Converting a crafted (pavlovian) stochastic policy to a reward-maximizing actor-critic

June 24, 2015

We have developed a robot system that avoids collisions using a fixed stochastic policy with a given representation, where the representation includes one prediction that is adapted by continual learning. This fixed stochastic policy is then readily converted into a reward-maximizing policy by means of an actor-critic algorithm.

The initial fixed policy is crafted by formalizing the characteristics desired from the behavior in terms of externally visible quantities and in terms of a prediction on the robot that encapsulates one piece of useful empirical knowledge.

The robot is a Create (or Create2) from iRobot with a USB webcam and a Raspberry Pi (or Pi2) as a computer. We abstracted the robot's interface to the world. The inputs include a high-dimensional feature vector (from the webcam) and a bumper sensor. The outputs are restricted to two actions, forward (F) and turn (T). The prediction learning algorithm is tasked with predicting the onset of a bump when the robot drives forward at a timescale of one second, which is described in more detail below. The fixed policy is written using this prediction as a feature.

The prediction $p0$ anticipates when the robot bumper will be pressed when driving forward, presumably from a collision with a wall. *Specify the prediction as a GVF here.* The ideal value of $p0$ is 1 when the bumper will be pressed on the next timestep if the robot drives forward, and it is 0 when the robot is infinitely far away from a wall. The learning agent interacts with the world every 30 milliseconds (for a rate of $H = 1000/30$ times per second). This is the rate at which the control learning algorithm receives observations and select actions. The prediction $p0$ has a gamma of 0 when the bumper is pressed, and a gamma of 0.97 otherwise. Thus in the absence of any bumper presses, the prediction extends for $33.\bar{3}(= (1-\gamma)^{-1})$ timesteps in expectation (i.e. one second). We want the robot to have an equal probability selecting a drive action or a turn action when the ideal prediction has a value of $eq = 0.5$ (what is the timescale τ? $0.97^{\tau} = 0.5$).

Since we don't want to be limited to predictions at the rate of action-selection on the robot, we instead specify the desired number of switches per second when the prediction is at this equilibrium point. Define a new constant,

$$TT = H/\text{desired-number-of-switches-per-second},$$

and set the desired-number-of-switches-per-second to be 5. After some algebraic manipulation, we find that there is a constant $K_s = log_e((TT-1)(numberOfActions - 1)$ that governs switching at equilibrium in the absence of more information from the prediction.

There is one more constant used to scale the relative influence of the prediction on the action selection, we chose the number 4 (so p0=.75 means turning is e times more likely than going forward).

The robot can get into a situation where it incorrectly predicts that going forward will always lead to an imminent collision, causing it to turn continually. To break out of such configurations, we provide an additional feature which is a memory trace of how long it has been turning. This feature was set to a timescale of 3 seconds ($\gamma_m = 1 - 1/(3H)$) and scaled to a value between 0 and 1 with the following update,

$$memTurn \leftarrow (1-\gamma_m)\mathbb{I}(LastAction == T) + \gamma_m memTurn.$$

When the robot drives forward, memTurn will decay to zero, and when it turns continually, it will increase to one. This feature is used to prefer driving forward, and is scaled to comparable to the other terms.

We can write the preferences and the policy in an intuitive form.

$$
\begin{aligned}
pref(F) &= -4K_s(p0 - eq) + K_s\mathbb{I}(LastAction == F) + K_s memTurn \\
pref(T) &= K_s\mathbb{I}(LastAction == T) \\
\pi(a = F|x) &= \exp(\text{pref}(F))/(\sum_{a=F,T} \exp(\text{pref}(a))) \\
\pi(a = T|x) &= \exp(\text{pref}(T))/(\sum_{a=F,T} \exp(\text{pref}(a)))
\end{aligned}
$$

We rewrite the policy in an exponential-linear form by first introducing a representation x, that has four components indexed from 0.

