AE598 Hw2: DQN

Dennis McCann
Department of Aerospace Engineering
University of Illinois at Urbana-Champaign
Urbana, IL, USA
dmccann3@illinois.edu

I. Deep Q-Learning

This homework investigates the implementation of the algorithm Deep Q-Learning on an inverse pendulum environment. Deep Q-Learning is an improvement upon the value based, model free TD control algorithm of Q-Learning. Basic Q-Learning will calculate the action value, Q, for each state in the state space. Because of the tabular method of calculating the action value for each state, basic Q-Learning is unable to be applied to large or continuous state spaces. This is where Deep Q-Learning improves upon the basic Q-Learning, using a deep neural network to estimate the action values for any state in a state space, allowing Deep Q-Learning to be applied to large and continuous state spaces.

A. Algorithm and Hyperparameters

The algorithm uses two deep neural networks to appro
imate the action value function from which the policy
derived from. One network desribed as $\hat{Q}$ is the target ne
work for generating targets in the Q-Learning update fro
which the gradient descent step is performed upon. In th
implementation of the DQN algorithm, the neural netwo
used 2 hidden layers of size 64 each with hyperbolic tang
activation functions. The loss function used as Huber Loss
SmooothL1Loss in pytorch. The algorithm also uses exper
ence replay, in which the agent's experience is stored at eac
time step $e_t = (s_t, a_t, r_t, s_{t+1})$ in a data set D pooled ov
many episodes. These experiences are then used to train the
network during minibatch updates. The network is optimize
every C episodes using a random sample from the repl
buffer. The hyperparameters used for training can be foun
in I.

These hyperparameters are used for all trained polici
unless specifically changed. In the ablation study of th
homework, four trained policies were compared. A standa
policy that uses the hyperparameters from I, one where t
target policy is not used, meaning the target network $\hat{Q}$
reset everytime, one where is the replay buffer is not use
meaning the replay buffer size is the same as the minibat
update size, and lastly, one that does not use a target or repl
buffer.

B. Learning Curve

The learning curve for a for standard DQN algorithm is shown in 1. The learning curve shows the majority of returns from each episode hitting above 10 for every episode after

TABLE I
Hyperparameters of DQN

Hyperparameter	**Value**
Number of Episodes	700
Learning Rate	1×10^{-3}
γ	0.95
Update Frequency	100
Replay Memory Size	$100,000$
Batch Size	64
Starting ϵ	1.0
Ending ϵ	0.02
ϵ Decay	$10,000$

100. There is a large amount of variance in reward but the majority stay between the range of 12 and 15.

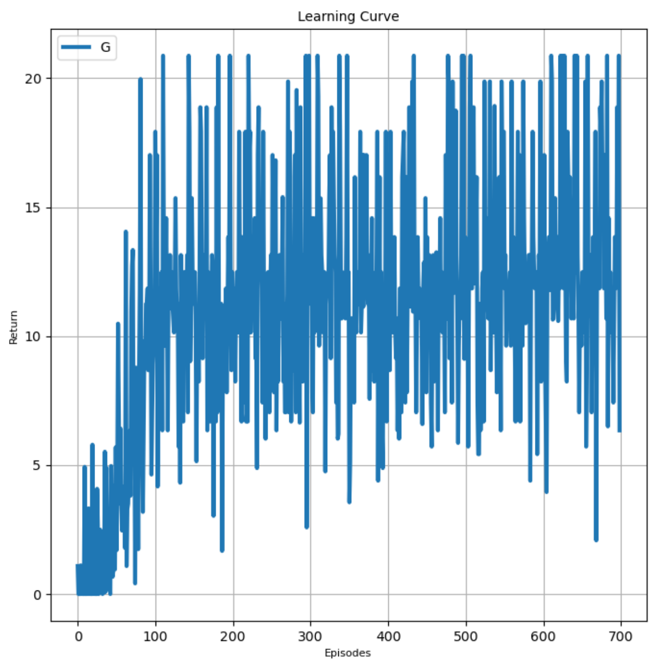

Fig. 1. Learning Curve for basic DQN

Figure 2 shows an example trajectory of the trained policy. The state reaches its inverted equilibrium around time step 15 and keep steady through the remainder of the episode.

