HW4 Report

Y.S.S.V Sasi Kiran

November 25, 2018

1 Learning Curves & Hyperparameters

For gridWorld, we obtain the best hyperparameters as $\epsilon = 0.2$ and initial learning rate $\alpha = 0.5$ while using SARSA with $e-greedy$ action selection. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 20 epsiodes N is incremented by only one after each episode and after 20 episodes N is incremented for every SARSA update. We similarly found that $\epsilon \leftarrow \epsilon/N$, where N is incremented for every SARSA update. This is done for the GLIE convergence assumption. Optimistic Initialization with all q-values set to 4 was used. We obtain the learning curve as follows.

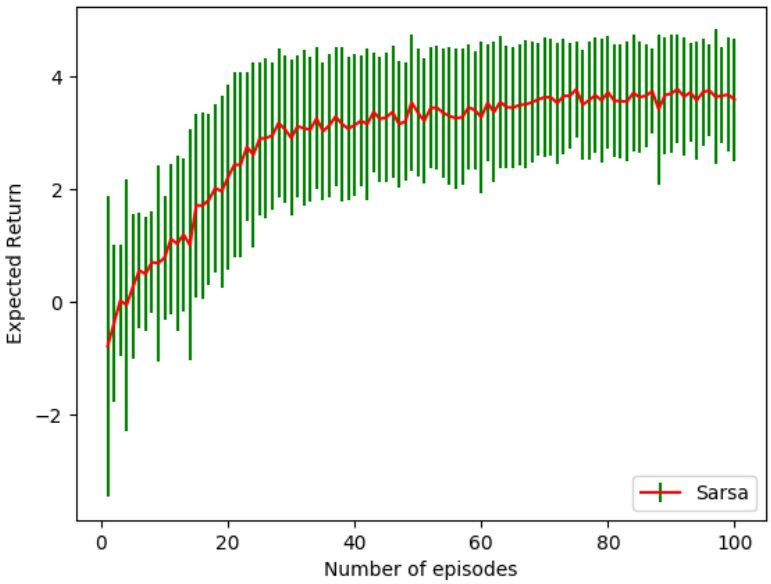

Figure 1.1: The learning curve for gridWorld with optimal hyperparameters and $e-$greedy action selection using SARSA. The standard deviation bars are colored in green.

Similarly for gridWorld domain with q-learning, we found that $\epsilon = 0.1$ and initial learning rate $\alpha = 0.5$ are the best hyperparameters with $e - greedy$ action selection. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 20 epsiodes N is incremented by only one after each episode and after 20 episodes N is incremented for every Q-learning update. Optimistic Initialization with all q-values set to 4 was used. We obtain the learning curve as follows.

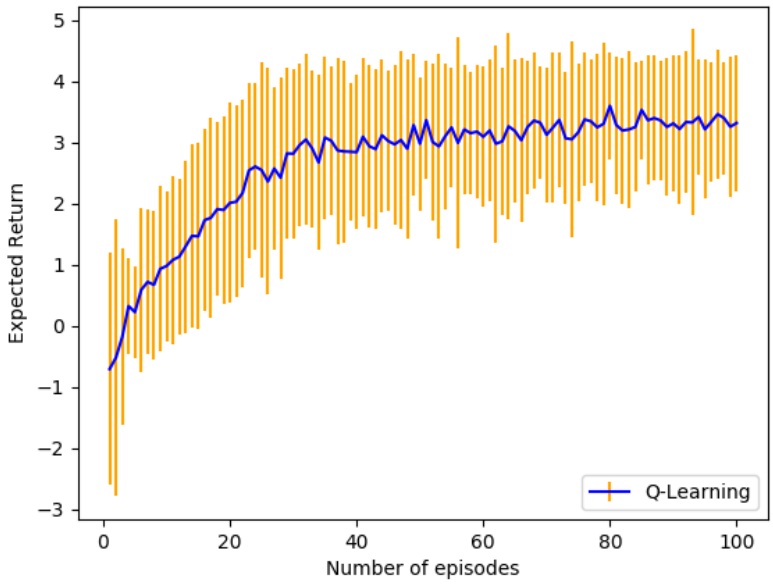

Figure 1.2: The learning curve for gridWorld with optimal hyperparameters and $\epsilon-$greedy action selection using q-learning. The standard deviation bars are colored in orange.

