Temporal Difference Learning Algorithms Analysis of SARSA, Q-Learning and Double Q-Learning Technical Report

Riashat Islam
McGill University
Reasoning and Learning Lab
riashat.islam@cs.mcgill.ca

February 9, 2017

1 Introduction

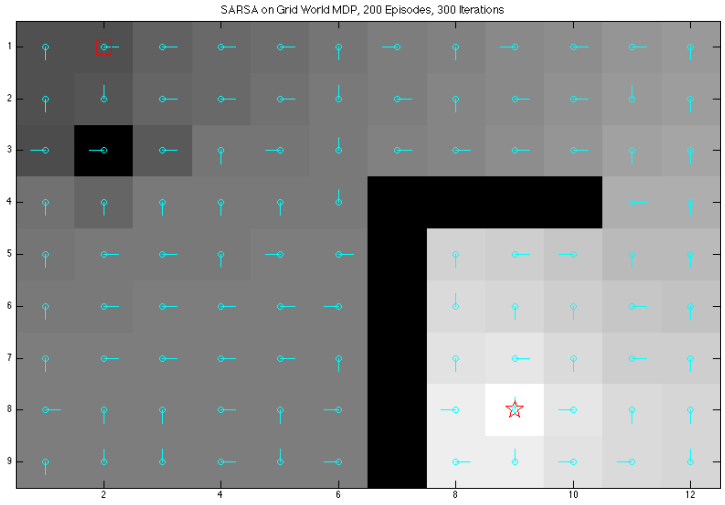
In this work, we experimentally examine the differences between different temporal difference learning algorithms. At first, we analyse differences in convergence and learning rate, and convergence to the optimal policy, for the on-policy SARSA and off-policy Q learning algorithm. We further evaluate the differences and analyse experimentally the benefits and drawbacks of using Double Q-learning algorithm, compared to using SARSA or Q-learning. We provide experimental results and analysis of the following:

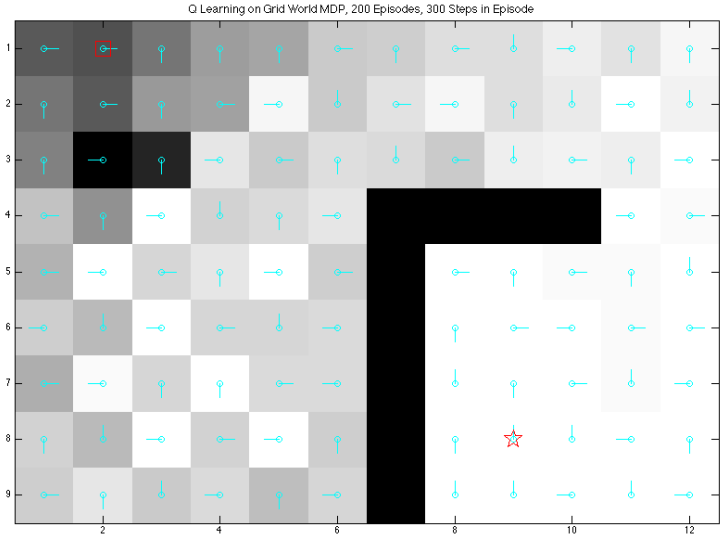
- Demonstration of SARSA and Q-Learning Algorithms on Grid World and Cliff World MDPs
- Compare SARSA and Q-Learning on Cliff World MDP
- Effect of ϵ-greedy exploration and step size on SARSA and Q-Learning
- Multi-Step TD : Convergence of Multi-step SARSA and Multi-Step Q learning algorithms
- Double Q Learning : Experimental Evaluation on Cliffworld MDP
- Off-Policy Behaviour Policy in Double Q-learning
- Comparing Q-Learning and Double Q-Learning
- Effect of Discount Factor γ on Q and Double-Q Learning
- Comparison of SARSA, Q-Learning and Double-Q Learning
- Analysis of Double Q-Learning Algorithm

