Taming the Noise in Reinforcement Learning via Soft Updates:

G-Learning

Roy Fox, Ari Pakman, Naftali Tishby

02/20/2018

Outline

Outline

Stakes and purpose

- RL : Learning on noisy environment can be a real challenge.

Stakes and purpose

- RL : Learning on noisy environment can be a real challenge.
- Q-learning performs poorly in noisy environments. Why ? In early stages, the min/max operator brings a bias, which slow down the estimation.

Stakes and purpose

- RL : Learning on noisy environment can be a real challenge.
- Q-learning performs poorly in noisy environments. Why ? In early stages, the min/max operator brings a bias, which slow down the estimation.
- Approach proposed : add a penalization to the cost/reward using Kullback-Leibler divergence (information theory). Thus, we can softly shifts from a randomized policy to a deterministic one.

Stakes and purpose

- RL : Learning on noisy environment can be a real challenge.
- Q-learning performs poorly in noisy environments. Why ? In early stages, the min/max operator brings a bias, which slow down the estimation.
- Approach proposed : add a penalization to the cost/reward using Kullback-Leibler divergence (information theory). Thus, we can softly shifts from a randomized policy to a deterministic one.
- This is G-learning, which [spoiler] performs better in noisy environments, regarding the results

Outline

Notation and hypothesis

- **S** and **A** are respectively the state space and the action space (finite).
- Stochastic Decision Process : $a_t \sim \pi(a_t|s_t)$(action), $c_t \sim \theta(s_t, a_t)$ (cost) and $s_{t+1} \sim p(s_t, a_t)$ (state) [non deterministic case]
- $V^\pi(s) = \sum_t \gamma^t E[c_t|s_0 = s]$
- $Q^\pi(s, a) = \sum_t \gamma^t E[c_t|s_0 = s, a_0 = a] = E_\theta[c|s, a] + \gamma E_p[V^\pi(s')|s, a]$
- Goal of Q-learning : find $Q^*(s, a) = \min_\pi Q^\pi(s, a)$
- In this paper, Q is model-free (table).

Outline

Q-Learning algorithm

Q-Learning

The Bellman equation and the temporal differences lead to :

$$Q(s_t, a_t) \leftarrow (1 - \alpha_t) Q(s_t, a_t) + \alpha_t \Big(c_t + \gamma \sum_{a'} \pi(a'|s_{t+1}) Q(s_{t+1}, a') \Big) \quad (1)$$

with some learning rate $0 \leq \alpha_t \leq 1$ and the the following policy :

$$\pi(a|s) = \delta_{a, a^*(s)}; \; a^* = \underset{a}{\text{argmin}}\, Q(s, a)$$

If the exploration policy returns to each state-action pair infinitely many times and if the learning rates satisfies :

$$\sum_t a_t = \infty; \; \sum_t \alpha_t^2 < \infty$$

then Q converge to Q^* with probability 1.

Outline

Optimistic bias

Q-learning induce a negative bias because of the *min* operator.

- Assume $\hat{Q}(s,a)$, is an unbiased estimate of $Q^*(s,a)$.
- Jensen inequality for f concave : $E[f(X)] \leq f(E[X])$
- $E[\min_a \hat{Q}(s,a)] \leq \min_a E(\hat{Q}(s,a)) = \min_a Q^*(s,a))$
- Equality only if argmin $\hat{Q}(s,a)$ is argmin $Q^*(s,a)$ with probability 1 $(Var(\hat{Q}(s,a)) = 0)$

There is an optimistic bias (winner's curse in auction theory) : the cost appear lower than it is : this is a problem.
Thus, at the end, the goal is to have a low $Var(\hat{Q}(s,a))$, by increasing the sample size which make the algorithm converge (but maybe very slowly).

Note : The expectation is with respect to any randomness in state transition, cost, exploration, or because of the use of a function approximation (not considered in the article). That's why this is a problem especially in noisy environments !

