy actor-critic learning; finally, we observe that under this objective the actor and critic can be unified in a single model that coherently fulfills both roles.

2

Notation & Background

We model an agent?s behavior by a parametric distribution ?? (a — s) defined by a neural network over a finite set of actions. At iteration t, the agent encounters a state st and performs an action at sampled from ?? (a — st). The environment then returns a scalar reward rt and transitions to the next state st+1 . Note: Our main results identify specific properties that hold for arbitrary action sequences. To keep the presentation clear and focus attention on the key properties, we provide a simplified presentation in the main body of this paper by assuming deterministic state dynamics. This restriction is not necessary, and in the Supplementary Material we provide a full treatment of the same concepts generalized to stochastic state dynamics. All of the desired properties continue to hold in the general case and the algorithms proposed remain unaffected. For simplicity, we assume the per-step reward rt and the next state st+1 are given by functions rt = r(st , at) and st+1 = f (st , at) specified by the environment. We begin the formulation by reviewing the key elements of Q-learning [43, 44], which uses a notion of hard-max Bellman backup to enable off-policy TD control. First, observe that the expected discounted reward objective, OER (s, ?), can be recursively expressed as, X OER (s, ?) = ?(a — s) [r(s, a) + ?OER (s0 , ?)] , where s0 = f (s, a) . (1) a ?

Let V (s) denote the optimal state value at a state s given by the maximum value of OER (s, ?) over policies, i.e., V ? (s) = max? OER (s, ?). Accordingly, let ? ? denote the optimal policy that results in V ? (s) (for simplicity, assume there is one unique optimal policy), i.e., ? ? = argmax? OER (s, ?). Such an optimal policy is a one-hot distribution that assigns a probability of 1 to an action with maximal return and 0 elsewhere. Thus we have V ? (s) = OER (s, ? ?) = max(r(s, a) + ?V ? (s0)). a

(2)

This is the well-known hard-max Bellman temporal consistency. Instead of state values, one can equivalently (and more commonly) express this consistency in terms of optimal action values, Q? : Q? (s, a) = r(s, a) + ? max Q? (s0 , a0) . 0 a

2

(3)

Q-learning relies on a value iteration algorithm based on (3), where Q(s, a) is bootstrapped based on successor action values Q(s0 , a0).

3

Softmax Temporal Consistency

In this paper, we study the optimal state and action values for a softmax form of temporal consistency [48, 47, 7], which arises by augmenting the standard expected reward objective with a discounted entropy regularizer. Entropy

regularization [46] encourages exploration and helps prevent early convergence to sub-optimal policies, as has been confirmed in practice (e.g., [21, 24]). In this case, one can express regularized expected reward as a sum of the expected reward and a discounted entropy term, OENT (s, ?) = OER (s, ?) + ? H(s, ?) , (4) where ? ? 0 is a user-specified temperature parameter that controls the degree of entropy regularization, and the discounted entropy H(s, ?) is recursively defined as X H(s, ?) = ?(a — s) [? log ?(a — s) + ? H(s0 , ?)] . (5) a

The objective OENT (s, ?) can then be re-expressed recursively as, X OENT (s, ?) = ?(a — s) [r(s, a) ? ? log ?(a — s) + ?OENT (s0 , ?)] .

(6)

a

Note that when ? = 1 this is equivalent to the entropy regularized objective proposed in [46]. Let V ? (s) = max? OENT (s, ?) denote the soft optimal state value at a state s and let ? ? (a — s) denote the optimal policy at s that attains the maximum of OENT (s, ?). When ? ¿ 0, the optimal policy is no longer a one-hot distribution, since the entropy term prefers the use of policies with more uncertainty. We characterize the optimal policy ? ? (a — s) in terms of the OENT -optimal state values of successor states V ? (s0) as a Boltzmann distribution of the form, ? ? (a — s) ? exp{(r(s, a) + ?V ? (s0))/? } .

