UNIVERSITÄT ZU LÜBECK
INSTITUTE FOR ROBOTICS
AND COGNITIVE SYSTEMS

Course: Reinforcement Learning (SS2020)
Prof. Dr. Elmar Rueckert & Honghu Xue, M.Sc.
Assignment Report Nr. V
Date: 15. Juli 2020

Team Nr. 1
Matthias Klatt
Abby Kandathil Abraham
Suneesh Omanakuttan Sumesh Nivas
Sameed Quais
Lakshith Nagarur Lakshminarayana Reddy
Contact: matthias.klatt@student.uni-luebeck.de

Task I, q.1:

With policy-gradient (**PG**) approaches we are able to learn deterministic or stochastic policies. Action-value methods have no natural way of finding stochastic optimal policies, whereas policy approximation methods can do [1].
Also with **PG** approaches we have a guarantee to converge on a local maximum if we are lucky this local maximum is also the global maximum but in general that is not the case.
With **PG** approaches which are using so called *soft-max* distribution in action preferences we can approximate a deterministic policy, value function based approaches which are using an ϵ greedy approach to select actions have always an ϵ probability of selecting a random action [1].
PG approaches are on policy approaches and because of that as all on-policy approaches it has in general a bad sample efficiency and due to that the convergence to a (local) optimum policy can take a long time. To make it short our gradients are very noisy which leads to a high variance and therefore different methods are developed to deal with this variance.

Task I, q.2:

In **PG** we have a policy parameter vector θ, the optimal policy with the optimal policy parameter θ^* is for a finite horizon problem defined as

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{\tau \sim p_\theta(\tau)}[\sum_t \mathcal{R}(s_t, a_t)], \tag{1}$$

with the τ the state-action values our agent takes from the beginning until the end ($\tau = [s_1, a_1, \ldots, s_T, a_T]$) and the probability of taking this state-action pairs (or trajectory) $p_\theta(\tau)$. In the following $\mathcal{R}(\tau)$ is used for the reward of the trajectory ($\mathcal{R}(\tau) = \sum_t \mathcal{R}(s_t, a_t)$). To measure how good the actual policy parameter are a scalar feedback is needed. This scalar feedback is needs to depends on the policy parameter and is defined as

$$\mathcal{J}(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)}[\mathcal{R}(\tau)], \tag{2}$$

which is in most cases difficult or impossible to calculate , for example if the transition probability is unknown. The performance measurement will be approximate the function with samples. If N samples are used the performance measurement (equation 2) is approximated as

$$\mathcal{J}(\theta) \approx \frac{1}{N} \sum_i \sum_t \mathcal{R}(s_{i,t}, a_{i,t}). \tag{3}$$

In the case of reward signals the performance measurement feedback needs to be maximizes (because we want to maximize the reward) and in case of costs the signal needs to be minimized. So gradient *ascent* will be used in the reward signal case and gradient *descent* in the cost function case.
For a continuous state space the performance measurement will change to

$$\mathcal{J}(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau, \tag{4}$$

this equation is in general full of unknowns, e.g. the transition probability because of that we will use an important trick in deeplearning and RL namely that the partial derivative of a function $f(x)$ is equal to $f(x)$-

times the partial derivative of the logarithm of $f(x)$, with that follows

$$\begin{aligned} f(x)\nabla_\theta \log f(x) &= f(x)\frac{\nabla_\theta f(x)}{f(x)} \\ &= \nabla_\theta f(x), \end{aligned}$$

if we replace $f(x)$ with $\pi_\theta(\tau)$ follows

$$\pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau) = \nabla_\theta \pi_\theta(\tau).$$

Therefore our performance measurement gradient becomes

$$\begin{aligned} \nabla_\theta \mathcal{J}(\theta) &= \int \nabla_\theta \pi_\theta(\tau)\mathcal{R}(\tau)d\tau \\ &= \int \pi_\theta \nabla_\theta \log \pi_\theta(\tau)\mathcal{R}(\tau)d\tau \\ &= \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau)\mathcal{R}(\tau)] \\ &= \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(s_t))(\sum_{t=1}^{T} \mathcal{R}(s_t, a_t))]. \end{aligned} \tag{5}$$

Because $\mathcal{J}(\theta)$ is is often estimated via sampling the gradient will change in this case to

$$\nabla_\theta \approx \frac{1}{N}\sum_{i=1}^{N}(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_{i,t}|s_{i,t}))(\sum_{t=1}^{T} \mathcal{R}(s_{i,t}, a_{i,t})), \tag{6}$$

and with that the policy parameter in REINFORCE are updated like

$$\theta \leftarrow \theta + \alpha\nabla_\theta \mathcal{J}(\theta), \tag{7}$$

where alpha is a learning factor between 0 and 1.