Outline

The interplay of value bias and policy suboptimality

We consider the effect of the optimistic bias on V^{π} with $\pi = \text{argmin}_a(\hat{Q}(s,a))$ regarding the gap $\delta = Q^*(s,a') - V^*(s)$ where a' is sub-optimal.

- If $Var(\hat{Q}(s,a)) << \delta$, then, a' will be sub-optimal with high probability, as desired (Q-learning converge normally)
- If $Var(\hat{Q}(s,a)) >> \delta$, then, confusing such a' has a limited effect, since a' is near-optimal.
- If $Var(\hat{Q}(s,a)) \sim \delta$, then this is a problem ! "a" probably suboptimal, and propagation of bias between states via updating !

Outline

A dynamic optimism-uncertainty loop

Usually, we use a ϵ-greedy exploration policy to accelerate the bias reduction : high variance is self corrected : it's called a "dynamic form of optimism under uncertainty" (did not catch this formulation). Optimism is generated by noise and self-corrected through exploration.

The purpose of this paper : explicitly represent the uncertainty and avoid the hard-min operator. This can be done by penalizing deterministic policies at the early learning stage.

Outline

The Free-energy function F : Notations

We define :

- Stochastic policy prior : $\rho(a|s)$
- Informative cost of a learned policy : $\log \frac{\pi(a|s)}{\rho(a|s)}$ (penalizes deviations from prior and regularizes to avoid deterministic policy)
- Total discounted expected information cost : $I^\pi(s) = \sum_t \gamma^t E[g^\pi(s_t, a_t)|s_0 = s]$

Then, we define a new total cost called *free energy function*:

$$F^\pi(s) = V^\pi(s) + \frac{1}{\beta} I^\pi(s)$$

For the moment, β is fixed.

The Free-energy function G : Notations

Similarly to the function Q, we define the *state- action free-energy function* :

$$G^{\pi}(s,a) = E[c|s,a] + \gamma E[F^{\pi}(s',a)|s,a] \tag{2}$$
$$= \sum_t \gamma^t E[c_t + \frac{\gamma}{\beta} g^{\pi}(s_{t+1},a_{t+1}))|a_0 = a, s_0 = s] \tag{3}$$

The informative cost is not taken into account for the first step, since we have already chosen the first action a.

Relationship between G and F

We can compute F^π with G^π by computing the expected value under all action :

$$F^\pi(s) = \sum_a \pi(a|s)\left[\frac{1}{\beta}\log\frac{\pi(a|s)}{\rho(a|s)} + G^\pi(a,s)\right] \tag{4}$$

We fix s. Using this, F^π and G^π has a null gradient at :

$$\pi(a|s) = \frac{\rho(a|s)e^{-\beta G^{\pi(s,a)}}}{\sum_{a'}\rho(a'|s)e^{-\beta G^{\pi(s,a')}}} \tag{5}$$

which is the optimal policy.

Opimality G^*

Evaluated at the optimal policy, F^π becomes :

$$F^*(s) = -\frac{1}{\beta} \log \sum_a \rho(a|s) e^{-\beta G^*(s,a)} \tag{6}$$

We plug this expression into the definition of G^π and obtain the fixed-point equation for G^* :

$$G^*(s,a) = E[c|s,a] - \frac{\gamma}{\beta} E[\log \sum_a \rho(a|s) e^{-\beta G^*(s,a)}] = \mathbf{B}^*[G^*]$$

G-Learning algorithm

G-Learning : an off-policy TD algorithm

Analogous to the Q-learning algorithm, the G-learning algorithm is :

$$G(s_t, a_t) \leftarrow (1-\alpha_t)G(s_t, a_t)+\alpha_t\left(c_t-\frac{\gamma}{\beta}\log\left(\sum_{a'}\pi(a'|s_{t+1})Q(s_{t+1}, a')\right)\right) \tag{7}$$

with some learning rate $0 \leq \alpha_t \leq 1$.