C. Example Trajectory

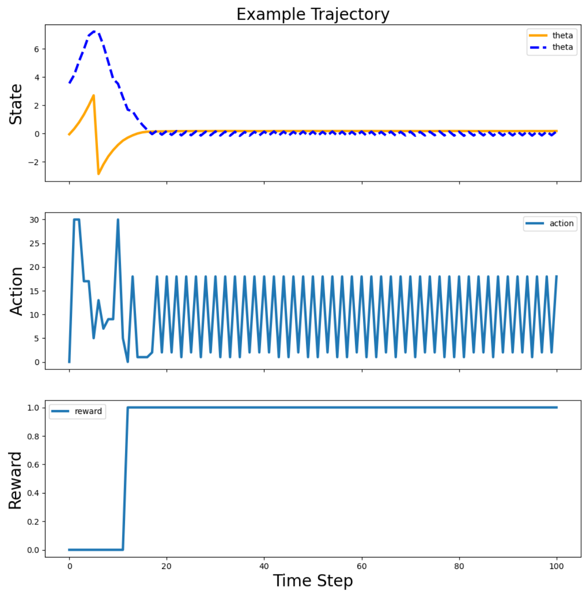

Fig. 2. Example trajectory for basic DQN

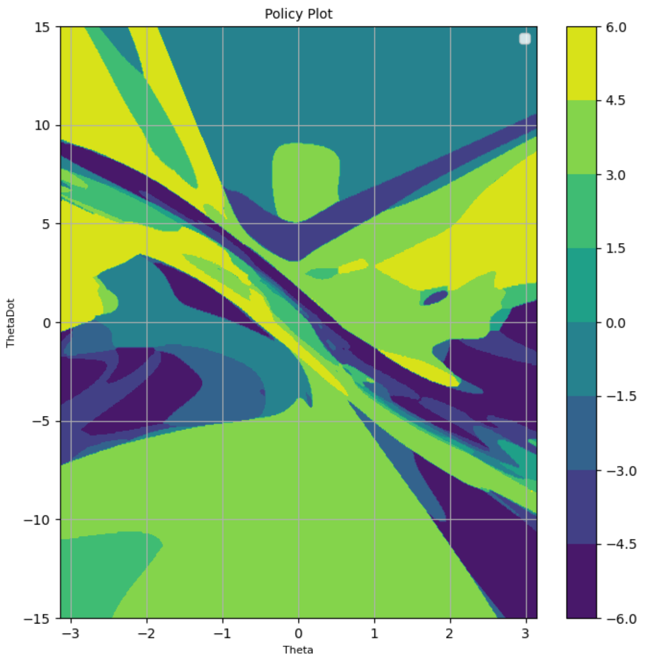

Fig. 3. Policy plot for basic DQN

D. Policy

The value function plot 4 for the standard algorithm show a state near the equilibrium being greater than states furth away. Intuitively this make sense as is validation that the polic trained is accurate.

E. Value Function

II. Ablation Study

The ablation study run compared four policies. A standa policy that uses the hyperparameters from I, one where tl target policy is not used, meaning the target network $\hat{Q}$ reset every time, one where is the replay buffer is not use meaning the replay buffer size is the same as the minibat update size, and lastly, one that does not use a target or repla buffer. The results from the standard algorithm are shown i the seciton above.

A. Without Target, With Replay

First, a policy without a target, but with a standard repla buffer was trained. The learning curve can be seen in 5. is similar to the learning curve from the standard algorith but with slightly worse returns, dipping under 10, at the er of training. The trajectory seen in 6 can especially show tl differences between a policy trained with no target and tl standard policy. There is much more oscillation of the state theta around 0 were as the standard algorithm was much more stable.