Task I, q.3:

The idea of adding a baseline to the REINFORCE algorithm is to reduce the variance. The baseline is subtracted from the optimization problem which is valid as long as the baseline b doesn't vary with a[1]. Subtracting a baseline is so called 'unbiased' in expectation if it isn't related to the θ parameter directly so if it is for example just a constant term it will not change the expectation and so it is still unbiased.
With the help of a baseline we can re-calibrate the rewards and with that makes much better action for a state more likely by increasing his probability to be chosen for that state.

A possible baseline would be the average baseline which is defined as

$$b = \frac{1}{N}\sum_{i=1}^{N} \mathcal{R}(\tau), \tag{8}$$

with this baseline we make something that gives a better reward than the average more likely.

Task I, q.4:

The idea behind *causality* is that future actions should not change past decisions and present actions should only impact the future.
Therefore, the performance measurement gradient $\nabla_\theta \mathcal{J}(\theta)$ (or object function) is changed to

$$\nabla_\theta \approx \frac{1}{N}\sum_{i=1}^{N}(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a_{i,t}|s_{i,t}))(\sum_{t'=t}^{T}\mathcal{R}(s_{i,t'},a_{i,t'})), \tag{9}$$

So the second summation is changed compared to equation 6, it starts at the time step t instead of 1.
The reward summation will get decrease if t increase until we reached the last time step. With that decreasing they are less rewards the higher t get and because of that the variance is also decreased.
A2C is an on-policy approach as it used the same policy π to estimate the Value function $\mathcal{V}$ as for training the policy parameter θ.

Task I, q.5:

REINFORCE with baseline method learns both a policy and a state value function but its state value function is used only as baseline which means it is not used for bootstrapping. Secondly,REINFORCE with baseline is unbiased approach and will converge asymptotically to a local minimum but it tends to learn slowly and produces estimates of high variance. In order to implement online or continuing problem, in the case of policy gradient methods we use actor–critic methods with a bootstrapping critic [1].
The advantages of one-step actor-critic methods are [1]:

. Fully online

. Incremental

. Avoiding the complexities of eligibility traces

. Replaces the full return of REINFORCE with the one-step return and use a learned state-value function as the baseline as follows [1]:

$$\theta_{t+1} = \theta_t + \alpha\delta_t \frac{\nabla\pi(A_t|S_t,\theta_t)}{\pi(A_t|S_t,\theta_t)} \tag{10}$$

Here:

$$\delta_t = (R_{t+1} + \gamma\hat{v}(S_{t+1},w) - \hat{v}(S_t,w) \tag{11}$$

Update the policy parameters by the expression shown below:[2]

$$\theta \leftarrow \theta + \alpha_\theta Q_w(s,a)\nabla_\theta \ln \pi_\theta(a|s) \tag{12}$$

Task I, q.6:

$$\pi_\theta(a|s,\theta) = \frac{1}{\sigma(s,\theta)\sqrt{2\pi}}\exp(-\frac{(a-\mu(s,\theta))^2}{2\sigma(s,\theta)^2}) \tag{13}$$

The policy can be determined in the form of the normal probability density [1].
Here:
μ represents mean where $\mu : S \times \mathbb{R}^{d'} \rightarrow \mathbb{R}$
σ represents standard deviation where $\sigma : \mathbb{S} \times \mathbb{R}^{d'} \rightarrow \mathbb{R}^+$.

The policy parameters vector θ is divided into two parts, $\theta = [\theta_\mu, \theta_\sigma]^T$.
θ_μ to be used for the approximation of the mean and it can be approximated as a linear function.
θ_σ for the approximation of the standard deviation.It must always be positive and is better approximated as the exponential of a linear function.
The derivation of Ex 13.4 from RL book is shown in figure 1:
TODO: write down how pi(a,s,theta) is defined (math) and explain all terms

TODO: how are the mean and the variance defined using theta in practice

TODO: show the complete derivation of exercise 13.4 on page 336 from book RL

Task II, q.1:

