TARGET NETWORK UPDATES IN DQN

Sara Silvestrelli
Dipartimento di Economia, Metodi Quantitativi e Strategie di Impresa
Università degli studi di Milano-Bicocca

ABSTRACT

The use of Target Network in a Deep Q-Network (DQN) approach improves the stability of training. The Target Network can be defined as a slow changing Neural Network by which we keep a copy of our Neural Network and use it for the Q values $Q(s\prime, a\prime)$. In this work we'll see the two main approaches to update the Target Network parameters: soft and hard updates.

Keywords Artificial Intelligence · DQN · Hard Update · Soft Update

1 Introduction

We can indirectly modify the value produced for $Q(s\prime, a\prime)$ by updating our Neural Networks' parameters to make $Q(s, a)$. This can destabilize our training. Making a copy of the Neural Network, the Target Network, fixes this problem using it for the $Q(s\prime, a\prime)$ values for the next states. The Target Network parameters are never trained but they are periodically synchronized with the parameters of the main Q-network. The predicted Q values of the Target Network are used to backpropagate through and train the main Q-network.

The goal is minimizing the distance between the Q-target and $Q(s, a)$ by way of usual gradient descent algorithms. The Q target is unknown, so we use the Bellman Optimality equation to define it at each iteration i as follows [1]:

$$Q_{target} = r_{t+1} + \gamma\, max_{a\prime}\, Q(s\prime, a\prime; \theta_i^{\,-}) \tag{1}$$

where $\theta_i^{\,-}$ are only updated periodically with the Q network parameters assigning $\theta_i^{\,-} = \theta_{i-1}$ and maintaining fixed the target value with the original policy network weights for *C* iterations.

What we want is then optimizing the following square loss of the predicted Q-value and the target Q-value:

$$L_i(\theta_i) = E\big[\big(r_{t+1} + \gamma\, max_{a\prime}\, Q(s\prime, a\prime; \theta_i^{\,-}) - Q(s, a; \theta_i)\big)^2\big] \tag{2}$$

where $\theta_i^{\,-}$ are the parameters used to calculate the Target Network at iteration i.

$Q(s, a; \theta)$ is cloned periodically to a separate target $\hat{Q}(s, a; \theta^{\,-})$ employing a second network that doesn't get trained and ensures that the Target Q values remain stable for a short period. If Target Network isn't used learning would become unstable because the target, $r_{t+1} + \gamma\, max_{a\prime}\, Q(s\prime, a\prime; \theta_i)$, and the prediction, $Q(s, a; \theta_i)$, are not independent as they both rely on θ.

There are two main strategies to update the parameters of the Target Network: the *Hard Update* and the *Soft Update*.

2 Hard Update and Soft Updates

The so called Hard update rule was proposed in the original DQN paper [2]. It consists in updating the Target Network weights by synchronizing them periodically with the Q-Network weights every $C \in \mathbb{N}$ steps.
More formally:

$$\theta^{\,-} \leftarrow \theta \quad \text{when} \;\; mod(c, C) = 0$$

where c is the number of iterations.

The so called Soft update rule was proposed in a following paper [3] and was used by the authors in continuous actions spaces. Unlike the previous case, a soft update is performed at each iteration.
More formally:

$$\theta^{-} \leftarrow \tau\theta + (1-\tau)\,\theta^{-} \quad \text{with } \tau \ll 1.$$

In the reference paper they used $\tau = 0.001$.

Both soft and hard updates ensures that the Target Network is not updated all at once but this happens gradually and frequently stabilizing the learning.

Article Settings

- C parameter defining the target updates was set equal to 20. This means that the target network parameters update happens every 20 episodes.
- For the soft update $\tau = 1e-3$.

References

[1] Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare, and Joelle Pineau. An introduction to deep reinforcement learning. *Foundations and Trends in Machine Learning*, 11(3-4), 2018.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[3] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2015.