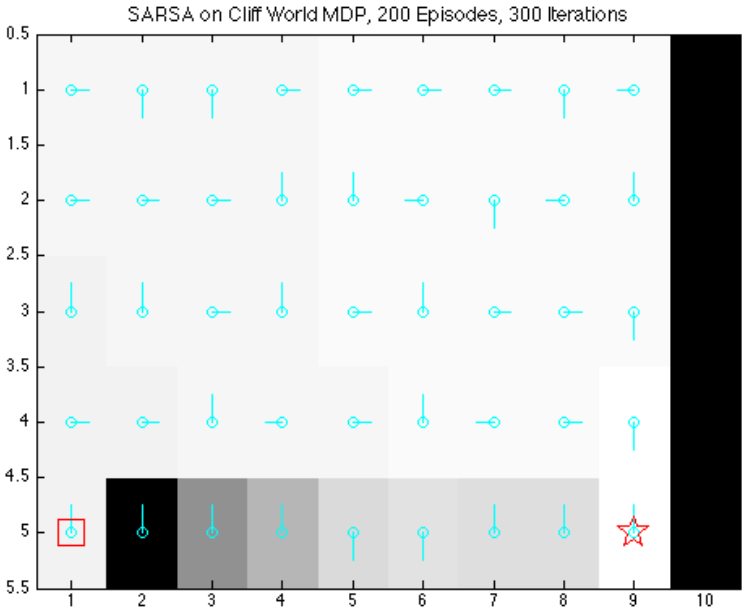
2 Experimental Results

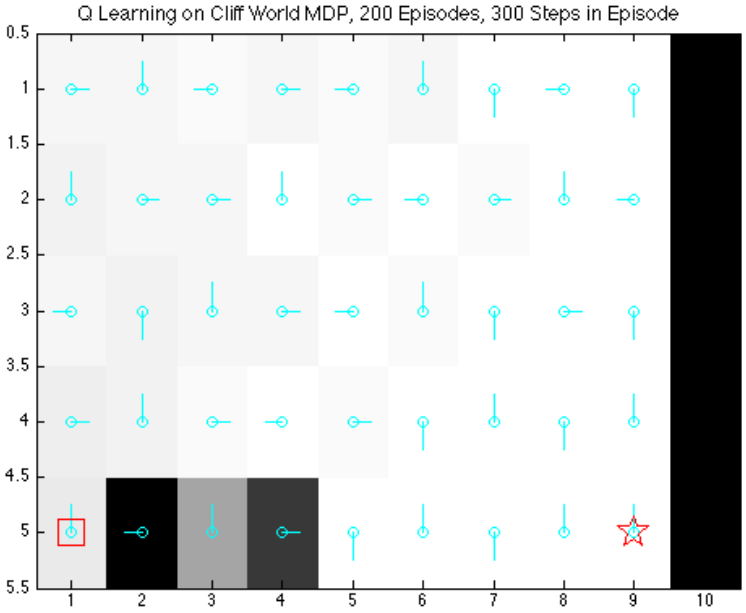
2.1 SARSA and Q-Learning

At first, we evaluate our implementation of the SARSA and Q-Learning algorithm on Grid World and Cliff World MDP. Our results show the regions where a better estimate of the value function is achieved, and regions where the value function is close to an optimal value function. The actions at every state are shown by the arrow sign.







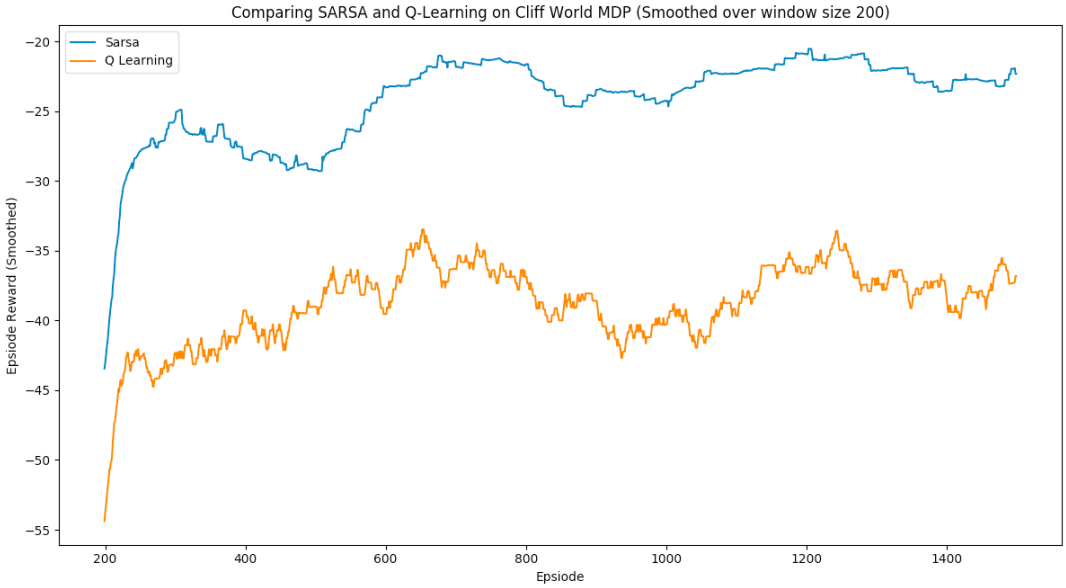


2.2 Comparing SARSA and Q-Learning

We then evaluate the differences in convergence to an optimal policy for the SARSA and Q-Learning algorithms on the cliffworld MDP. Experimental results show that SARSA converges to a better op-

