Deep Query Network

Gedu College of Business Studies

Gedu, Chhukha

Assignment Submitted by:

03240363	Jigme Namgyel Dorji
03240343	Chetan Ghimiray
03240357	Gayley Namgay

Bachelor of Business Intellingence, 3rd Sem,MLA

12-12-2025

Royal University of Bhutan

DECLARATION

Module Code:	MLA202	Type of Course Work:	Group Task
Module Title:	Reinforcement Learning	Module Tutor:	Mr.Douglas Sim
Date of Submission:			

We hereby declare that this academic work is our own and those referred ideas from other sources have been appropriately acknowledged. The material in this submission has not been previously submitted for assessments. We understand that if found otherwise, our academic work will be cancelled and no marks will be awarded besides legal consequences.

03240363	Jigme Namgyel Dorji
03240343	Chetan Ghimiray
03240357	Gayley Namgay

FOR MODULE TUTOR

Sl. No	***Marking Criteria***	***Marks Assigned***	***Marks obtained***
1	*Introduction and Problem Statement*	***2***	
2	*Methodology and Literature Review*	***3***	
3	*Findings and Discussion*	***3***	
4	*Organization and References*	***2***	
Total			

Feedback:

Signature of Module Tutor

1. Introduction & Problem Statement

The aim of reinforcement learning (RL) is to learn the behavior of agents that maximize decisions by interacting with an environment. During a significant number of years, by having linear function approximators, the value-based RL algorithm, Q-learning, was used in conjunction to control high state spaces (Watkins and Dayan, 1992). However, integration of non-linear function approximators e.g. neural networks in RL were found to be famously unstable and often divergent. This instability is based on two main factors, first, sequential, correlated observations in RL violate the independent and identically distributed (i.i.d.) assumption on stable neural-network training, second, the simultaneous change in target Q-values across updates creates an inherent problem of a moving target that triggers harmful feedback dynamics (Mnih et al., 2015).

In the journal Nature the introduction of Deep Q -network (DQN) represented a breakthrough in deep reinforcement learning (Mnih et al., 2015). DQN confirmed the usefulness of neural networks to solve complex RL problems by showing that one algorithm could learn to play a set of heterogeneous atari 2600 games to a human-competitive level only with the use of raw pixels as input signals. More importantly, this success was not based on a paradigm shift of Q-learning itself, but on the groundbreaking advances in architecture that would help to curb the instability inherent in a neural-network-based approximation of functions in RL.

The report focuses on the implementation and analysis of the core DQN algorithm without involving pixel-based games on the way but canonical continuous-controlled tasks, i.e., CartPole -1 environment. Eliminating the complexities of the convolutional networks of visual perception, the study allows a more focused look at the essence of DQN, that is, experience replay and target networks, and clarifies how they enable the training stability and the process of effective learning of high-dimensional states observations.

2. Methodology and Literature Review

The proposed methodology is based on the seminal DQN formulation suggested by Mnih et al. (2015). The algorithm is a variation of Q -learning, where the action-value function, Q (s, a) is estimated using a deep neural network with parameters 0.

Core Algorithm Components:

The foundation is the Q-learning update rule:
$\mathbf{Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]}$

where α is the learning rate and γ is the discount factor. In DQN, this is implemented by minimizing a sequence of loss functions.

The critical innovations of DQN that enable stable learning are:

Experience Replay: The agent's observations at each time step, $\mathbf{e_t = (s_t, a_t, r_t, s_{t+1})}$, are represented as an experience in a replay buffer **D**. Sampling of this buffer is done randomly, in mini-batches, during training. This operation eliminates the time correlations of the successive samples, thus smooth data distribution and improved efficiency of data is achieved by reusing the past experiences (Lin, 1992).

Target Network: A separate target network with parameters θ^- is used to compute the Q-learning target $y = r + \gamma \max_{a'} Q(s',a'; \theta^-)$. The parameters θ^- are copied from the online network θ only every C steps and are held fixed between updates. This decouples the target from the rapidly changing online estimates, dramatically reducing instability.