The advantage function ($\mathcal{A}_\pi(s_t, a_t)$) for a policy π of a state s_t and an action a_t is defined as a subtraction of the state value function ($V_\pi(s_t)$) from the state-action function ($\mathcal{Q}_\pi(s_t, a_t)$). The $\mathcal{Q}$ function is defined as

$$\mathcal{Q}_\pi(s_t, a_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_\theta}[\mathcal{R}(s_t, a_t)|s_t, a_t], \tag{14}$$

and the state-value function as

$$V_\pi(s_t) = \mathbb{E}_{a_t \sim \pi_\theta(a_t, s_t)}[\mathcal{Q}_\pi(s_t, a_t)], \tag{15}$$

with that follows for the advantage function

$$\mathcal{A}_\pi(s_t, a_t) = \mathcal{Q}_\pi(s_t, a_t) - V_\pi(s_t), \tag{16}$$

in practice are some approximation used, e.g. if we approximate the $\mathcal{Q}$ function (equation 14) as $\mathcal{R}(s_t, a_t) + V_\pi(s_{t+1})$

$$\mathcal{A}_\pi(s_t, a_t) \approx \mathcal{R}(s_t, a_t) + V_\pi(s_{t+1}) - V_\pi(s_t). \tag{17}$$

Also other approximations are possible and it always depend on the task what approximation is better.

Task II, q.2:

In figure 2 are the resulting training returns for our REINFORCE with causality reduction algorithm for the 'CartPoleBulletEnv-v1' environment. One training episode is done by each team member and as you can see because of the high variance the return is oscillation highly.

Task II, q.3:

Bonus Task, q.1

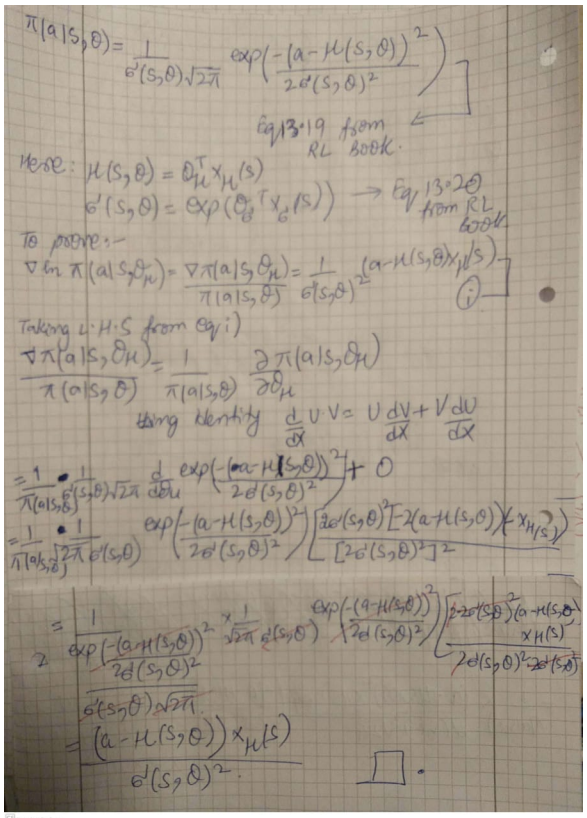
Generalized advantage estimation (or GAE) is a generalized estimator of the advantage function which allows us a trade-off between bias vs variance using a paramter λ, similar to TD(λ). The GAE advantage function is defined as

$$\hat{\mathcal{A}}_\pi = \sum_{t'=t}^{\infty} (\gamma\lambda)^{t'-t} \delta_{t'}, \tag{18}$$

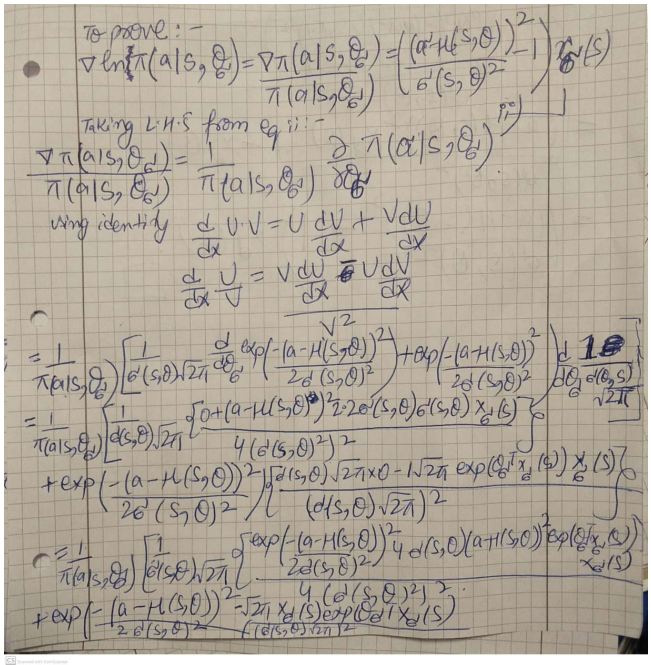
with

$$\delta_{t'} = \mathcal{R}(s_{t'}, a_{t'}) + \gamma \hat{V}_\pi(s_{t'+1}) - \hat{V}_\pi(s_{t'}),$$