- $(x_0)_t \equiv 1$: a bias unit.
- $(x_1)_t \equiv p0 \approx \mathbb{E}[G_t^{r,\gamma,\pi}]$ where for $k = 1, 2, \ldots$, we have $\pi(F|x_{t+k}) = 1, \gamma_{t+k} = 0$ on bump, 0.97 otherwise, and $r_{t+k} = \mathbb{I}(Bump_{t+k})$. This is a learned policy-contingent prediction (also known as a general value function) of the bumper for the option of driving forward with termination either on a bump or with a 3% probability per timestep.
- $(x_2)_t \equiv \mathbb{I}(A_{t-1} == F)$: the previous action was forward.
- $(x_3)_t \equiv \mathbb{I}(A_{t-1} == T)$: the previous action was to turn counterclockwise.
- $(x_4)_t \equiv memTurn$: the memory trace of turning at a 3 second timescale.

We rewrite the stochastic policy, with action selection probabilities given by exponentiated preferences for each action, where each preference is a linear function of the feature vector x, and the policy parameters u are initialized to the constants from above.

$$
\begin{aligned}
\text{pref}(F) &= u_0x_0 + u_1x_1 + u_2x_2 + u_4x_4 \\
\text{pref}(T) &= u_3x_3 \\
\pi(a = F|x) &= \exp(\text{pref}(F))/(\sum_{a=F,T} \exp(\text{pref}(a))) \\
\pi(a = T|x) &= \exp(\text{pref}(T))/(\sum_{a=F,T} \exp(\text{pref}(a))) \\
u_0 = 4eqK_s, \quad u_1 = -4K_s, \quad u_2 &= K_s, \quad u_3 = K_s, \quad u_4 = K_s
\end{aligned}
$$

Note that the parameters and features in the preferences are formed by a simple dot product. This leads to standard gradients computations, and it is simple to extend. Coupling parameters is also possible, but it complicates the gradients.

Now an average-reward actor-critic algorithm can be used to tune the parameters u so as to maximize a given reward signal over a long term. The average reward formulation is given in the Actor-Critic algorithms in practice paper, and it is listed below.

```
Average Reward Actor-Critic Algorithm
for each step with feature vector $x(S)$ do
    Choose $A$ according to $\pi_u(\cdot|S)$
    Take action $A$
    Observe $x(S')$, $R$
    $\delta \leftarrow R - \bar{R} + v^\top x(S') - v^\top x(S)$
    $\bar{R} \leftarrow \alpha_1 \delta$
    $e_v \leftarrow \lambda e_v + x(S)$
    $v \leftarrow v + \alpha_2 \delta e_v$
    $e_u \leftarrow \lambda e_u + \frac{\nabla_u \pi(A|x(S))}{\pi(A|x(S))}$
    $u \leftarrow u + \alpha_3 \delta e_u$
end for
```

The algorithm requires the gradients of the policy to be known, namely $\nabla_u \pi(a = k|x)/\pi(a = k|x)$.

Given the mathematical definition of the policy, the gradients can be found through the simple calculations shown below.

$$\frac{\nabla_u \pi}{\pi} = ((\partial_{u_0}\pi)/\pi, (\partial_{u_1}\pi)/\pi, (\partial_{u_2}\pi)/\pi)$$

To compute these, it is useful to precompute $\partial_{u_j} \exp(\text{pref}(i))$ for all i and j.

$$\begin{aligned}
\partial_{u_0} \exp(\text{pref}(F)) &= \partial_{u_0} \exp(u_0x_0 + u_1x_1 + u_2x_2 + u_4x_4) \\
&= \exp(u_0x_0 + u_1x_1 + u_2x_2 + u_4x_4)\partial_{u_0}(u_0x_0 + u_1x_1 + u_2x_2 + u_4x_4) \\
&= \exp(pref(F))x_0 \\
\partial_{u_1} \exp(\text{pref}(F)) &= \exp(\text{pref}(F))x_1 \\
\partial_{u_2} \exp(\text{pref}(F)) &= \exp(\text{pref}(F))x_2 \\
\partial_{u_3} \exp(\text{pref}(F)) &= 0 \\
\partial_{u_4} \exp(\text{pref}(F)) &= \exp(\text{pref}(F))x_4
\end{aligned}$$

$$\begin{aligned}
\partial_{u_0} \exp(\text{pref}(T)) &= 0 \\
\partial_{u_1} \exp(\text{pref}(T)) &= 0 \\
\partial_{u_2} \exp(\text{pref}(T)) &= 0 \\
\partial_{u_3} \exp(\text{pref}(T)) &= \exp(\text{pref}(T))x_3 \\
\partial_{u_4} \exp(\text{pref}(T)) &= 0
\end{aligned}$$