Loss Function: The network is trained by reducing the mean squared error (MSE) between the Q-learning target and the existing Q-value prediction:

$\mathbf{L(\theta) = [(r + \gamma \max_{a'} Q(s',a'; \theta^-) - Q(s,a; \theta))^2]}$

State and Action Spaces:

The CartPole-v1 problem has a four-dimensional continuous state space, which consists of the cart position, cart velocity, pole angle, and pole angular velocity. The action space is discrete and has two actions, which are either application of force to the left (0) or to the right (1). It receives a reward of +1 at every step in time, when the pole is upright, but the episode is completed when the angular displacement of the pole exceeds a predetermined limit or when the displacement of the cart reaches a preset limit.

Network Architecture:

The implemented Q-network is a relatively simple multi-layer perceptron (MLP):

- Input Layer: 4 neurons (state dimensions).
- Hidden Layers: Two fully connected layers with 128 neurons each, utilizing the Rectified Linear Unit (ReLU) activation function to introduce non-linearity.
- Output Layer: 2 neurons, representing the predicted Q-value for each possible discrete action (left, right). The agent selects actions using an ε-greedy policy to balance exploration and exploitation.

3. Findings & Discussion

Convergence of Training and Stability: are achieved during the implementation, whereby the implementation learns how to master the CartPole-v1 environment to achieve a mean score of above 195 within 100 consecutive episodes. As seen in Figure 1, the learning curves have the canonical behaviour of deep Q -learning: a low and fluctuating performance at the start of the replay buffer accumulation phase, and then a rapid but comparatively stable improvement phase, and finally a plateau of near-optimal policy performance.

Impact of Experience Replay: Systematic ablation was used to investigate the effect of experience replay. Removal of replay resulted in significantly unstable learning dynamics and inability to obtain a reliable convergence. The performance of the agent showed strong oscillations based on catastrophic forgetting which is due to temporal correlated updates. Samples of replay buffer were found to be invaluable in creating training data that resembles i.i.d. properties and thus stabilises learning.

Effect of Target Network Update Frequency:A key, important hyperparameter is the target-network update frequency, which is denoted by C. A very high rate, such as updating on a stepwise basis was reintroducing instability, since the target network reflected the instability of the online network. On the other hand, too low frequency slowed down learning process as target values became stale. A mid-frequency was found to give an optimal trade-off, to provide the constant targets, yet to allow timely delivery of learning progress.

Comparison with Vanilla Q-learning: Compared to the traditional tabular Q-learning, the fact that the latter cannot be applied to a continuous state space like CartPole-v1 without significant discretization highlights that the latter method is highly problematic itself and further exacerbates the situation. Furthermore, a linear function approximator used in a conventional Q-learning system did not achieve the comparable performance on a consistent basis, which highlights the need of the nonlinear representational ability of DQN along with the stabilisation procedures in addressing this type of issues.

Limitations:

- **Sample Inefficiency:** DQN requires a huge amount of interactions between it and the environment, on the order of millions of frames in the case of Atari simulations, to develop competent performance, thus making it computationally expensive when compared to modern model-based or policy-based methods.

- **Overestimation Bias:** Van Hasselt et al. (2016) have shown that the maximization operator used in the Q-learning target produces an upward bias in Q-value estimates in turn, compromising performance. This initial weakness of the baseline DQN has then been overcome by other algorithms like the Double DQN.
- **Discrete Action Limitation:** The standard DQN architecture is fundamentally designed for discrete action spaces, limiting its direct applicability to continuous control tasks without significant modification (e.g., Actor-Critic methods).

4. Organization

The report is organized in a way that it presents the historical problem and the solution to be offered by Deep Q - Networks, presents the methodological elements of the solution, which is based on seminal literature, and finally presents the analysis of empirical results and limitations.

References

Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine Learning, 8(3–4), 293–321. https://doi.org/10.1007/BF00992699

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529–533. https://doi.org/10.1038/nature14236

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). The MIT Press.

van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-learning. Proceedings of the AAAI Conference on Artificial Intelligence, 30(1). https://doi.org/10.1609/aaai.v30i1.10295

Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. Machine Learning, 8(3–4), 279–292. https://doi.org/10.1007/BF00992698