Actor-Critic Methods

Sai Krishna, Pratik Mandlecha, Naresh Manwani and Praveen Paruchuri

In collaboration with CognitiveScale

krishna.munnangi@research.iiit.ac.in, pratik.mandlecha@students.iiit.ac.in,
{naresh.manwani, praveen.p}@iiit.ac.in

February 11, 2019

Recap Actor-Critic algorithm

Repeat until convergence:

1. take action $a \sim \pi_\theta(a|s)$, get (s, a, s', r)
2. update $\hat{V}^\pi_\phi$ using target $r(s, a) + \gamma \hat{V}^\pi_\phi(s')$
3. $A^\pi(s, a) = r(s, a) + \gamma \hat{V}^\pi_\phi(s') - \hat{V}^\pi_\phi(s)$
4. $\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta(a|s) A^\pi(s, a)$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Implementation Details

$$\text{take action } a \sim \pi_\theta(a|s)$$

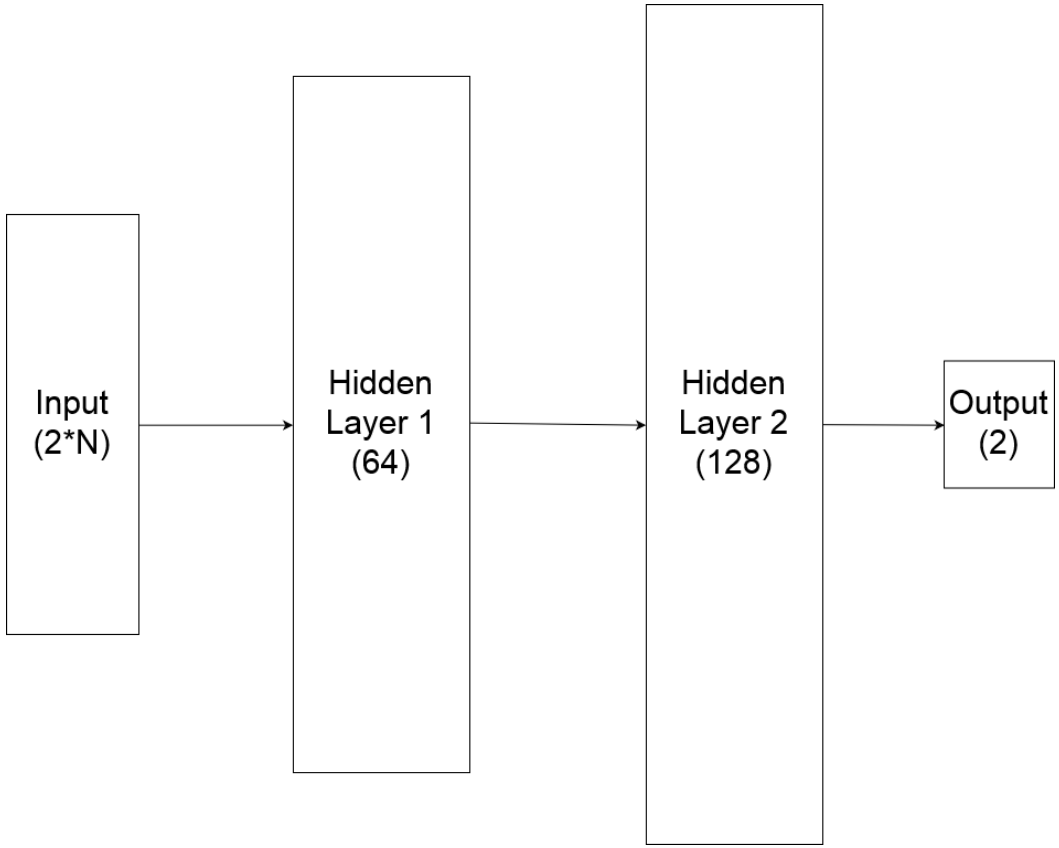
CTO problem is continuous observation-action space problem. To choose action a from the policy $\pi_\theta(a|s)$, we assume the policy is defined using Gaussian Distribution,

$$\mathcal{N}(\mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right)$$

The mean μ and the covariance matrix Σ is computed by the actor network.
The idea is to learn appropriate mean and covariance matrix values during training, so that given a new state as input, the distribution peaks at one point - optimal action a^* for that state.