(7)

It can be verified that this is the solution by noting that the OENT (s, ?) objective is simply a ? -scaled constant-shifted KL-divergence between ? and ? ? , hence the optimum is achieved when ? = ? ? . To derive V ? (s) in terms of V ? (s0), the policy ? ? (a — s) can be substituted into (6), which after some manipulation yields the intuitive definition of optimal state value in terms of a softmax (i.e., log-sum-exp) backup, X exp{(r(s, a) + ?V ? (s0))/? } . (8) V ? (s) = OENT (s, ? ?) = ? log a

Note that in the ? ? 0 limit one recovers the hard-max state values defined in (2). Therefore we can equivalently state softmax temporal consistency in terms of optimal action values Q? (s, a) as, X Q? (s, a) = r(s, a) + ?V ? (s0) = r(s, a) + ?? log exp(Q? (s0 , a0)/?) . (9) 0 a

Now, much like Q-learning, the consistency equation (9) can be used to perform one-step backups to asynchronously bootstrap Q? (s, a) based on Q? (s0 , a0). In the Supplementary Material we prove that such a procedure, in the tabular case, converges to a unique fixed point representing the optimal values. We point out that the notion of softmax Q-values has been studied in previous work (e.g., [47, 48, 13, 5, 3, 7]). Concurrently to our work, [11] has also proposed a soft Q-learning algorithm for continuous control that is based on a similar notion of softmax temporal consistency. However, we contribute new observations below that lead to the novel training principles we explore.

4

Consistency Between Optimal Value & Policy

We now describe the main technical contributions of this paper, which lead to the development of two novel off-policy RL algorithms in Section 5. The first key observation is that, for the softmax 3

value function V ? in (8), the quantity exp{V ? (s)/? } also serves as the normalization factor of the optimal policy ? ? (a — s) in (7); that is, ? ? (a — s) =

exp{(r(s, a) + ?V ? (s0))/? } . exp{V ? (s)/? }

(10)

Manipulation of (10) by taking the log of both sides then reveals an important connection between the optimal state value V ? (s), the value V ? (s0) of the successor state s0 reached from any action a taken in s, and the corresponding action probability under the optimal log-policy, log ? ? (a — s). Theorem 1. For ? ¿ 0, the policy ? ? that maximizes OENT and state values V ? (s) = max? OENT (s, ?) satisfy the following temporal consistency property for any state s and action a (where s0 = f (s, a)), V ? (s) ? ?V ? (s0) = r(s, a) ? ? log ? ? (a — s) .

(11)

Proof. All theorems are established for the general case of a stochastic environment and discounted infinite horizon problems in the Supplementary Material. Theorem 1 follows as a special case. Note that one can also characterize ? ? in terms of Q? as ? ? (a — s) = exp{(Q? (s, a) ? V ? (s))/? } .

(12)

An important property of the one-step softmax consistency established in (11) is that it can be extended to a multi-step consistency defined on any action sequence from any given state. That is, the softmax optimal state values at the beginning and end of any action sequence can be related to the rewards and optimal log-probabilities observed along the trajectory. Corollary 2. For ? ¿ 0, the optimal policy ? ? and optimal state values V ? satisfy the following extended temporal consistency property, for any state s1 and any action sequence a1 , ..., at?1 (where si+1 = f (si , ai)): V ? (s1) ? ? t?1 V ? (st) =

t?1 X

? i?1 [r(si , ai) ? ? log ? ? (ai — si)] .

(13)

i=1

Proof. The proof in the Supplementary Material applies (the generalized version of) Theorem 1 to any s1 and sequence a1 , ..., at?1 , summing the left and right hand sides of (the generalized version of) (11) to induce telescopic cancellation of intermediate state values. Corollary 2 follows as a special case. Importantly, the converse of Theorem 1 (and Corollary 2) also holds: Theorem 3. If a policy ?(a — s) and state value function V (s) satisfy the consistency property (11) for all states s and actions a (where s0 = f (s, a)), then ? = ? ? and V = V ? . (See the Supplementary Material.) Theorem 3 motivates the use of one-step and multi-step path-wise consistencies as the foundation of RL algorithms that aim to learn parameterized policy and value estimates by minimizing the discrepancy between the left and right hand sides of (11) and (13).