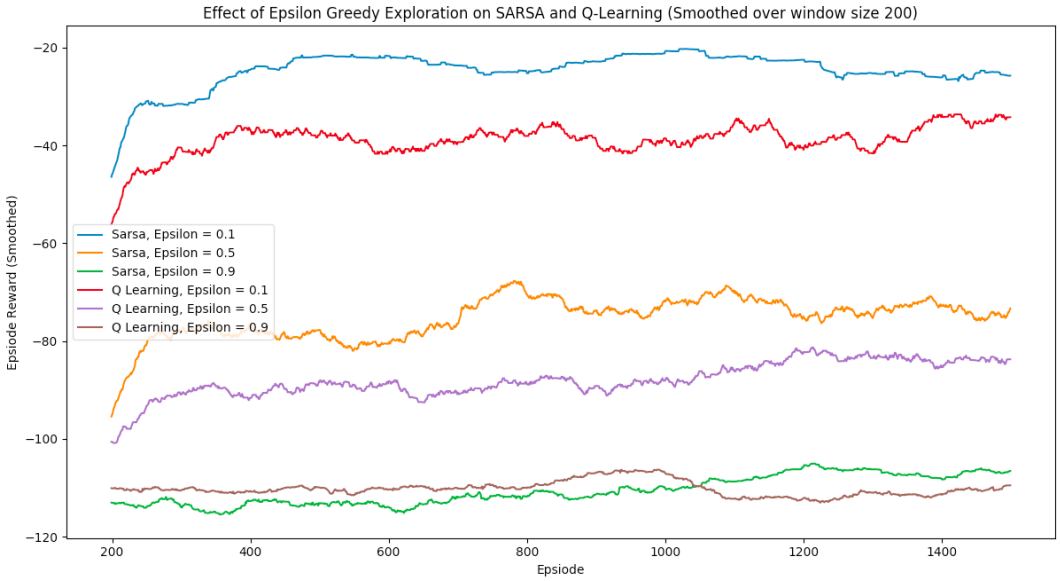
timal value function that the Q-Learning algorithm.

Since Q-learning is an off-policy algorithm whose action selection also depends on the ϵ-greedy behaviour policy, the agent following Q-learning occasionally falls off the cliff. As demonstrated in the results, the SARSA algorithm follows a safer path being an on-policy algorithm. Here, both the algorithms reach convergence and learns an optimal policy. However, the on-line performance of Q-learning is shown to be worse than that of SARSA.



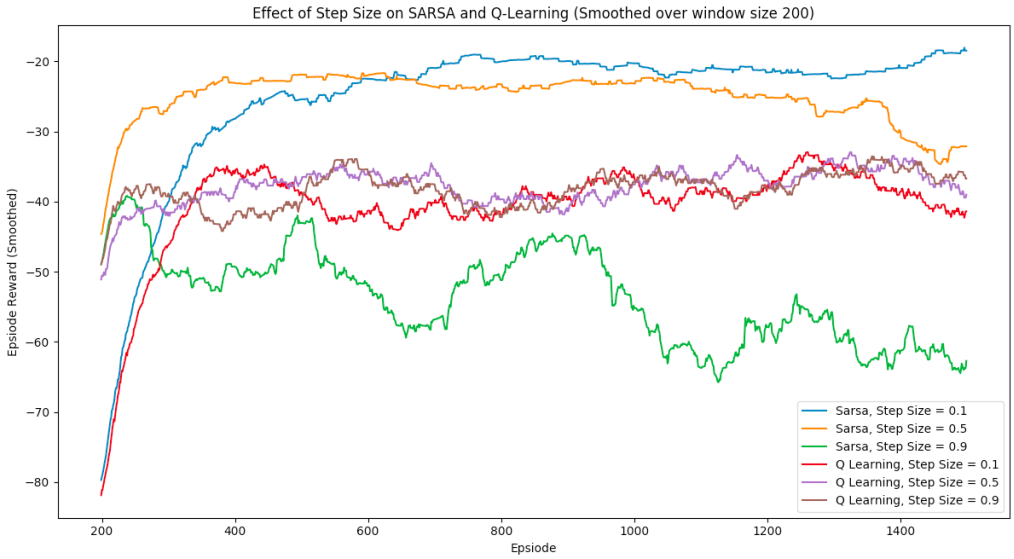
Effect of changing ϵ parameter for the greedy policy

We then evaluate the significance of chaing the ϵ parameter on the Q-learning algorithm. Experimental results show that as ϵ parameter is reduced for Q-learning, it asymptotically converges to the optimal policy, and finds a path which is almost as good as the SARSA algorithm. SARSA, being the on-policy algorithm, also shows that high ϵ values indeed worsens the performance for SARSA, and also for the Q-learning algorithm.



