Actor Critic Agent

Marco Marini

June 22, 2020

Abstract

Description of Actor Critic Agent and definitions of performance indicators

Contents

1 Actor Critic Agent

The Policy Actor Critic Agent is composed by an artificial neural network (ANN) that generates the differential state value of critic component and the policy parameters for action selection of each actor (the action dimensions) from the state input signals. At each iteration with the environment, the agent computes the average reward r_π and the differential TD error δ_t

$$\delta_t = r_t - r_\pi + v_\pi(s_{t+1}) - v_\pi(s_t) \tag{1}$$

The differential TD error indicate how different is the differential state value determined by the current reward from the estimated differential state value.

If the error is positive the agent changes the policy reinforcing the current strategy. It increments the probabilities of selected actions and at the same time reduces the probabilities of the other actions.

If the error is negative the agent changes the policy weakening the current strategy. It reduces the probabilities of selected actions and increments the probabilities of the other actions.

The magnitude of changes is proportional to the TD Error by a α_a parameter that allows to control the convergence of the algorithm.

The TD error is used to determine the updated output values and train the AAN with these expected outputs.

1.1 Discrete Policy Actor

The policy actor computes the probabilities $\pi_a(s)$ of choose action a at status s. The function is the softmax of the actions preferences $h_a(s)$ given by the ANN

$$\pi(a,s) = \frac{e^{h_a(s)}}{\sum_k e^{h_k(s)}} \tag{2}$$

simplifying the notation with

$$\begin{aligned} \pi(a,s) &= \pi_a \\ h_a(s) &= h_a \end{aligned}$$

To train the ANN the agent computes the updated actor preferences at each step t

$$h^*_a(s_t) = h_a(s_t) + \alpha_a \delta_{h_a}(t) \tag{3}$$

where $\delta_{h_a}(t)$ is the TD error back-propagated to the preference h_a

$$\delta_{h_a}(t) = \delta(t) \nabla \ln \pi_a$$

The policy gradient function is

$$\begin{aligned} \nabla \ln \pi_a &= \frac{1}{\pi_a} \frac{\partial}{\partial h_a} \pi_a \\ &= \frac{1}{\pi_a (\sum_k e^{h_k})^2} \left[e^{h_a} \nabla h_a - e^{h_a} \nabla \sum_k e^{h_k} \right] \\ &= \frac{1}{\sum_k e^{h_k}} \left[\nabla h_a - \sum_k \nabla e^{h_k} \right] \\ &= \frac{1}{\sum_k e^{h_k}} \left[\nabla h_a - \sum_k e^{h_k} \nabla h_k \right] \end{aligned}$$

Let be

$$\begin{aligned} A_i(a) = 1, &\Rightarrow i = a \\ A_i(a) = 0 &\Rightarrow i \neq a \end{aligned}$$

then

$$\begin{aligned} \nabla \ln \pi_a &= \frac{1}{\sum_k e^{h_k}} \sum_i \left[A_i(a) - e^{h_i} \right] \nabla h_i \\ &= \sum_i \left[\frac{A_i(a)}{\sum_k e^{h_k}} - \pi_i \right] \nabla h_i \end{aligned}$$

Finally

$$\delta_{h_a}(t) = \delta(s_t) \sum_i \left[\frac{A_i(a_t)}{\sum_k e^{h_k}} - \pi_i \right] \tag{4}$$

1.2 Gaussian policy actor

The Gaussian policy actor computes the probabilities $\pi(a, s)$ of choose a continuous action a at status s as a normal distributed function of two parameters $\mu(s)$ and $\sigma(s)$.

We change the notation to avoid ambiguity between the constant $\pi = 3.14\ldots$ and the policy $\pi(a, s)$:

$$\begin{aligned}\pi(a, s) &= p(a, s) \\ &= \frac{1}{\sigma(s)\sqrt{2\pi}} e^{-\frac{(a-\mu(s))^2}{\sigma(s)^2}}\end{aligned} \tag{5}$$

$$\sigma(s) = e^{h_\sigma(s)} \tag{6}$$

Futhermore we change the notation to simplifying the equation:

$$\begin{aligned}p(a, s) &= p \\ \mu(s) &= \mu \\ \sigma(s) &= \sigma \\ h_\sigma(s) &= h_\sigma\end{aligned}$$

to infere the gradient of logarithm of p

$$\nabla \ln p = \left(\frac{\partial}{\partial \mu} + \frac{\partial}{\partial h_\sigma}\right) \ln p$$