5

Path Consistency Learning (PCL)

The temporal consistency properties between the optimal policy and optimal state values developed above lead to a natural path-wise objective for training a policy ?? , parameterized by ?, and a state value function V? , parameterized by ?, via the minimization of a soft consistency error. Based on (13), we first define a notion of soft consistency for a d-length sub-trajectory si:i+d ? (si , ai , . . . , si+d?1 , ai+d?1 , si+d) as a function of ? and ?: Xd?1 C(si:i+d , ?, ?) = ?V? (si) + ? d V? (si+d) + ? j [r(si+j , ai+j) ? ? log ?? (ai+j — si+j)] . (14) j=0

The goal of a learning algorithm can then be to find V? and ?? such that C(si:i+d , ?, ?) is as close to 0 as possible for all sub-trajectories si:i+d . Accordingly, we propose a new learning algorithm, called 4

Path Consistency Learning (PCL), that attempts to minimize the squared soft consistency error over a set of sub-trajectories E, X 1 OPCL (?, ?) = C(si:i+d , ?, ?)2 . (15) 2 si:i+d ?E

The PCL update rules for ? and ? are derived by calculating the gradient of (15). For a given trajectory si:i+d these take the form, Xd?1 ?? = ?? C(si:i+d , ?, ?) ? j ?? log ?? (ai+j — si+j) , (16) j=0

?? = ?v C(si:i+d , ?, ?) ?? V? (si) ? ? d ?? V? (si+d) , (17) where ?v and ?? denote the value and policy learning rates respectively. Given that the consistency property must hold on any path, the PCL algorithm applies the updates (16) and (17) both to trajectories sampled on-policy from ?? as well as trajectories sampled from a replay buffer. The union of these trajectories comprise the set E used in (15) to define OPCL . Specifically, given a fixed rollout parameter d, at each iteration, PCL samples a batch of on-policy trajectories and computes the corresponding parameter updates for each sub-trajectory of length d. Then PCL exploits off-policy trajectories by maintaining a replay buffer and applying additional updates based on a batch of episodes sampled from the buffer at each iteration. We have found it beneficial to sample replay episodes proportionally to exponentiated reward, mixed with a uniform distribution, although we did not exhaustively experiment with this sampling procedure. In particular, we sample a full episode s0:T from the replay buffer of size B with probability 0.1/B + 0.9 ? PT ?1 exp(? i=0 r(si , ai))/Z, where we use no discounting on the sum of rewards, Z is a normalization factor, and ? is a hyper-parameter. Pseudocode of PCL is provided in the Appendix. We note that in stochastic settings, our squared inconsistency objective approximated by Monte Carlo samples is a biased estimate of the true squared inconsistency (in which an expectation over stochastic dynamics occurs inside rather than outside the square). This issue arises in Q-learning as well, and others have proposed possible remedies which can also be applied to PCL [2]. 5.1

Unified Path Consistency Learning (Unified PCL)

The PCL algorithm maintains a separate model for the policy and the state value approximation. However, given the soft consistency between the state and action value functions (e.g.,in (9)), one can express the soft consistency errors strictly in terms of Q-values. Let Q? denote a model of action values parameterized by ?, based on which one can estimate both the state values and the policy as, X V? (s) = ? log exp{Q? (s, a)/? } , (18) a

?? (a — s)
=
exp{(Q? (s, a) ? V? (s))/? } .
(19)

Given this unified parameterization of policy and value, we can formulate an alternative algorithm, called Unified Path Consistency Learning (Unified PCL), which optimizes the same objective (i.e., (15)) as PCL but differs by combining the policy and value function into a single model. Merging the policy and value function models in this way is significant because it presents a new actor-critic paradigm where the policy (actor) is not distinct from the values (critic). We note that in practice, we have found it beneficial to apply updates to ? from V? and ?? using different learning rates, very much like PCL. Accordingly, the update rule for ? takes the form, Xd?1 ?? = ?? C(si:i+d , ?) ? j ?? log ?? (ai+j — si+j) + (20) j=0

?v C(si:i+d , ?) ?? V? (si) ? ? d ?? V? (si+d) . (21) 5.2

Connections to Actor-Critic and Q-learning

To those familiar with advantage-actor-critic methods [21] (A2C and its asynchronous analogue A3C) PCL?s update rules might appear to be similar. In particular, advantage-actor-critic is an on-policy method that exploits the expected value function, X V ? (s) = ?(a — s) [r(s, a) + ?V ? (s0)] , (22) a