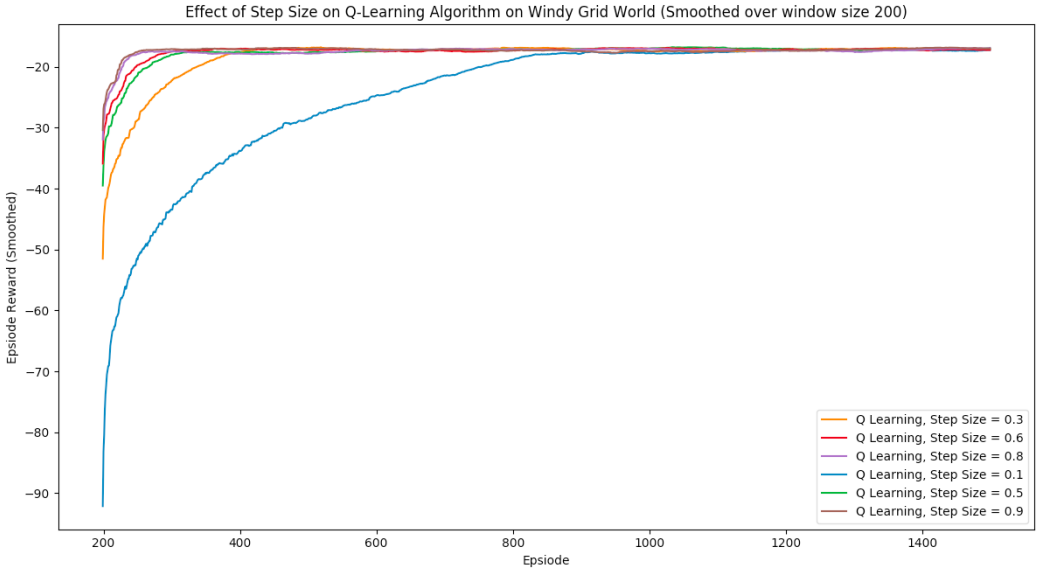
Effect of Step Size on Q-Learning Algorithm

We further investigate the effect of the step size parameter on both Q-learning and SARSA, keeping the ϵ parameter same at 0.1. The step size parameter affects the learning rate of the agents.



Experimental results show that, with different step size parameters, SARSA is more affected than the Q-learning algorithm. The convergence to optimal policy for Q-learning remains the same, even though speed of convergence may vary for Q-learning. In contrast, for a high step size, SARSA significanly performs worse. This is mainly because SARSA being an on-policy algorithm, it's value function updates are highly dependent on the step size. However, since in Q-learning, we do a maximisation over the Q-function, this dominates the estimation of action-values in Q-learning, making

Q-learning less dependent on the step size to reach convergence.

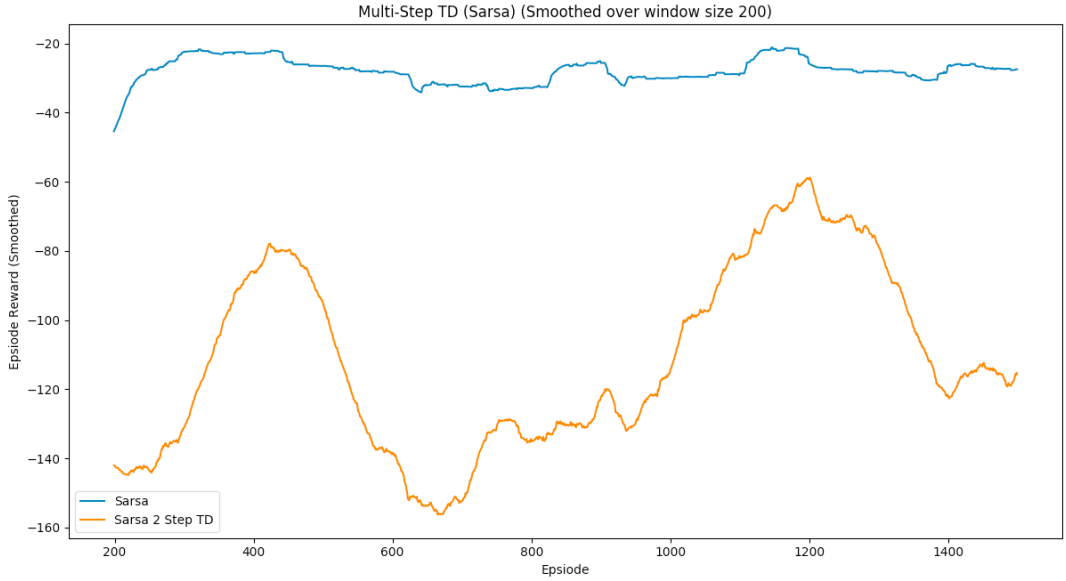


Result above for Q-learning further justifies that the Q-learning algorithm always converges to the same optimal policy, irrespective of the step size. However, the speed of convergence, as expected, gets affected by the step size parameter in Q-learning.