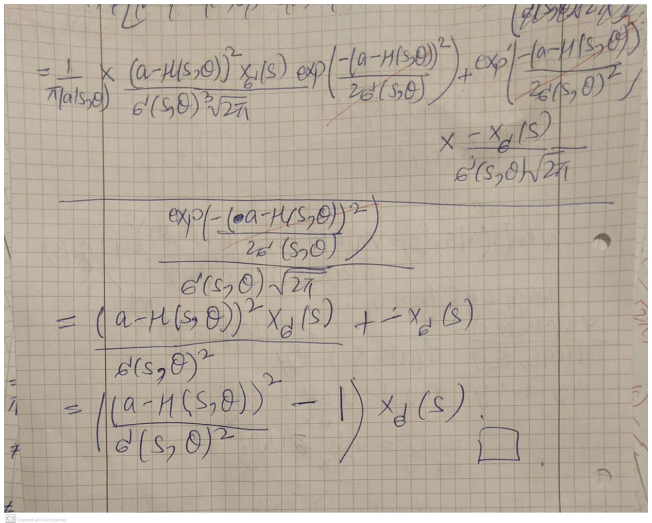
the temporal difference residual. λ can be chosen between 0 and 1 if λ is equal to one the variance is very large but their is no bias if we set λ equal to one the variance will be very small but the bias will be high.



(a) Ex13.4 part a



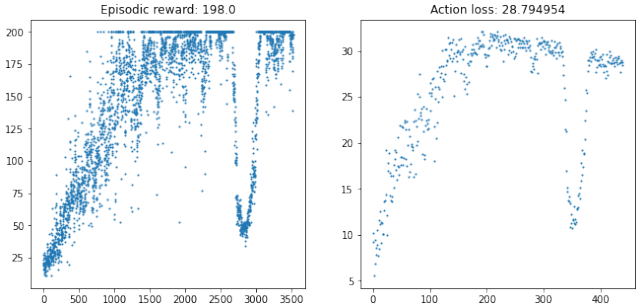
(b) Ex13.4 part b



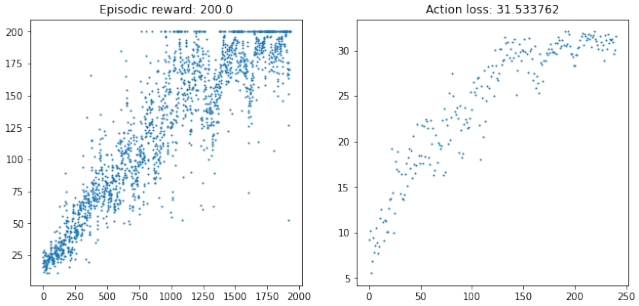
(c) Ex13.4 part b cont.

Figure 1 The derivative of exercise 13.4 from [1]

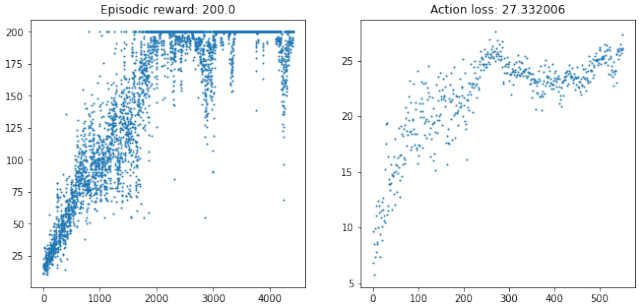
(a) *Matthias*



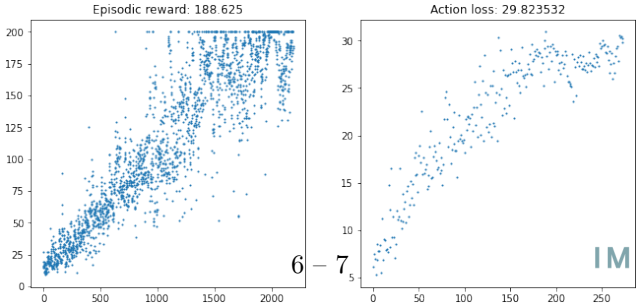
(b) *Abby*



(c) *Suneesh*



(d) *Sameed*



(e) *Lakshith*

Figure 2 The learning curves for the REINFORCE task one learning curve from each team member

Literatur

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: an Introduction*. Second edition.

[2] Lilian Weng. Policy gradient algorithms. Website, 2018. url: `https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html#actor-critic`; Retrieved on 01.07.2020.