Outline

The role of the prior

The prior policy can encode any prior knowledge that we have about the domain. In the examples, the authors only use uniform prior.

This is a kind of regularization (cf. a Ridge-regularization is a $\mathbf{N(0,1)}$ prior) and we can enhance convergence by avoiding some exploration softly.

This soft-exploration formulation avoids using the min operator.

Outline

1. Introduction
   - Stakes and purpose
2. Learning in noisy environments with Q-Learning.
   - Notation and hypothesis
   - Q-Learning
   - Bias and commitment
   - The interplay of value bias and policy suboptimality
   - A dynamic optimism-uncertainty loop
3. G-Learning : Learning with soft updates
   - The Free-energy function G and G-Learning
   - The role of the prior
   - Convergence
4. Scheduling β
   - Scheduling β
5. Related work
   - Related work
6. Examples

Convergence

They use the following lemma :

Lemma

The operator $B^*[G]_{s,a}$ defined before verifies :

$$|\mathbf{B}^*[G_1]_{s,a} - \mathbf{B}^*[G_1]_{s,a}|_\infty < \gamma|G_1 - G_2|_\infty \tag{8}$$

They use also the fact that :

$$G_{t+1}(s_t, a_t) = (1 - \alpha_t)G_t(s_t, a_t) + \alpha_t(\mathbf{B}^*[G_t]_{(s,a)} + z_t(c_t, s_{t+1}))$$

where

$$z_t(c_t, s_{t+1}) = -\mathbf{B}^*[G_t]_{s_t,a_t} + c_t - \frac{\gamma}{\beta}\log\sum_a^{\prime} \rho(a'|s_{t+1})e^{-\beta G_t(s_{t+1},a')}$$

They note that $E[z_t] = 0$ which concludes the proof, given $|z_t < \infty|$.

Outline

Scheduling β

For a fixed β the algorithm converges with probability 1. A few comments :

- When $\beta = \infty$, the equations for G^* and F^* degenerate into the equations for Q^* and V^*, and G-learning becomes Q-learning. (better in early stage cf. noisy Q function)
- When $\beta = 0$, the update policy π is equal to the prior ρ. This case, denoted Q^ρ-learning, converges to Q^ρ. (better in late stage because better policy than the prior)

Solution : smooth-transition from Q^ρ-learning to Q-learning

Oracle Scheduling

At each step, there is always a β for which the update rule such an unbiased estimate remains unbiased. They do not prove it, but give an intuition. Let G be an unbiased estimate of G^*

- G still unbisaed with the following update

$$c_t + \gamma G(s_{t+1}, a^*)$$

  where is the real true right action : $a^* = \text{argmin}_{a'} G^*(s_{_{t+1},a'})$
- If we update with the G-learning algorithm and $\beta = 0$, there is a positive bias :

$$G_{t+1} \leftarrow c_t + \gamma \sum_{a'} \rho(a'|s_{t+1}) G(s_{t+1}, a')$$

- If $\beta = \infty$, there is a negative bias :

$$c_t + \gamma min_{a'} G(s_{t+1}, a')$$

Practical scheduling

β is updated as following :

$$\beta_t = kt$$

Simple but efficient according to authors (as efficient than an updating using the Bellman-error)

Outline

Related work

Other solutions to noisy environment :

- Double Q-learning : use of two estimators to update without bias.
- Advantage learning : learning $A(s, a) = Q(s, a) - V(s)$ seems to be faster than Q-learning in noisy environments
- Q-Learning with KL-divergence : very similar to G-learning.

Zoom on other KL-divergence techniques

There has been similar approaches using KL-divergence (penalty on information).