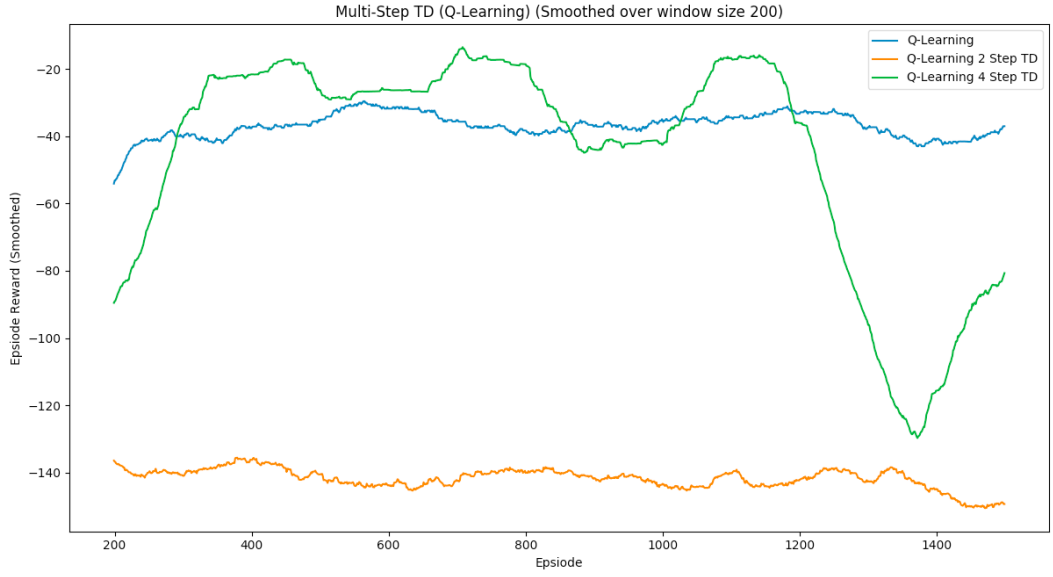
2.3 Multi-Step TD Learning

In this section, we further evaluate multi-step TD learning algorithms. Our results are again demonstrated on the cliffworld MDP.

On evaluating the SARSA 1-step and SARSA 2-step TD updates for cliffworld, our results show that SARSA 2-step performs worse. This is because in the cliffworld environment, falling off the cliff results in large penalty. Since SARSA 2-step updates value estimates based on two steps ahead, the chance of the agent falling off is higher for following 2-step TD learning. This results in the worsen performance of SARSA(2) compared to SARSA(0).

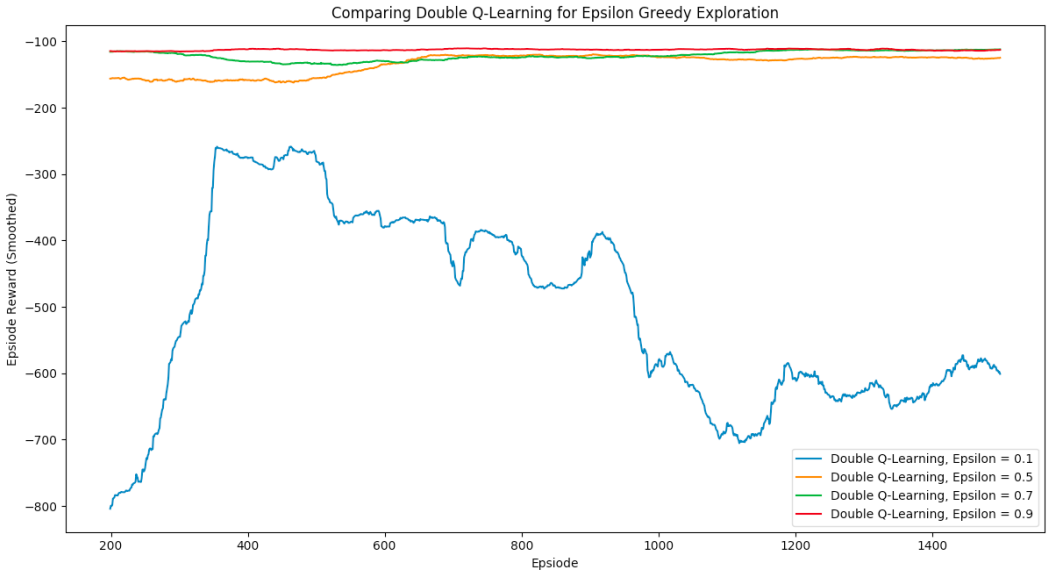


This is further evaluated by analysing Q-leanring with multi-step TD updates. Surprisingly, results show that Q learning(2) performs worse than Q learning(4). This may also be because of the dependence on off-policies. Q-learning follows an off-policy behaviour policy, which is also epsilon greedy in this case, and the off-policy exploration (which has random effects due to the exploration strategy) may cause for such results as below.