We combine both these curves into a single plot to compare SARSA and Q-learning for gridWorld domain. We sample error bars at some interval to avoid overlap. We obtain the figure shown below. We note that both of them have almost the same performance (AUC) with SARSA being slightly better at the convergence.

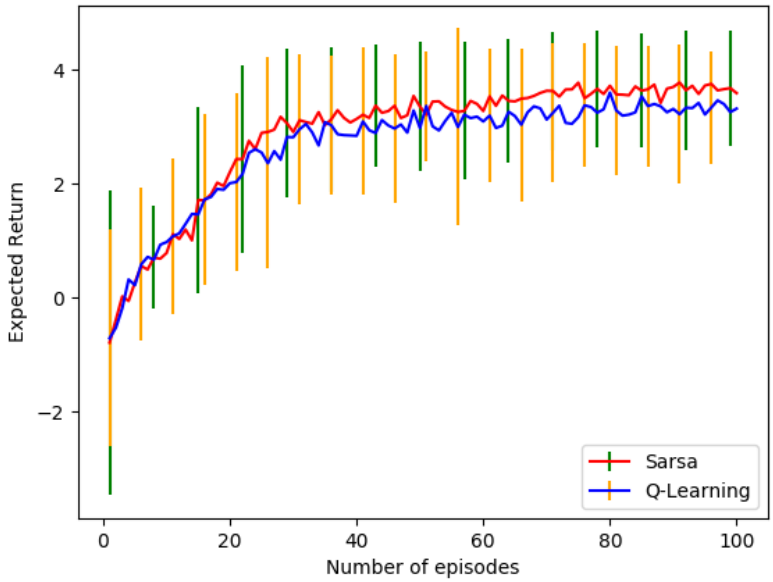

Figure 1.3: The learning curve for gridWorld for both SARSA and Q-learning.

For cartPole, we obtain the best hyperparameters as $\epsilon = 0.02$ and initial learning rate $\alpha = 0.002$ while using SARSA with $e - greedy$ action selection and fourier basis with $order = 5$. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 50 epsiodes N is unchanged i.e. $N = 1$ and after 50 episodes N is incremented for every SARSA update and with a maximum of 20 updates per episode. We similarly found that $\epsilon \leftarrow \epsilon/N$, where N is incremented for every SARSA update. This is done for the GLIE convergence assumption. All the weight vectors' components were initialized to 1. We obtain the learning curve as follows.

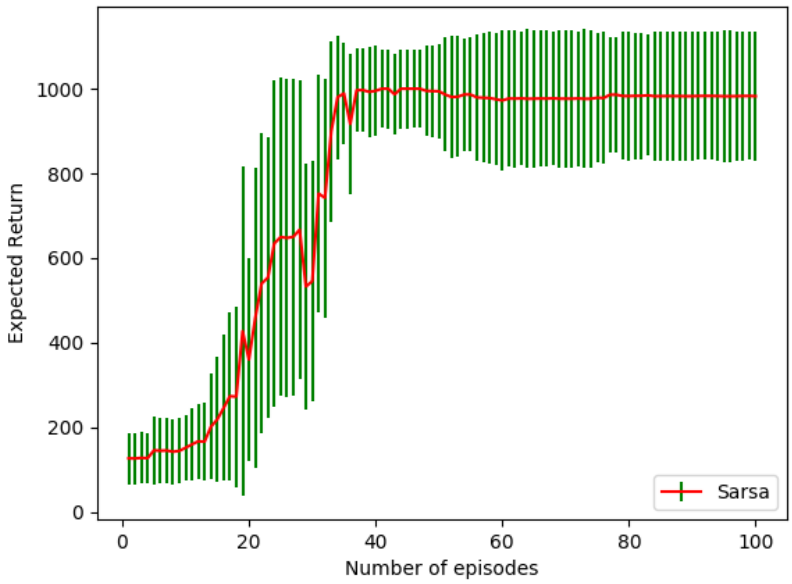

Figure 1.4: The learning curve for cartPole with fourier basis, optimal hyperparameters and $\epsilon-$greedy action selection using SARSA. The standard deviation bars are colored in green.

Similarly for cartPole domain with q-learning, we found that $\epsilon = 0.05$ and initial learning rate $\alpha = 0.002$ are the best hyperparameters with $e - greedy$ action selection and fourier basis of $order = 5$. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 50 epsiodes N is unchanged i.e. $N = 1$ and after 50 episodes N is incremented for every Q-learning update and with a maximum of 10 updates per episode. All the weight vectors' components were initialized to 1. We obtain the learning curve as follows.