5

to reduce the variance of policy gradient, in service of maximizing the expected reward. As in PCL, two models are trained concurrently: an actor ?? that determines the policy, and a critic V? that is trained to estimate V ?? . A fixed rollout parameter d is chosen, and the advantage of an on-policy trajectory si:i+d is estimated by Xd?1 A(si:i+d , ?) = ?V? (si) + ? d V? (si+d) + ? j r(si+j , ai+j) . (23) j=0

The advantage-actor-critic updates for ? and ? can then be written as, ??
= ?? Esi:i+d —? [A(si:i+d , ?)?? log ?? (ai —si)] ,
(24)
??
= ?v Esi:i+d —? [A(si:i+d , ?)?? V? (si)] ,
(25)

where the expectation Esi:i+d —? denotes sampling from the current policy ?? . These updates exhibit a striking similarity to the updates expressed in (16) and (17). In fact, if one takes PCL with ? ? 0 and omits the replay buffer, a slight variation of A2C is recovered. In this sense, one can interpret PCL as a generalization of A2C. Moreover, while A2C is restricted to on-policy samples, PCL minimizes an inconsistency measure that is defined on any path, hence it can exploit replay data to enhance its efficiency via off-policy learning. It is also important to note that for A2C, it is essential that V? tracks the non-stationary target V ?? to ensure suitable variance reduction. In PCL, no such tracking is required. This difference is more dramatic in Unified PCL, where a single model is trained both as an actor and a critic. That is, it is not necessary to have a separate actor and critic; the actor itself can serve as its own critic. One can also compare PCL to hard-max temporal consistency RL algorithms,

such as Q-learning [43]. In fact, setting the rollout to d = 1 in Unified PCL leads to a form of soft Q-learning, with the degree of softness determined by ? . We therefore conclude that the path consistency-based algorithms developed in this paper also generalize Q-learning. Importantly, PCL and Unified PCL are not restricted to single step consistencies, which is a major limitation of Q-learning. While some have proposed using multi-step backups for hard-max Q-learning [26, 21], such an approach is not theoretically sound, since the rewards received after a non-optimal action do not relate to the hard-max Q-values Q? . Therefore, one can interpret the notion of temporal consistency proposed in this paper as a sound generalization of the one-step temporal consistency given by hard-max Q-values.

6

Related Work

Connections between softmax Q-values and optimal entropy-regularized policies have been previously noted. In some cases entropy regularization is expressed in the form of relative entropy [4, 6, 7, 31], and in other cases it is the standard entropy [47]. While these papers derive similar relationships to (7) and (8), they stop short of stating the single- and multi-step consistencies over all action choices we highlight. Moreover, the algorithms proposed in those works are essentially single-step Q-learning variants, which suffer from the limitation of using single-step backups. Another recent work [25] uses the softmax relationship in the limit of ? ? 0 and proposes to augment an actor-critic algorithm with offline updates that minimize a set of single-step hard-max Bellman errors. Again, the methods we propose are differentiated by the multi-step path-wise consistencies which allow the resulting algorithms to utilize multi-step trajectories from off-policy samples in addition to on-policy samples. The proposed PCL and Unified PCL algorithms bear some similarity to multi-step Q-learning [26], which rather than minimizing one-step hard-max Bellman error, optimizes a Q-value function approximator by unrolling the trajectory for some number of steps before using a hard-max backup. While this method has shown some empirical success [21], its theoretical justification is lacking, since rewards received after a non-optimal action no longer relate to the hard-max Q-values Q? . In contrast, the algorithms we propose incorporate the log-probabilities of the actions on a multi-step rollout, which is crucial for the version of softmax consistency we consider. Other notions of temporal consistency similar to softmax consistency have been discussed in the RL literature. Previous work has used a Boltzmann weighted average operator [20, 5]. In particular, this operator has been used by [5] to propose an iterative algorithm converging to the optimal maximum reward policy inspired by the work of [15, 39]. While they use the Boltzmann weighted average, they briefly mention that a softmax (log-sum-exp) operator would have similar theoretical properties. More recently [3] proposed a mellowmax operator, defined as log-average-exp. These log-averageexp operators share a similar non-expansion property, and the proofs of non-expansion are related. 6