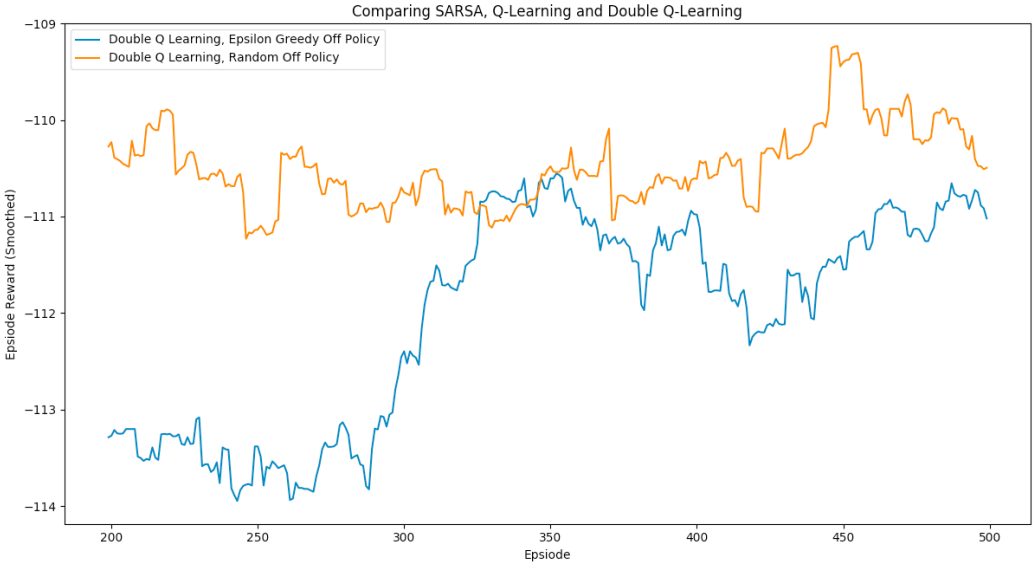
2.4 Double-Q-Learning

In this section, we present experimental results demonstrating the Double Q-Learning algorithm. Double Q-learning, also being an off-policy algorithm, therefore depends on the ϵ greedy behaviour policy.



Our results show that DoubeQ in fact performs better for higher values of ϵ compared to the ϵ parameter being low. This indicates that in DoubleQ, maybe a random off policy is perhaps more favoured than an epsilon greedy policy which computes the greedy maximisation based on the average of the Q functions. We further evaluate for this, by comparing DoubleQ with different off-policy behaviour policies (ϵ-greedy and random off policy).