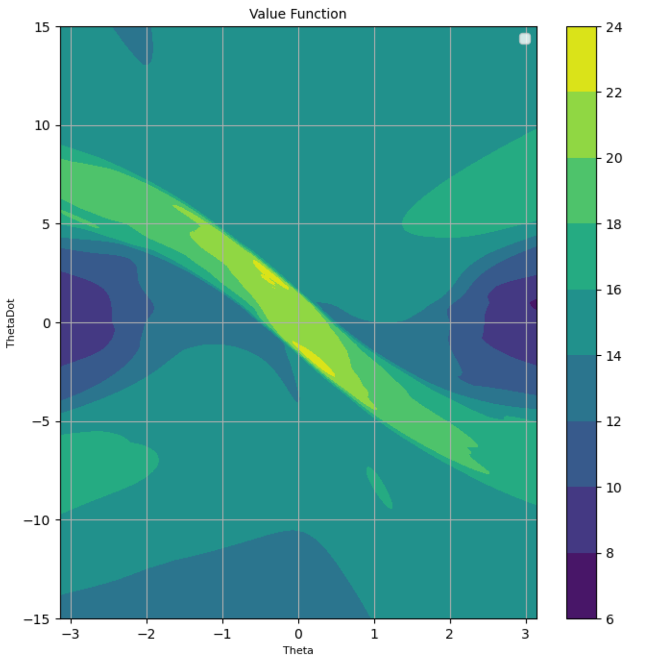

Fig. 4. Value Function for basic DQN

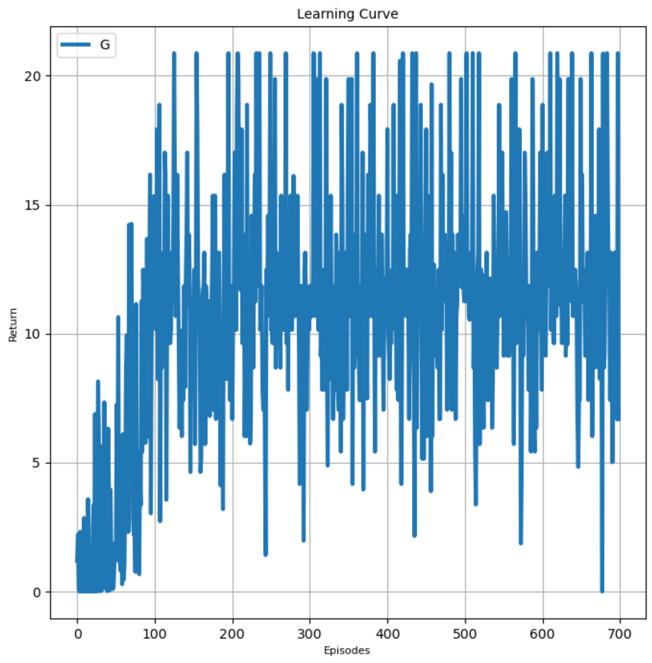

Fig. 5. Learning Curve for DQN without using a target Q and with using replay buffer

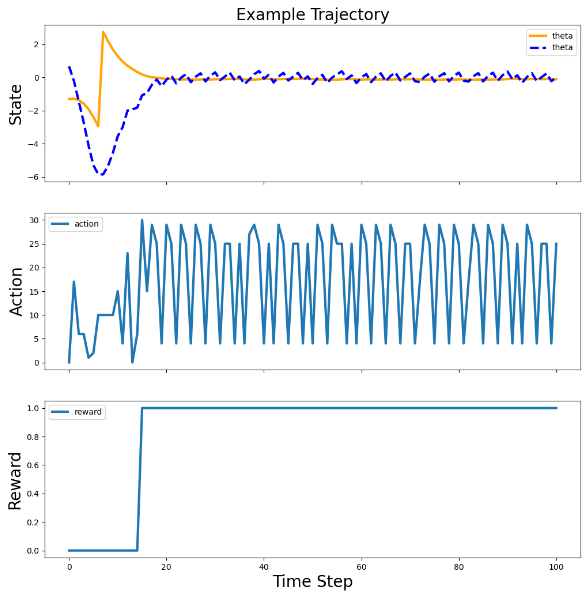

Fig. 6. Example trajectory for DQN without using a target Q and with using a replay buffer

The value function plot seen in 8 shows higher values in the pocket around $(0, 0)$ which is better than most other permutations of the DQN ablation study.