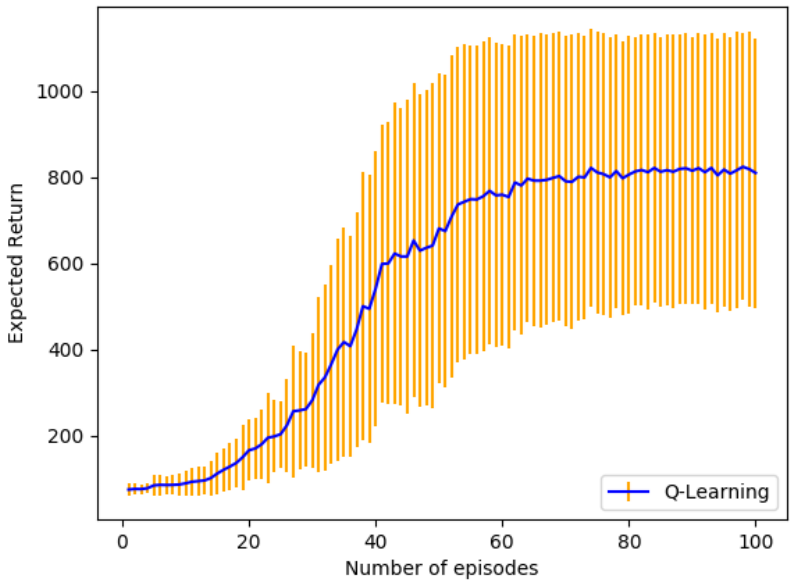

Figure 1.5: The learning curve for cartPole with fourier basis, optimal hyperparameters and $\epsilon-$greedy action selection using Q-learning. The standard deviation bars are colored in orange.

Though on average we converge to ~ 800, we reach optimum of 1010 on some trials for Q-learning as also shown below. This happens because Q-learning diverges for some trials also explaining the high variance.

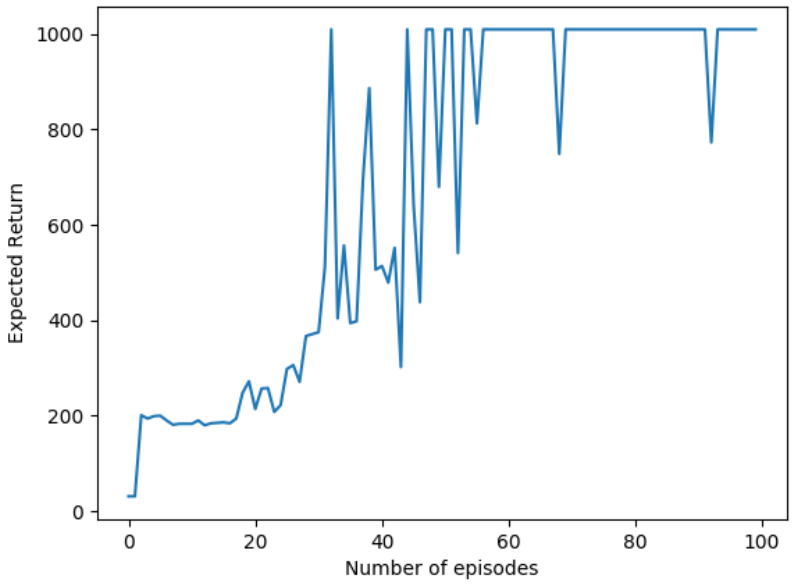

Figure 1.6: The learning curve for certain trial (one random seed) with same hyperparameters as above using Q-learning with fourier basis on CartPole. We observe the convergence to 1010 here.

We combine both these curves into a single plot to compare SARSA and Q-learning for cartPole domain. We sample error bars at some interval to avoid overlap. We obtain the figure shown below. We observe that SARSA converges very easily to max reward i.e. 1010 with very less deviation whereas Q-learning converges to ~ 800 on average only. This is because for some trials, Q-learning diverges resulting to the average coming down.

