DM887 Assignment2 Q1

Jiawei Zhao

19.03.2024

Algorithm 1 Least-Squares Temporal Differences (LSTD) Deep Q-Learning with Nonlinear Feature Extraction that maps a state to lower-dimensional latent embedding. While the action-value function should be a linear function of the output of the feature extractor.

Initialize a variational autoencoder $\mathbf{A}$ as the feature extraction network with initial weights θ_0 [θ_0 are drawn from a zero-mean Gaussian distribution]

Initialize the number of maximum episode $N_0 = 10$, $N_1 = 10$, and $N_2 = 30$ for each phase of training [It would be a wise choice to seperate the warm-up phase, the antoencoder improvement phase, and the LSTD improvement phase] [Initialization of training hyperparameters]

Initialize the number of maximum time step per episode $t = 5000$

Initialize the weights w_0 for linear approximation of the action-value function $y = f(phi(s))$ which will be used for LSTD

Initialize a replay memory buffer $\mathcal{D}$ with a capacity $N = t(N_0 + N_1 + N_2)$

Initialize a discount factor $\gamma = 0.9$ [A relatively large discount factor encourages long-term planning and faster convergence during training]

[The warm-up phase starts here]

Freeze the weights of $\mathbf{A}$ as θ_0

Freeze the weights of the $f(phi(s))$ as w_0

for *episode* $e = 1$ to N_0 **do**

  Initialize state s_0

  Preprocess s into s_1 to adapt it as input of $\mathbf{A}$ [preprocessing of high-dimensional states is necessary w.r.t. autoenconders]

  for each time step $t = 1$ to T **do**

    Encode state s_t using $\mathbf{A}$ to get latent embedding $\phi(s_t)$

    Select action a_t using ϵ-greedy policy with $\epsilon = 0.3$ [The learning curves of Game B of Assignment 1 corroborate that a relatively high ϵ could improve Q more efficiently when an e does not end prematurely in the majority of cases]

    Execute a_t and obtain reward r_t and new state s_{t+1}

    Store the transition of the current t, i.e. (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$

    Encode state s_t using $\mathbf{A}$ to get latent embedding $\phi(s_t)$

    Select action a_t using ϵ-greedy policy with $\epsilon = 0.3$ [The learning curves of Game B of Assignment 1 corroborate that a relatively high ϵ could improve Q more efficiently when an e does not end prematurely in the majority of cases]

    Execute a_t and obtain reward r_t and new state s_{t+1}

    Store the transition of the current t, i.e. (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$

  end for

end for

[1]

Algorithm 2 The LSTD update starts here

```
Unfreeze the weights of A, i.e. θ
for episode e = 1 to N_1 do
    Repeat the same steps at phase 1
    for each time step t = 1 to T do
        Repeat the same steps at phase 1
[Start using the autoencoder algorithm to update w]
        Sample a minibatch of transitions from D with a batchsize with a randomly selected
    end for
end for
```

[1]

Algorithm 3 The autoencoder + LSTD training starts here

```
Freeze the weights of A, i.e. θ
Unfreeze the weights of f(phi(s))
for episode e = 1 to N_2 do
    Repeat the same steps at phase 1
    for each time step t = 1 to T do
        Repeat the same steps at phase 1
[Start using the online LSTD algorithm to update w]
        Sample a minibatch of transitions from 𝒟 with a batchsize
    end for
end for
```

[1]