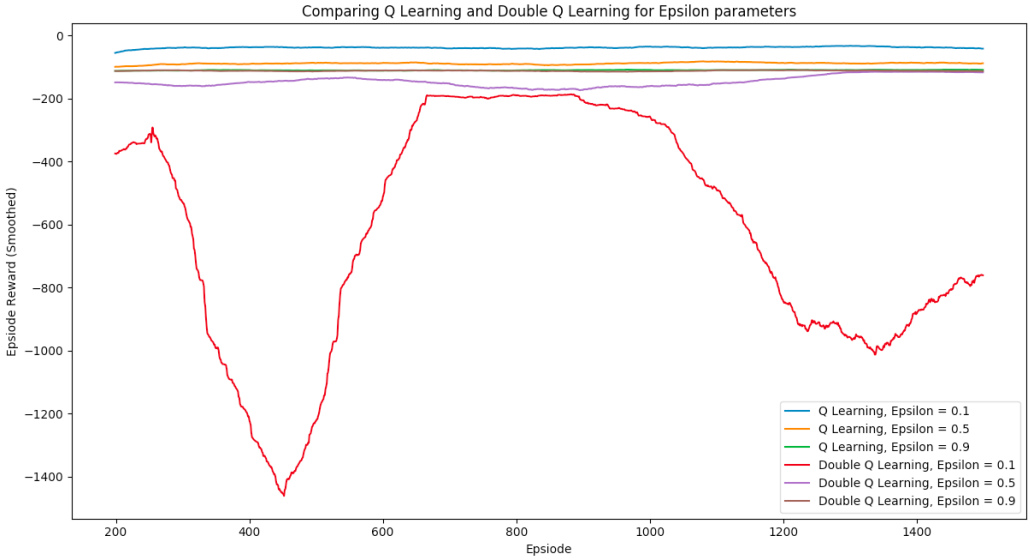
2.5 Off-Policy in Double Q Learning



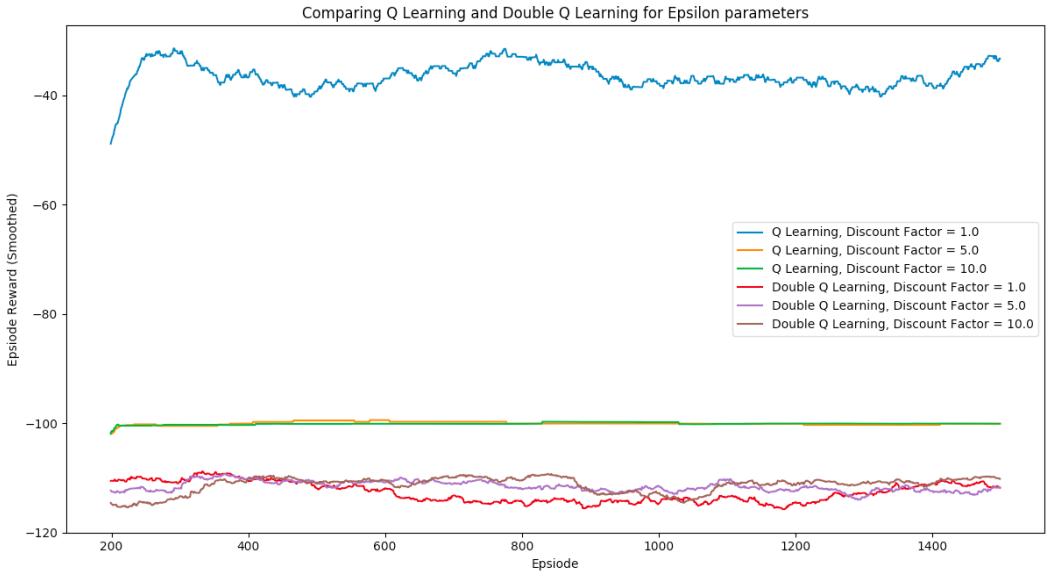
Experimental results show that for Double Q-Learning, perhaps a random behaviour policy maybe more favoured than an epsilon greedy behaviour policy. The agent following greedy exploration only starts performing well after certain number of episodes.

2.6 Comparing Q-Learning and Double Q-Learning

We then compared Q-learning and Double Q-learning for different ϵ parameters. Results show that Q-learning performs well with a high value of ϵ, whereas Double Q-learning prefers lower ϵ parameter.



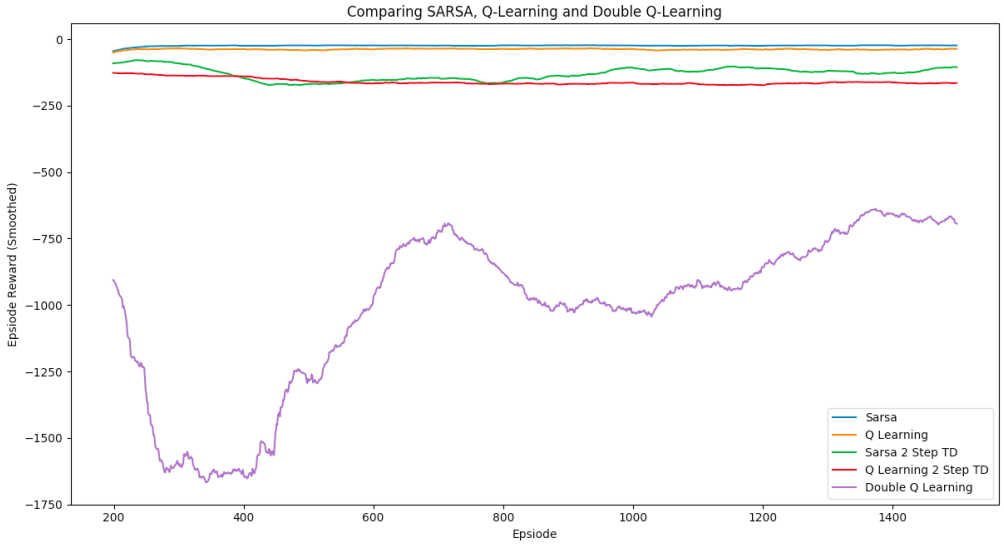
Effect of the Discount Factor γ:
As claimed in the original DoubleQ paper, Q-learning seems to be highly affected the discount factor γ. This is because Q-learning makes a biased estiamte of the action-value function. For the double Q learning algorithm, there seems to be no effect on its learning performancce depending on the discount factor.