Synthetic Tree

Copy

Synthetic Tree 20
DuplicatedInput
Copy
35
RepeatCopy
100
14
30 15
RepeatCopy
DuplicatedInput
16
80
12
25
10
20
60
8
10
15 10
5 0 0
50
4
5
2
0
0
100
40
6
0
Reverse
1000
2000
20 0 0
ReversedAddition
Reverse
2000
3000
ReversedAddition3
20
4000
Hard ReversedAddition
30 25
15
25

20
2000
Hard ReversedAddition
30
25
0
ReversedAddition3
ReversedAddition
35
30
1000
20
20 10
15
15
10
10
5
5
0
0 0
5000
10000
15 10
5 5 0 0
5000
PCL
10000
0 0
A3C
20000
40000
60000
0
5000
10000
DQN

Figure 1: The results of PCL against A3C and DQN baselines. Each plot shows average reward across 5 random training runs (10 for Synthetic Tree) after choosing best hyperparameters. We also show a single standard deviation bar clipped at the min and max. The x-axis is number of training iterations. PCL exhibits comparable performance to A3C in some tasks, but clearly outperforms A3C on the more challenging tasks. Across all tasks, the performance of DQN is worse than PCL. Additionally it is possible to show that when restricted to an infinite horizon setting, the fixed point of the mellowmax operator is a constant shift of the Q? investigated here. In all these cases, the suggested training

algorithm optimizes a single-step consistency unlike PCL and Unified PCL, which optimizes a multi-step consistency. Moreover, these papers do not present a clear relationship between the action values at the fixed point and the entropy regularized expected reward objective, which was key to the formulation and algorithmic development in this paper. Finally, there has been a considerable amount of work in reinforcement learning using off-policy data to design more sample efficient algorithms. Broadly speaking, these methods can be understood as trading off bias [36, 34, 19, 9] and variance [28, 23]. Previous work that has considered multi-step off-policy learning has typically used a correction (e.g., via importance-sampling [29] or truncated importance sampling with bias correction [23], or eligibility traces [28]). By contrast, our method defines an unbiased consistency for an entire trajectory applicable to on- and off-policy data. An empirical comparison with all these methods remains however an interesting avenue for future work.

7

Experiments

We evaluate the proposed algorithms, namely PCL & Unified PCL, across several different tasks and compare them to an A3C implementation, based on [21], and an implementation of double Q-learning with prioritized experience replay, based on [30]. We find that PCL can consistently match or beat the performance of these baselines. We also provide a comparison between PCL and Unified PCL and find that the use of a single unified model for both values and policy can be competitive with PCL. These new algorithms are easily amenable to incorporate expert trajectories. Thus, for the more difficult tasks we also experiment with seeding the replay buffer with 10 randomly sampled expert trajectories. During training we ensure that these trajectories are not removed from the replay buffer and always have a maximal priority. The details of the tasks and the experimental setup are provided in the Appendix. 7.1

Results

We present the results of each of the variants PCL, A3C, and DQN in Figure 1. After finding the best hyperparameters (see the Supplementary Material), we plot the average reward over training iterations for five randomly seeded runs. For the Synthetic Tree environment, the same protocol is performed but with ten seeds instead. 7

Synthetic Tree

Copy

Synthetic Tree 20

DuplicatedInput

Copy

35

RepeatCopy

100

14

30 15

RepeatCopy

DuplicatedInput

16
80
12
25
10
20
60
8
10
15 10
5 0 0
50
4
5
2
0
0
100
40
6
0
Reverse
1000
2000
20 0 0
ReversedAddition
Reverse
1000
2000
3000
0
ReversedAddition3
ReversedAddition
ReversedAddition3
30
30
30
25
25
25
25
20
20
20
20
15

15
15
15
10
10
10
10
5
5
5
5
0
0
0
5000
10000
0
5000
10000
4000
Hard ReversedAddition
30
0
2000
Hard ReversedAddition
0 0
PCL
20000
40000
60000
0
5000
10000
Unified PCL

Figure 2: The results of PCL vs. Unified PCL. Overall we find that using a single model for both values and policy is not detrimental to training. Although in some of the simpler tasks PCL has an edge over Unified PCL, on the more difficult tasks, Unified PCL preforms better. Reverse
ReversedAddition
ReversedAddition3
Reverse
ReversedAddition
ReversedAddition3
Hard ReversedAddition Hard ReversedAddition
30
30

30
30
25
25
25
25
20
20
20
20
15
15
15
15
10
10
10
10
5
5
5
0
0 0
2000
4000
5
0 0
2000
4000
0 0
PCL
20000
40000
60000
0
5000
10000
PCL + Expert

Figure 3: The results of PCL vs. PCL augmented with a small number of expert trajectories on the hardest algorithmic tasks. We find that incorporating expert trajectories greatly improves performance.