Implementation Details - Actor

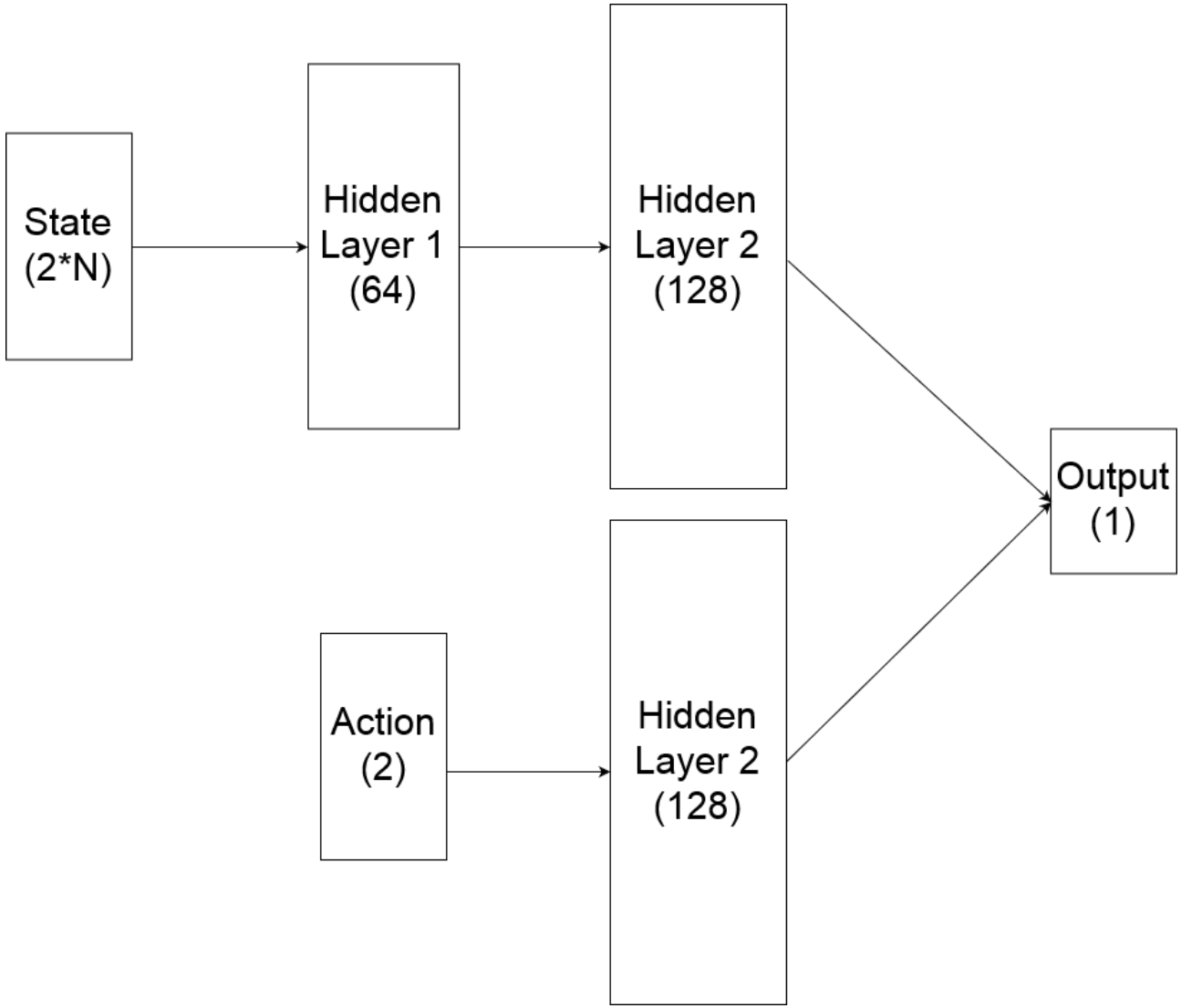
The actor for the CTO problem is implemented by following neural network.