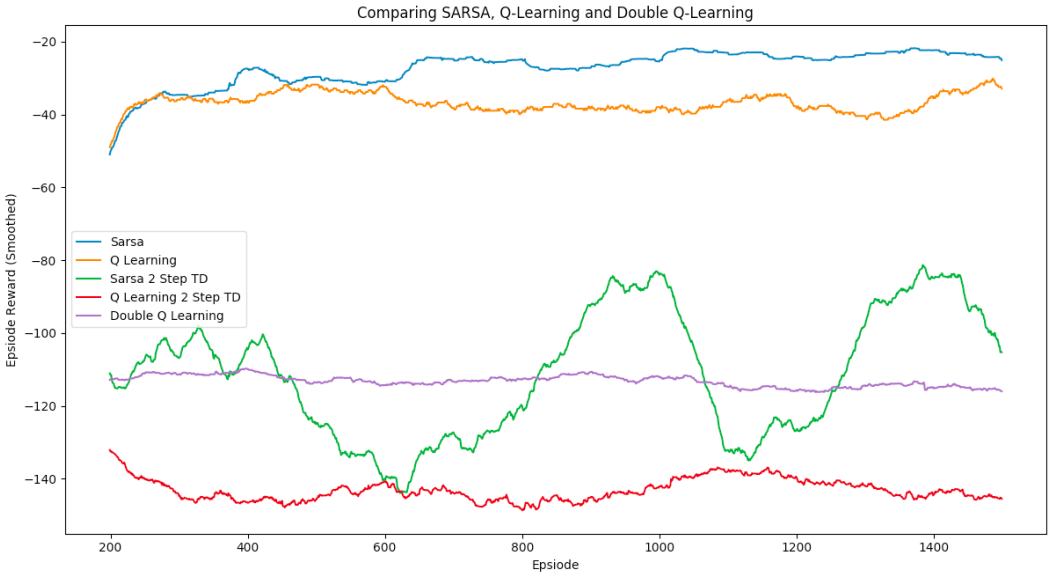


However, note that for our experimental results, a well tuned Q-learning with right values of γ and ϵ seems to outperform the Double Q learning algorithm. This is also because the biased estimate of Q-learning is not visible on all environments. It is only in certain cases (MDPs) where the biased expectation that Q-learning makes actually affects its learning performance.

2.7 Comparison of SARSA, Q-Learning and Double Q-Learning

Finally, we make an overall comparison between SARSA, Q-learning and Double Q-learning algorithms, which are all forms of TD learning based on computing the next step prediction, and then updating the value estimates based on the next step prediction.





Our results show that SARSA and Q-learning both outperforms the Double Q-learning algorithm for the Cliff World MDP that we considered here.

3 Analysis of Double Q-Learning Algorithm

From our experiments above, we show that Double Q-learning performs almost as good as Q-learning under the MDPs considered. Even though the original motivation of double Q learning was to show that in some settings, it performs better than Q-learning due to its overestimation, our results here in the tabular setting in fact shows that Q-learning performs better does not produce overestimations of actions values.

In both Q-learning and double Q-learning, the goal is to produce an approximate estimate of the expected value function. Q-learning uses direct maximisation of the expected value function given by:

$$max_i E[X_i] = max_i E[\mu_i] \tag{1}$$

However, the approximation made by Q is an unbiased estimate. In double Q-learning, the algorithm uses a double estimator underestimates and the approximation is a weighted estimate of the unbiased expected values which is lower or equal to the maximum expected value.

Q-learning algorithm uses a single estimate for :

$$E[max_a Q_t(s_{t+1}, a)] \tag{2}$$

which in turn approximates $max_a E Q_t(s_{t+1}, a)$. Since the double Q learning algorithm uses two Q functions Q^A and QB, and each Q function is updated with the value from the other Q function.

Double Q learning has an unbiased estimate since for example, the update of Q^A depends on Q^B, and Q^B has already been updated with a different set of experience samples, so the estimate of Q^A can be considered as an unbiased estimate. The algorithm basically calculates an average of the two Q values for each action and performs greedy exploration based on the average of the Q values. The double estimator is therefore used to determine the value of the next state, using two different estimators, where one Q is updated with repsect to the unbiased updated estimate of the other Q function.

Based on our results above, we however show that the Double Q learning algorithm underestimates the action values for our given MDPs. However, it does not suffer from overestimation bias of the Q learning algorithm that it does under certain cases. Our results above also justify the fact that Q-learning is highly affected by the value of the discount factor, whereas Double Q learning is almost unaffected by this value.