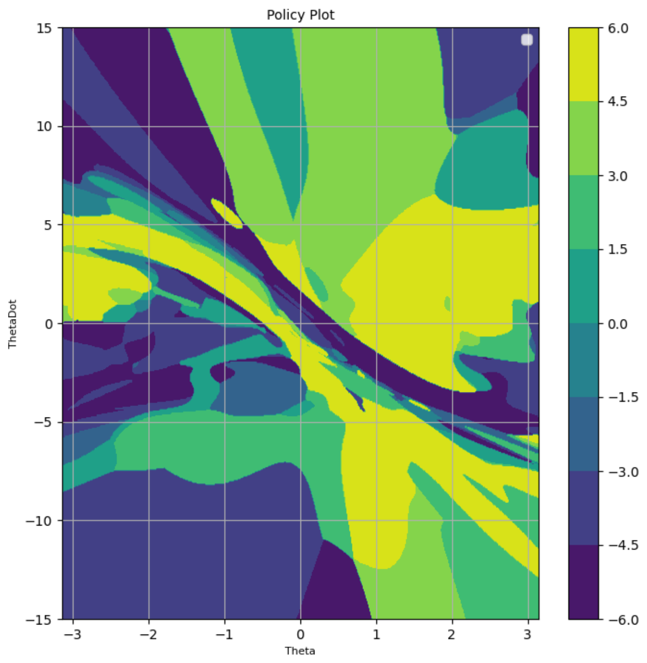

Fig. 7. Policy plot for DQN without using a target Q and with using a replay buffer

B. *With Target, Without Replay*

This experiment used a target but set the size of the replay buffer to be the same size as the batch update. The learning curve 9 from this trained policy has an interesting feature where the return becomes very negative for one epidsode before jumping back up to around 10. One reasone for this could be because of the small amount of data the buffer was using. The data is the buffer at the time could have been bad data with no option but to choose to use that data to update the Q network using these experiences. A larger buffer gives a greater distribution of data which allows the network to be trained on a more accurate sample of data from the environment. The trajectory 10 from this trained policy has similar characteristics from the policy trained with no target. The state theta has a higher amount of oscillation around 0 than the standard trained algorithm. This again can be atributed to the lack of data to train the network, leading to a worse policy.

The state value function for this trained policy shows a large range of high values throughout the state space. The state values do increase around 0 but are generally to high for states that are far away from the equilibrium point.

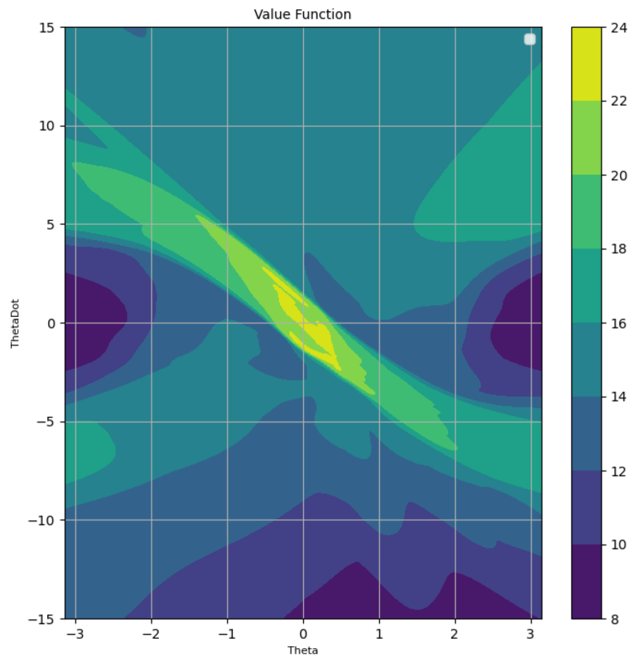

Fig. 8. Value Function for DQN without using a target Q and with using a replay buffer