the partial derivative by μ is

$$\begin{aligned}
\frac{\partial}{\partial \mu} \ln p &= \frac{1}{p}\frac{\partial p}{\partial \mu} \\
&= \frac{1}{p\sigma\sqrt{2\pi}} e^{\frac{-(a-\mu)^2}{\sigma^2}} \frac{\partial}{\partial \mu}\left[-\frac{(a-\mu)^2}{\sigma^2}\right] \\
&= -\frac{1}{\sigma^2}[2(a-\mu)(-1))] \\
&= \frac{2}{\sigma^2}(a-\mu) \\
\frac{\partial}{\partial h_\sigma} \ln p &= \frac{1}{p}\frac{\partial p}{\partial \sigma}\frac{\partial \sigma}{\partial h_\sigma} \\
\frac{\partial p}{\partial \sigma} &= \frac{1}{\sigma^2\sqrt{2\pi}}\left\{\sigma\frac{\partial}{\partial \sigma}\left[e^{-\frac{(a-\mu)^2}{\sigma^2}}\right] - e^{-\frac{(a-\mu)^2}{\sigma^2}}\right\} \\
&= \frac{p}{\sigma}\left\{\sigma\frac{\partial}{\partial \sigma}\left[-(a-\mu)^2\sigma^{-2}\right] - 1\right\} \\
&= \frac{p}{\sigma}\left[-\sigma(a-\mu)^2(-2\sigma^{-3}) - 1)\right] \\
&= \frac{p}{\sigma}\left[2(a-\mu)^2\sigma^{-2} - 1\right] \\
\frac{\partial \sigma}{\partial h_\sigma} &= \sigma\frac{\partial}{\partial h_\sigma}\ln p \\
&= \frac{1}{p}\frac{p}{\sigma}\left[2(a-\mu)^2\sigma^{-2} - 1\right]\sigma \\
&= 2(a-\mu)^2\sigma^{-2} - 1
\end{aligned}$$

The backward propagated TD errors to the output network layer are

$$\delta_\mu(t) = \frac{2}{\sigma^2}(a-\mu)\delta(t) \tag{7}$$

$$\delta_{h_\sigma}(t) = \left[2\frac{(a-\mu)^2}{\sigma^2} - 1\right]\delta(t) \tag{8}$$

The updated actor parameters are:

$$\mu^*(s_t) = \mu(s_t) + \alpha_\mu \delta_\mu(t) \tag{9}$$

$$h_\sigma^*(s_t) = h_\sigma(s_t) + \alpha_{h_\sigma}\delta_{h_\sigma}(t) \tag{10}$$

1.3 Learning process

Initially the agent do not have any experience and the differential state value generated by the ANN starts from a random initial estimation. The same is for the behavior generated by the actors where the preferences are random initialized.

Since the very first interactions the agent generates wrong differential state value estimation with high value of TD Error. Consequently the agent easily changes the policy behavior, trying different policies with the exploration of both the policy and state space. It tries to improve the average reward, the value state estimation at the same time selecting better policies.

Different factors influence the learning process. External agent factors like the number of states, actions and policies not yet explored tent to generate high error rate. The error increases as the agent experiences new environment states or try new policies, but as the agent faces states already visited it can exploit the past experiences reducing the error and improving the state value estimation and the policy.

Instead the agent internal factor, such as the ANN approximation, generates errors that can be reduced only to a low-bound limit value.

The ANN learning paramters can influence both in the quality of approximated functions and in the speed of adaptation.

High value of step-size parameter η increases the correction of weights increasing consequently the speed of adaptation, but can easily generate corrective value too big bringing the network to not converging behaviors. On the other hand small value of step-size parameter weakens the correction of weights decreasing the speed of adaptation.

Also the actor parameters α_a, which modulate the policy changes from the critic, influence the improvement capacity. High value of α_a determine strong policy changes even in the face of small errors, making it difficult a fine correction, the other way around very small value of α_a bring very little variations on policy without improvement.

If the actor parameters that govern the policy have a limited range of values we may choose an α_a so that the average of corrections is ε_a a fraction of the limited range.

$$\alpha_a = \frac{\varepsilon_a}{\sqrt{\frac{1}{n}\sum_i^n \delta_{h_a}^2(i)}}$$

$$\delta_{h_a}(i) = \frac{h_a^*(i) - h_a(i)}{\alpha'_a}$$

$$\alpha_a = \frac{\varepsilon_a}{\sqrt{\frac{1}{n}\sum_i^n (h_a^*(i) - h_a(i))^2}}\alpha'_a \tag{11}$$

1.4 Performance

The average reward is the main indicators of learning quality, since the agent interacts with the environment the average reward grows up.

Because the process is adaptive the average reward can vary a lot during the interaction with the environment therefore it is necessary to evaluate the trend of the indicator. The logarithmic regression can be used to evaluate the performance

$$r_\pi(t) = \ln(mt + q) \tag{12}$$

An increasing trend indicates the agent is selecting better policies and is getting higher and higher rewards. A decreasing trend indicates the agent is not selecting good policies, this may be caused by low α_a or η parameters.