- Instead of using a prior ρ, they use the empirical distribution generated by the previous policy : *Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In AAAI, 2010.*
- Approach of [33] and [34] (references in the paper) : $\psi - learning$.

$$\psi(s_t, a_t) \leftarrow \psi(s_t, a_t) + \alpha_t(c_t + \gamma\bar{\psi}(s_{t+1}) - \psi(\bar{s_t}))i$$

with $\bar{\psi} = -\log \sum_a \rho(a|s)e^{-\psi(s,a)}$

6) Examples

Hyper-parameters :

- Learning rate :

$$\alpha_t = n_t(s_t, a_t)^{-\omega}$$

  where $n_t(s_t, a_t)$ is the number of times the pair (s_t, a_t) was visited. They choose $\omega = 0.8$.

- Discount factor :

$$\gamma = 0.95$$

Outline

Description of the environment

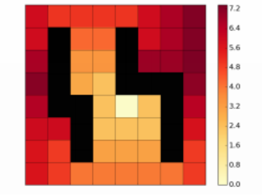


Figure: Gridworld domain. The agent can choose an adjacent square as the target to move to, and then may end up stochastically in a square adjacent to that target. The color scale indicates the optimal values V^* with a fixed cost of 1 per step.

For each case, the evolution over 250,000 algorithm iterations of the following three measures, averaged over N = 100 runs is computed.

Measures for the 1st example

Three measures :

- Empirical bias :

$$\frac{1}{nN}\sum_i\sum_s(V_{i,t}(s) - V^*(s)) \tag{9}$$

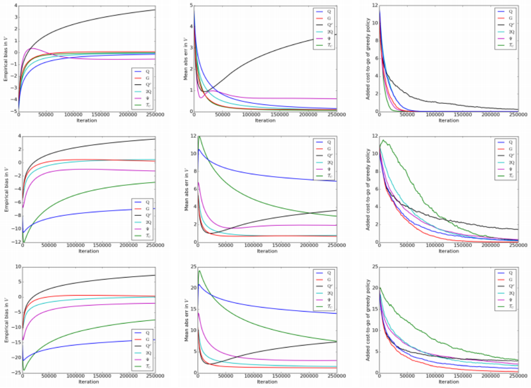
- Absolute error :

$$\frac{1}{nN}\sum_i\sum_s|V_{i,t}(s) - V^*(s)| \tag{10}$$

- Increase in cost-to-go, relative to the optimal policy :

$$\frac{1}{nN}\sum_i\sum_s(V^{\pi_{i,t}}(s) - V^*(s)) \tag{11}$$

Results



[]

Figure: Row 1 : cost is 1. Row 2 : cost is c $\sim N(1,2)$. Row 3: In each run, the domain is generated by drawing each E[c—s, a] uniformly over [1, 3]. The cost in each step is distributed as $N(E[c|s,a], 4^2)$.

Outline

Game

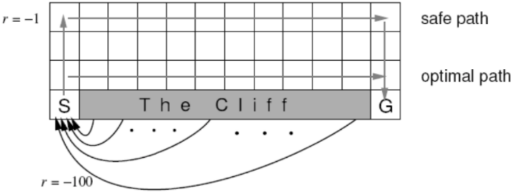


Figure: Cliff Waling environment

Cliff walking is a standard example in reinforcement learning , that demonstrates an advantage of on-policy algorithms such as SARSA and Expected-SARSA over off-policy learning approaches such as Q-learning.

Results

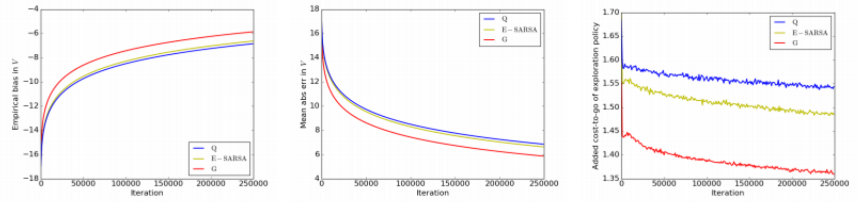


Figure: Cliff walking results

Conlusion

- Avoid the slow convergence in noisy environments caused by the bias generated (min operator)
- Explicit exploration
- Could be applied to other model-free setting, such as TD(λ)
- G-fit-learning ?

THANK YOU !