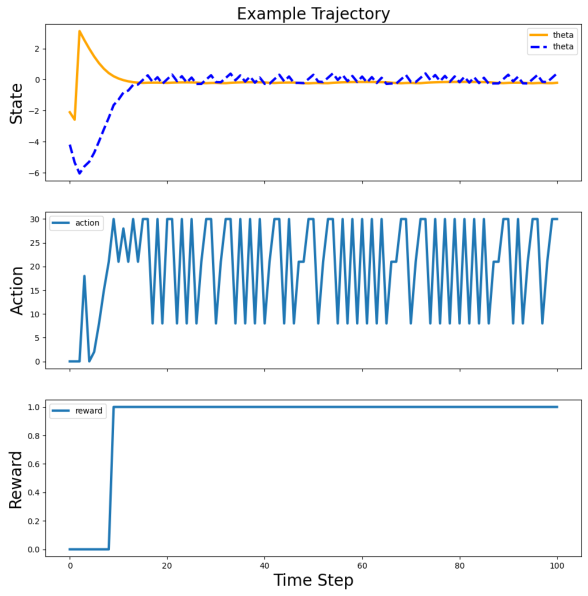

Fig. 10. Example trajectory for DQN with using a target Q and without using a replay buffer

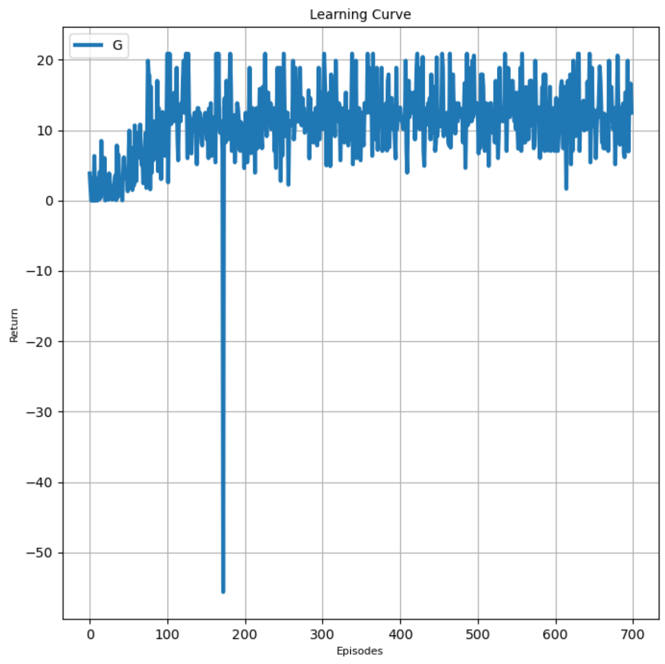

Fig. 9. Learning Curve for DQN with using a target Q and without using a replay buffer

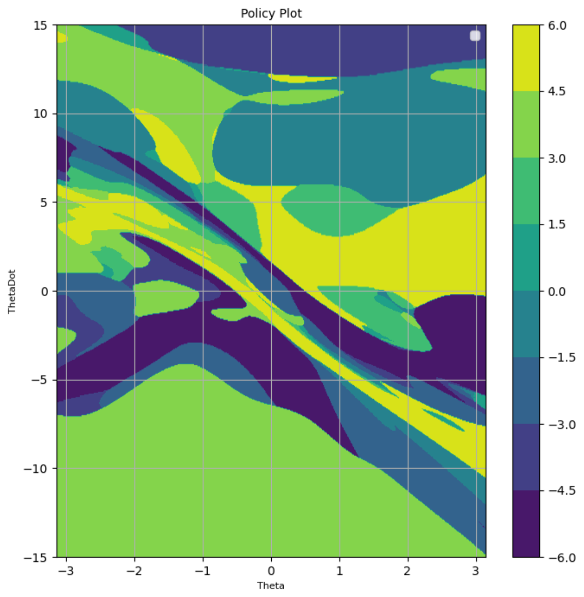

Fig. 11. Policy plot for DQN with using a target Q and without using a replay buffer