For the purpose of experimentation, Σ is chosen to be $\mathcal{I}_{2\times 2}$.

Implementation Details - Critic

The critic for the CTO problem is implemented by following neural network.



Experimentation

Formally the CTO problem has been defined as follows: Given,

S : a two-dimensional, bounded, enclosed spatial region

O : a team of m observer robots with observation sensors of limited range, denoted by $sensor\_range$

X : a set of n targets

Let $A(t)$ be a $m \times n$ matrix where,

$$a_{ij}(t) = \begin{cases} 1 & \text{if observer } o_i \text{ is monitoring target } x_j \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

The logical OR operator over a vector H is defined as,

$$\bigvee_{i=1}^{k} h_i = \begin{cases} 1 & \text{if there exists an } i \text{ such that } h_i = 1 \\ 0 & \text{otherwise} \end{cases}$$

Experimentation (contd.)

The goal of the problem is to maximize

$$\sum_{t=0}^{T}\sum_{j=1}^{n}\bigvee_{i=1}^{m} a_{ij}(t), \text{ given } \bigcup_{o_i \in O} sensor\_range(o_i) \ll S$$

- For the purpose of experimentation, we define S as a 150×150 grid with no obstacles.
- We assume the observers know beforehand the possible list of targets. The total number of targets in the grid is fixed to 10. Each target moves randomly in a straight line, changing its trajectory once every 100 time-steps.
- The algorithm is tested with only one observer whose sensor range is 15 units. The targets and observers operate at same speed.
- The update rate is set to 10 i.e., observers make decisions once every 10 time-steps. The state representation has N rows of (x, y) coordinates where (x, y) is the position of target i if it is in the sensor range of observer and $(0, 0)$ otherwise. Hence the input size is $2N$ when flattened.

Experimentation (contd.)

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY HYDERABAD

- Ideally each observer would have a separate actor-critic network and different policy. Each observer would try to improve its policy based on its own observations.
- We trained the actor-critic network for the single observer, for over 1,50,000 decision making instances.
- Yet the agent's performance hasn't been improved, with agent seemingly behave randomly instead of exploiting targets within the sensor range.
- With many hyper parameters to fine tune, it is hard to differentiate whether there is an error in the implementation or the network needs fine tuning or some unknown factor influence needs to be rectified.
- So instead of directly using neural networks, we plan to scale down and use linear models where actor and critic will be modeled as simple linear transformations of the state while the gradients, updates etc., remain same.

Plan of Action

- Similar to the line of work in Kimura et al., we plan to use the actor parameterized by

$$\theta = \begin{bmatrix} \theta_{1,1} & \theta_{1,2} & \theta_{1,3} & \dots & \theta_{1,2N} & \theta_{1,2N+1} \\ \theta_{2,1} & \theta_{2,2} & \theta_{2,3} & \dots & \theta_{2,2N} & \theta_{2,2N+1} \end{bmatrix}$$

  and the critic parameterized by

$$\omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \dots & \omega_{2N} \end{bmatrix}$$

- While $\omega.state^T$ gives the value of a state, $\theta.state^T$ provides mean and variance of the gaussian policy $\pi_\theta(a|s) \approx \mathcal{N}(\mu, \Sigma)$.

$$\text{Mean } \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \text{ where } \mu_1 = \frac{150}{1 + exp(-\sum_{k=1}^{2N} s_k \theta_{1,k})} \text{ and}$$

$$\mu_2 = \frac{150}{1 + exp(-\sum_{k=1}^{2N} s_k \theta_{2,k})}$$

Plan of Action (contd.)

$$\text{Covariance } \Sigma = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}, \text{ where } \sigma_1 = \frac{1}{1 + exp(-\theta_{1,2N+1})} \text{ and}$$

$$\sigma_2 = \frac{1}{1 + exp(-\theta_{2,2N+1})}$$

- An action is then sampled from the resulting bivariate gaussian distribution, which will be the destination of the observer.
- Steps 1 - 5 are repeated until the convergence of θ and ω.
- Ideally, while training the variance values must shrink resulting in a peak at μ which will the optimal action identified by actor.