The gap between PCL and A3C is hard to discern in some of the more simple tasks such as Copy, Reverse, and RepeatCopy. However, a noticeable gap is observed in the Synthetic Tree and DuplicatedInput results and more significant gaps are clear in the harder tasks, including ReversedAddition, ReversedAddition3, and Hard ReversedAddition. Across all of the experiments,

it is clear that the prioritized DQN performs worse than PCL. These results suggest that PCL is a competitive RL algorithm, which in some cases significantly outperforms strong baselines. We compare PCL to Unified PCL in Figure 2. The same protocol is performed to find the best hyperparameters and plot the average reward over several training iterations. We find that using a single model for both values and policy in Unified PCL is slightly detrimental on the simpler tasks, but on the more difficult tasks Unified PCL is competitive or even better than PCL. We present the results of PCL along with PCL augmented with expert trajectories in Figure 3. We observe that the incorporation of expert trajectories helps a considerable amount. Despite only using a small number of expert trajectories (i.e., 10) as opposed to the mini-batch size of 400, the inclusion of expert trajectories in the training process significantly improves the agent?s performance. We performed similar experiments with Unified PCL and observed a similar lift from using expert trajectories. Incorporating expert trajectories in PCL is relatively trivial compared to the specialized methods developed for other policy based algorithms [1, 12]. While we did not compare to other algorithms that take advantage of expert trajectories, this success shows the promise of using pathwise consistencies. Importantly, the ability of PCL to incorporate expert trajectories without requiring adjustment or correction is a desirable property in real-world applications such as robotics. 8

8

Conclusion

We study the characteristics of the optimal policy and state values for a maximum expected reward objective in the presence of discounted entropy regularization. The introduction of an entropy regularizer induces an interesting softmax consistency between the optimal policy and optimal state values, which may be expressed as either a single-step or multi-step consistency. This softmax consistency leads to the development of Path Consistency Learning (PCL), an RL algorithm that resembles actor-critic in that it maintains and jointly learns a model of the state values and a model of the policy, and is similar to Q-learning in that it minimizes a measure of temporal consistency error. We also propose the variant Unified PCL which maintains a single model for both the policy and the values, thus upending the actor-critic paradigm of separating the actor from the critic. Unlike standard policy based RL algorithms, PCL and Unified PCL apply to both on-policy and off-policy trajectory samples. Further, unlike value based RL algorithms, PCL and Unified PCL can take advantage of multi-step consistencies. Empirically, PCL and Unified PCL exhibit a significant improvement over baseline methods across several algorithmic benchmarks.

9

Acknowledgment

We thank Rafael Cosman, Brendan O?Donoghue, Volodymyr Mnih, George Tucker, Irwan Bello, and the Google Brain team for insightful comments and discussions.

2 References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In Proceedings of the twenty-first international conference on Machine learning, page 1. ACM, 2004. [2] A. Antos, C. Szepesv?ri, and R. Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. Machine Learning, 71(1):89? 129, 2008. [3] K. Asadi and M. L. Littman. arXiv:1612.05628, 2016.

A new softmax operator for reinforcement learning.