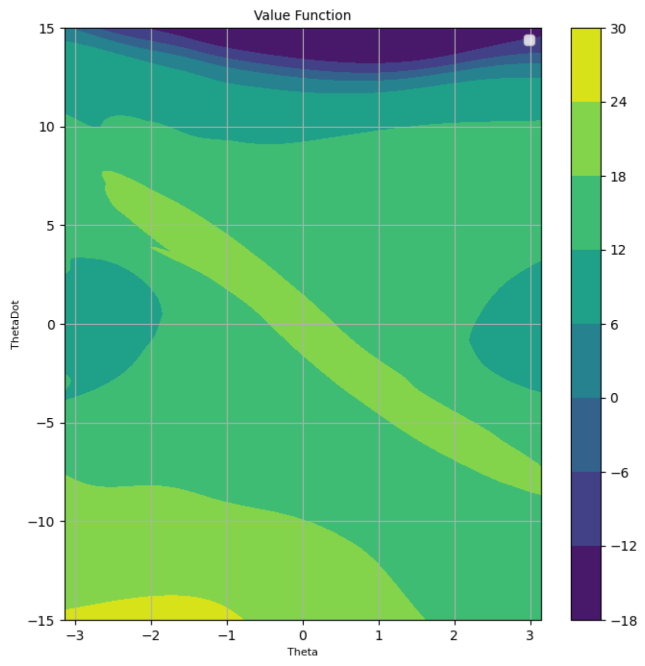

Fig. 12. Value Function for DQN with using a target Q and without using a replay buffer

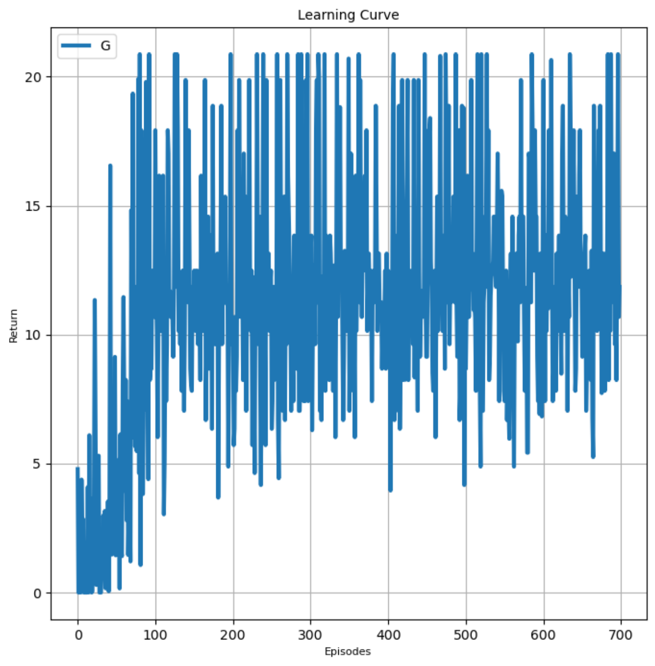

Fig. 13. Learning Curve for DQN without using a target Q or using a replay buffer

C. Without Target, Without Replay

Lastly a policy was trained without a target or a replay buffer. This policy, however, seemed to perform better than the policy trained with a target but without a replay buffer. The learning curve from this study 13 was consistently above 10 but mostly between the ranges of 10 and 13. The trajectory showed less oscillation than the policies trained with target and without replay, and without target but with replay, but still showed more oscillation than the standard algorithm.

The value function looks very similar to the value funciton from the standard algorithm, showing that the performance of the policy trained without target or policy is still accurate.

D. Ablation Study Summary

To summarize the results from the ablation study, the mean reward for each permutation of target and replay buffer use was average over 200 episodes. The results of these simulations can be seen in II. From the table we can see that the standard algorithm performed the best out of all of the trained polices, with the policy trained without target or replay coming in second. The policy using a target but no replay performed the worst. This could be from the fact that final trained Q-Network was not trained on a full distribution of experience from the environment.

