Policy search with Softmax

February 24, 2018

Abstract

A short guide on how the agent for the policy search method works.

1 Parametrization

First of all, a policy is defined:

$$p(a = \hat{a}|s) = \pi_\Theta(a = \hat{a}, s) = \frac{e^{s^T w_{\hat{a}}}}{\sum_{k=1}^{A} e^{s^T w_k}} \cdot f(s) \tag{1}$$

where

$$f(s) = \begin{cases} 1 & \text{if s allowes } \hat{a} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

A is the number of possible actions $A = n_a^{(k+1)}$ ($n_a = 3$ number of actions per state, $k = 3$ number of stores). The parameter matrix Θ is assembled the following way:

$$\Theta = \big(w_1|w_2|...|w_A\big) \in \mathbb{R}^{(k+2)\times A} \tag{3}$$

and $w_i \in \mathbb{R}^{(k+1)+1}$ (including a bias).

2 Gradient ascent

In order to do the gradient ascent we have to compute the gradient:

$$\nabla_{w_i} J(\Theta) = \nabla_{w_i} ln(\pi_\Theta(a_t = a_j, s_t))D_t = \begin{cases} (1 - \sigma_i(s_t)) \cdot s_t \cdot D_t & \text{if } i = j \\ -\sigma_i(s_t) \cdot s_t \cdot D_t & \text{if } i \neq j \end{cases} \tag{4}$$

where σ is the softmax function. Now every vector w_i of Θ can be updated.

3 Questions/ to do

3.1 in the Theory

- f(s) drops out in the differentiation (because of ln), therefore all weights, even for unfeasible actions have to be updated? Does this make sense?
- define σ better. - if we want to increase the actions per warehouse space, the action space explodes - Problem if alpha is too big, w_i gets changed so much that the exponential goes to infinity =¿ numerical problems

3.2 in the Code:

- currently the available actions are hardcoded