Another indicator of performance is the squared TD error, as for the average reward it is necessary to evaluated the trend. The error has not negative values that should approach zero when the agent select the local best policy. The exponential regression can be used to evaluate the performance

$$\delta^2(t) = e^{(mt+q)} \tag{13}$$

A decreasing trend indicates the estimation of value states is getting better. An increasing trend indicates the estimation of value states is getting worse it may be caused by the ANN which cannot correct the estimation due to η parameter too high.

Both the critic and actors approximate the value function and policy function with neural network. To monitor the learning activity of ANN we compute the MSE of estimated functions.

The critic computes the updated value of current state by applying the the bootstrap equation:

$$v^*(s_t) = v(s_{t+1}) + r_t - r_\pi$$

The square error of critic output is

$$\delta^2(t) = [v^*(s_t) - v(s_{t+1})]^2$$

The square error of actors outputs are

$$J_i = \sum_i (h_i^*(s_t) - h_i(s_t))^2$$

where i is index of the actor output.

Every time the network is trained the label outputs are fed to correct the weights of ANN. The resulting network has therefore a MSE that should be less than the original.

$$\begin{aligned}\delta'^2(t) &= [v^*(s_t) - v'(s_{t+1})]^2 \\ J'_i(t) &= \sum_i (h_i^*(s_t) - h'_i(s_t))^2\end{aligned}$$

The ratio between the total MSE after and before the training activity indicates the quality of such activity.

$$
\begin{aligned}
J(t) &= \delta^2(t) + \sum_i J_i(t) \\
J'(t) &= \delta'^2(t) + \sum_i J'_i(t) \\
K(t) &= \frac{J'(t)}{J(t)}
\end{aligned}
\tag{14}
$$

A ratio $K(t) \geq 1$ means the error after training gets worst due a step-size parameter α too high. A ratio $K(t) = 1$ means no change on error and therefore no improvement. This can be affected by a local minimum reached or a step-size parameter too low with very poor capacity of learning. A ratio $K(t) < 1$ means an improvement of neural network due to correct step-size parameter. A ratio $K(t) = 0$ means a perfect fit of neural network.

We can classify the steps in three class:

C_0 The steps that created a bad approximation with an increased of MSE ($J > \varepsilon \cup K > 1$)

C_1 The steps that create a trivial approximation with a small reduction of MSE ($J > \varepsilon \cup K_0 \leq K \leq 1$ with $K_0 = 0.9$)

C_2 The remaining steps that have a small MSE or that have reduced significantly the MSE

The ideal distribution should have

$$
\begin{aligned}
C_0 &= 0 \\
C_1 &= 0 \\
C_2 &= 1
\end{aligned}
\tag{15}
$$

A step parameter η too high generates over correction increasing C_0 and reducing C_2.

An η parameter too small generates under corrections increasing C_1 and reducing C_2.

Because it is difficult to determine the effects of step parameter changes on the result MSE, an empirical way to reduce or increment the parameter is applied, for example increasing or reducing by exponential factors (…, $\times 0.01$, $\times 0.03$, $\times 0.1$, $\times 0.3$, $\times 3$, $\times 10$, $\times 30$, $\times 100$, …).

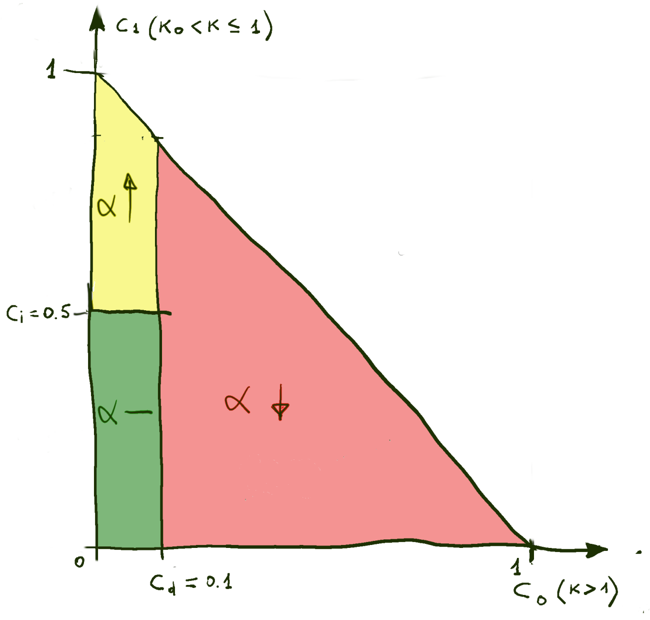

Figure 1: Step classification