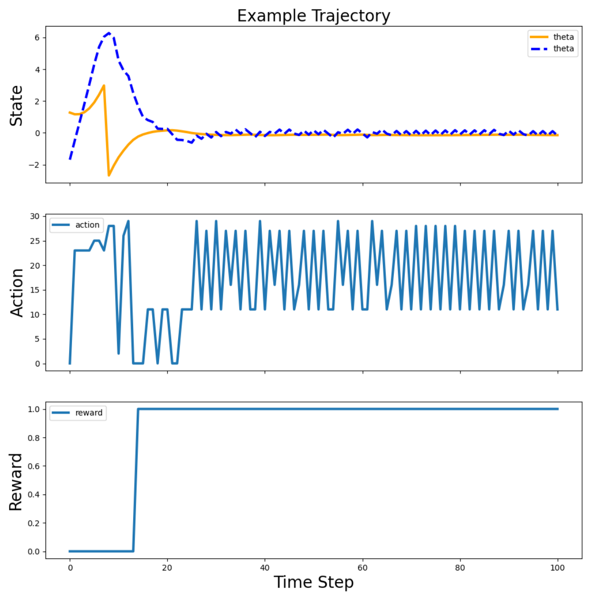

Fig. 14. Example trajectory for DQN without using a target Q or using a replay buffer

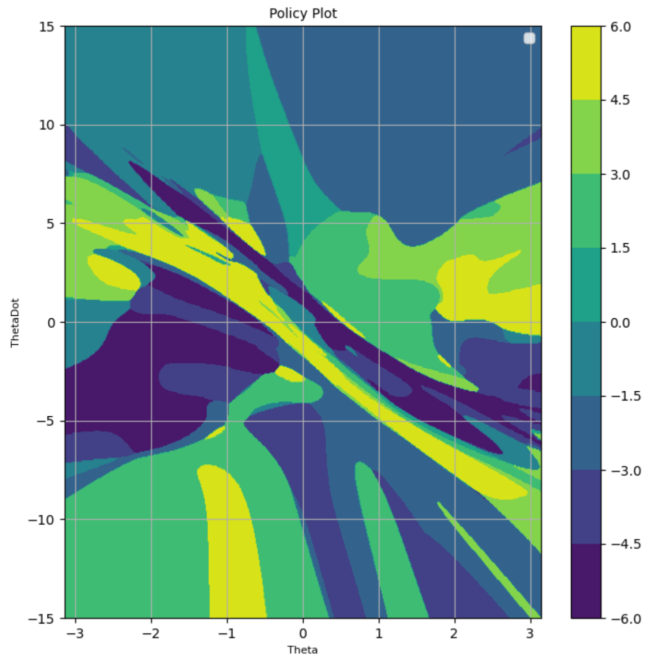

Fig. 15. Policy plot for DQN without using a target Q or using a replay buffer

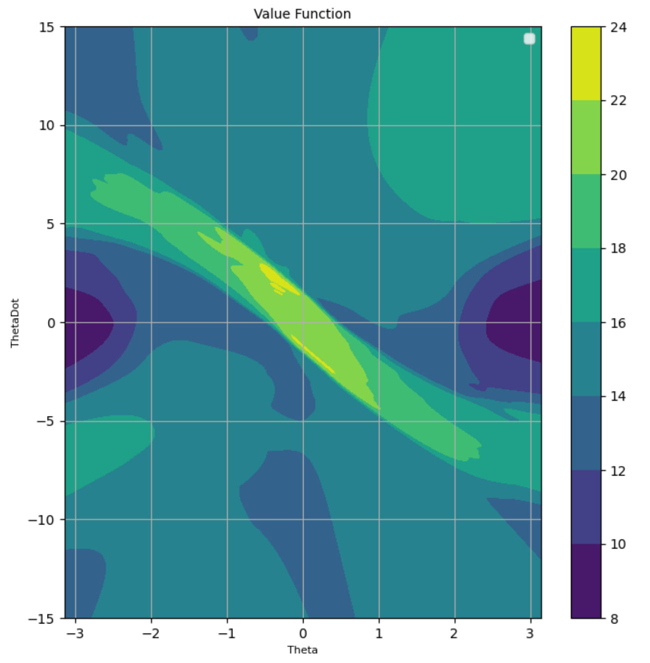

Fig. 16. Value Function for DQN without using a target Q or using a replay buffer

TABLE II
AVERAGE SCORE FROM EACH TRAINED POLICY OVER 200 EPISODES

Trained Policy	**Average Reward**
Standard DQN	0.898
No Target DQN	0.883
No Replay DQN	0.881
No Target No Replay DQN	0.888