Noname manuscript No.
(will be inserted by the editor)

Extensions for DDPG and analysis of its components

subtitle here

Maximilian Gehrke · Tabea Wilke · Yannik Frisch

Received: date / Accepted: date

Abstract TODO

Keywords DDPG · DQN · DPG

1 Introduction

Deep Deterministic Policy Gradients (DDPG) arises from Deterministic Policy Gradients (DPG) and Deep Q-Learning (DQN). In the following we describe the underlying algorithms DPG and DQN and which aspects DDPG uses of both of them.

1.1 Deterministic Policy Gradient (DPG)

1.2 Deep Q-Learning (DQN)

DQN is the combination of neural networks and q-learning. It works on a deterministic environment with the goal to achieve the optimal action-value function. This means finding the best action with respect to the rewards also in the future to a given state. In terms of a formula it is represented by

$$Q^*(s_t, a_t) = \max_\pi E\left[\sum_{t'=t}^{T} \gamma^{t'-t} r_{t'} | s_t = s, a_t = a, \pi\right]$$

F. Author
first address
Tel.: +123-45-678910
Fax: +123-45-678910
E-mail: fauthor@example.com

S. Author
second address

with λ as discount factor smaller but close to 1, so the agent takes also future reward into account. The rewards of the future will have impact on the result but the influence decreases with time. For estimating the action-value function a deep network is used. Furthermore, a replay buffer is used which will save samples of the environment. Therefore, it is possible to achieve a non correlated batch. There are different ways of estimating the expected q-values. Either with a target network with the same structure as the network for the action-value function or the normal network. If a target network is used, the target weights need to be updated after some training steps.
The loss is calculated through the mean-squared-loss of the expected q-value and the estimated q-value:

$$L_i(\theta_i) = E_{(s,a,r,s')}\left[\left(r + \gamma \max_{a'} Q(s^{'}, a^{'}|\theta_i^{'}) - Q(s,a|\theta_i)\right)^2\right]$$

DQN can only handle discrete and low-dimensional action spaces.

Algorithm 1 Deep Q-Learning (DQN)

Initialize: Replay buffer D with high capacity
Initialize: Neural network for action-value function Q with random weights θ
Initialize: Neural network for target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
  for episode 1 **to** M **do**
    reset environment to state s_1
    for $t = 1$ **to** T **do**
      if random $i \leq \epsilon$ **then**
        random action a_t
      else
        $a_t = \text{argmin}_a\, Q(s_t, a|\theta)$
      end if
      execute $a_t \rightarrow$ reward r_t and next state s_{t+1}
      save (s_t, a_t, r_t, s_{t+1}) in D
      sample minibatch (s_i, a_i, r_i, s_{i+1}) from D
      $q_i = \begin{cases} r_i & \textit{if episode terminates at step i+1} \\ r_i + \gamma \max_{a'} \hat{Q}(s_{i+1}, a_i|\theta^-) & \textit{else} \end{cases}$
      perform gradient descent on $(q_i - Q(s_i, a_i|\theta))^2_\theta$
      every C steps update $\hat{Q} = Q$
    end for
  end for

- DQN uses deep networks to estimate the action-value function
  - it can only handle discrete and low-dim action spaces
- discretizing the action space often suffers from the course of dimensionality
- PolicyGradientTheorem from continous space to discrete space presented in DPG paper
- naive extension of DPG with nns turns out to be unstable for challenging problems
- Deep DPG (DDPG): combination of DQN and DPG, where:

  - networks are trained off-policy with samples from a replay buffer to minimize the temporal correlations between samples
  - the networks are trained with target networks to give consistent targets during temporal difference backups
  - batch normalization is used
- DDPG is able to learn from low dim observations (torques etc.), aswell as from high dim observations in pixel space

Your text comes here. Separate text sections with

2 Extensions to the Algorithm

References

[Mnih et al.(2013)Mnih, Kavukcuoglu, Silver, Graves, Antonoglou, Wierstra, and Riedmiller] Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. arXiv preprint arXiv:13125602

[Mnih et al.(2015)Mnih, Kavukcuoglu, Silver, Rusu, Veness, Bellemare, Graves, Riedmiller, Fidjeland, Ostrovski et al.] Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, et al. (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529