[4] M. G. Azar, V. G?mez, and H. J. Kappen. Dynamic policy programming with function approximation. AISTATS, 2011. [5] M. G. Azar, V. G?mez, and H. J. Kappen. Dynamic policy programming. JMLR, 13(Nov), 2012. [6] M. G. Azar, V. G?mez, and H. J. Kappen. Optimal control as a graphical model inference problem. Mach. Learn. J., 87, 2012. [7] R. Fox, A. Pakman, and N. Tishby. G-learning: Taming the noise in reinforcement learning via soft updates. UAI, 2016. [8] A. Gruslys, M. G. Azar, M. G. Bellemare, and R. Munos. The reactor: A sample-efficient actor-critic architecture. arXiv preprint arXiv:1704.04651, 2017. [9] S. Gu, E. Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. ICRA, 2016. [10] S. Gu, T. Lillicrap, Z. Ghahramani, R. E. Turner, and S. Levine. Q-Prop: Sample-efficient policy gradient with an off-policy critic. ICLR, 2017. [11] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine. Reinforcement learning with deep energy-based policies. arXiv:1702.08165, 2017. [12] J. Ho and S. Ermon. Generative adversarial imitation learning. In Advances in Neural Information Processing Systems, pages 4565?4573, 2016. 9

[13] D.-A. Huang, A.-m. Farahmand, K. M. Kitani, and J. A. Bagnell. Approximate maxent inverse optimal control and its application for mental simulation of human interactions. 2015. [14] S. Kakade. A natural policy gradient. NIPS, 2001. [15] H. J. Kappen. Path integrals and symmetry breaking for optimal control theory. Journal of statistical mechanics: theory and experiment, 2005(11):P11011, 2005. [16] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. IJRR, 2013. [17] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. JMLR, 17(39), 2016. [18] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. 2010. [19] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. ICLR, 2016. [20] M. L. Littman. Algorithms for sequential decision making. PhD thesis, Brown University, 1996. [21] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. ICML, 2016. [22] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning. Nature, 2015. [23] R. Munos, T. Stepleton, A. Harutyunyan, and M. Bellemare. Safe and efficient off-policy reinforcement learning. NIPS, 2016. [24] O. Nachum, M. Norouzi, and D. Schuurmans. Improving policy gradient by exploring underappreciated rewards. ICLR, 2017. [25] B. O?Donoghue, R.

Munos, K. Kavukcuoglu, and V. Mnih. PGQ: Combining policy gradient and Q-learning. ICLR, 2017. [26] J. Peng and R. J. Williams. Incremental multi-step Q-learning. Machine learning, 22(1-3):283? 290, 1996. [27] J. Peters, K. M?ling, and Y. Altun. Relative entropy policy search. AAAI, 2010. [28] D. Precup. Eligibility traces for off-policy policy evaluation. Computer Science Department Faculty Publication Series, page 80, 2000. [29] D. Precup, R. S. Sutton, and S. Dasgupta. Off-policy temporal-difference learning with function approximation. 2001. [30] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. ICLR, 2016. [31] J. Schulman, X. Chen, and P. Abbeel. Equivalence between policy gradients and soft Q-learning. arXiv:1704.06440, 2017. [32] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel. Trust region policy optimization. ICML, 2015. [33] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. ICLR, 2016. [34] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. ICML, 2014. [35] R. S. Sutton and A. G. Barto. Introduction to Reinforcement Learning. MIT Press, 2nd edition, 2017. Preliminary Draft. [36] R. S. Sutton, D. A. McAllester, S. P. Singh, Y. Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. NIPS, 1999.
10

[37] G. Tesauro. Temporal difference learning and TD-gammon. CACM, 1995. [38] G. Theocharous, P. S. Thomas, and M. Ghavamzadeh. Personalized ad recommendation systems for life-time value optimization with guarantees. IJCAI, 2015. [39] E. Todorov. Linearly-solvable Markov decision problems. NIPS, 2006. [40] E. Todorov. Policy gradients in linearly-solvable MDPs. NIPS, 2010. [41] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas. Sample efficient actor-critic with experience replay. ICLR, 2017. [42] Z. Wang, N. de Freitas, and M. Lanctot. Dueling network architectures for deep reinforcement learning. ICLR, 2016. [43] C. J. Watkins. Learning from delayed rewards. PhD thesis, University of Cambridge England, 1989. [44] C. J. Watkins and P. Dayan. Q-learning. Machine learning, 8(3-4):279?292, 1992. [45] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn. J., 1992. [46] R. J. Williams and J. Peng. Function optimization using connectionist reinforcement learning algorithms. Connection Science, 1991. [47] B. D. Ziebart. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. PhD thesis, CMU, 2010. [48] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. AAAI, 2008.
11