Let $W = \sum_i \exp(pref(i))$, and we compute its partial derivatives.

$$\begin{aligned}
\partial_{u_0} W &= \exp(\text{pref}(F))x_0 \\
\partial_{u_1} W &= \exp(\text{pref}(F))x_1 \\
\partial_{u_2} W &= \exp(\text{pref}(F))x_2 \\
\partial_{u_3} W &= \exp(\text{pref}(T))x_3 \\
\partial_{u_4} W &= \exp(\text{pref}(F))x_4
\end{aligned}$$

Now we can write the relevant gradients directly, recalling that $\pi(a|x) = \exp(\text{pref}(a))/W$, and also recalling that $\partial(f/g) = g^{-2}((\partial f)g - f(\partial g))$.

$$
\begin{aligned}
\frac{\partial_{u_0}\pi(F|x)}{\pi(F|x)} &= \frac{W}{\exp(\text{pref}(F))}\partial_{u_0}\frac{\exp(\text{pref}(F))}{W} \\
&= \frac{W}{\exp(\text{pref}(F))}\frac{1}{W^2}((\partial_{u_0}\exp(\text{pref}(F)))W - \exp(\text{pref}(F))\partial_{u_0}W) \\
&= \frac{W}{\exp(\text{pref}(F))}\frac{1}{W^2}(\exp(\text{pref}(F))x_0W - \exp(\text{pref}(F))\exp(\text{pref}(F))x_0) \\
&= \frac{1}{1}\frac{1}{W}(x_0W - \exp(\text{pref}(F))x_0) \\
&= x_0(1 - \frac{\exp(\text{pref}(F))}{W}) \\
&= x_0(1 - \pi(F|x)) \\
&= x_0\pi(T|x)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial_{u_1}\pi(F|x)}{\pi(F|x)} &= \frac{W}{\exp(\text{pref}(F))}\partial_{u_1}\frac{\exp(\text{pref}(F))}{W} \\
&= x_1\pi(T|x)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial_{u_2}\pi(F|x)}{\pi(F|x)} &= \frac{W}{\exp(\text{pref}(F))}\partial_{u_2}\frac{\exp(\text{pref}(F))}{W} \\
&= x_2\pi(T|x)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial_{u_3}\pi(F|x)}{\pi(F|x)} &= \frac{W}{\exp(\text{pref}(F))}\partial_{u_3}\frac{\exp(\text{pref}(F))}{W} \\
&= \frac{W}{\exp(\text{pref}(F))}\frac{1}{W^2}((\partial_{u_3}\exp(\text{pref}(F)))W - \exp(\text{pref}(F))\partial_{u_3}W) \\
&= \frac{1}{\exp(\text{pref}(F))}\frac{1}{W}(0 - \exp(\text{pref}(F))\exp(\text{pref}(T))x_3) \\
&= -\frac{\exp(\text{pref}(T))}{W}x_3 \\
&= -x_3\pi(T|x)
\end{aligned}
$$

$$
\frac{\partial_{u_4}\pi(F|x)}{\pi(F|x)} = x_4\pi(T|x)
$$

By noting the regularity in the derivations, we can quickly find the remaining gradients.

$$
\frac{\partial_{u_0}\pi(T|x)}{\pi(T|x)} = -x_0\pi(F|x)
$$

$$\frac{\partial_{u_1}\pi(T|x)}{\pi(T|x)} = -x_1\pi(F|x)$$

$$\frac{\partial_{u_2}\pi(T|x)}{\pi(T|x)} = -x_2\pi(F|x)$$

$$\frac{\partial_{u_3}\pi(T|x)}{\pi(T|x)} = x_3\pi(F|x)$$

$$\frac{\partial_{u_4}\pi(T|x)}{\pi(T|x)} = -x_4\pi(F|x)$$