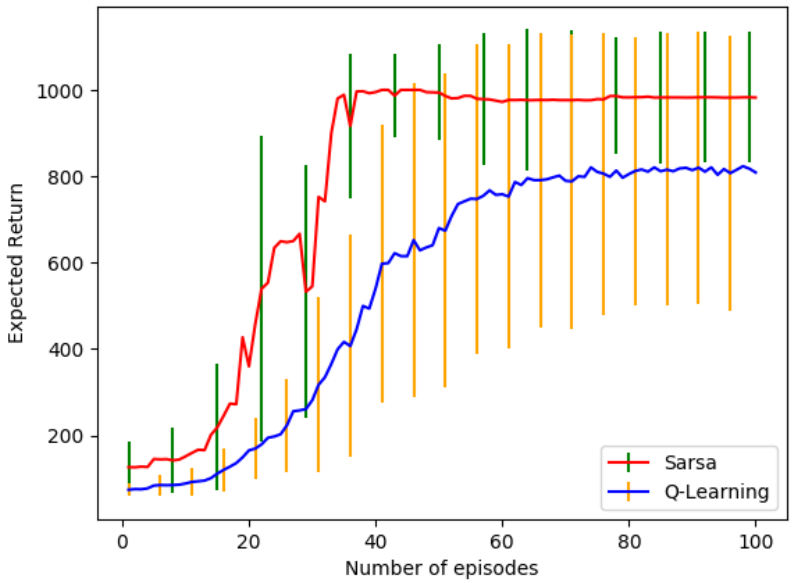

Figure 1.7: The learning curve for cartPole for both SARSA and Q-learning with fourier basis.

2 Alternate Function Approximator

2.1 Polynomial Basis

Polynomial basis uses the same expansion vectors as fourier basis and the ith term is given by $T_i = \prod_{j=1}^{n} (s_j)^{c_{i,j}}$, where s_j is the jth dimension of the state vector. We try to tune the CartPole domain for SARSA and Q-learning with polynomial basis and find it really hard to tune this. The main reason for this hardness is the scale of the expanded state. While fourier uses *cos*

where values lie in $[0, 1]$, the polynomial multiplies the state terms resulting in the values being very small easily with $order = 3$ and $order = 5$. Hence, for these orders, if the learning rate is small there will be no updates and if it is large, it diverges very easily. We obtain learning curves as shown below for single trials.

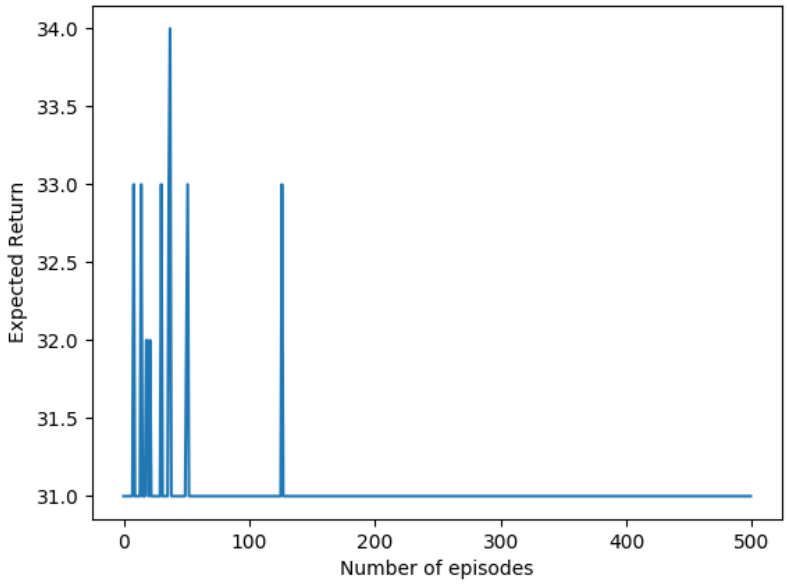

Figure 2.1: Example of learning curves for most of the hyperparameters with polynomial basis of order 3 and 5

Hence, we switch to $order = 1$ and $order = 2$. For order 1, we found that $\epsilon = 0.05$ and initial learning rate $\alpha = 0.05$ were the best using gridSearch. For order 2, we found that $\epsilon = 0.02$ and initial learning rate $\alpha = 0.0275$ were the best using gridSearch. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 50 epsiodes N is unchanged i.e. $N = 1$ and after 50 episodes N is incremented once for every epsiode. We similarly found that $\epsilon \leftarrow \epsilon/N$, where N is updated similarly as above. All the weights are initialized to 100. With these settings, we obtain the training curve as shown below.

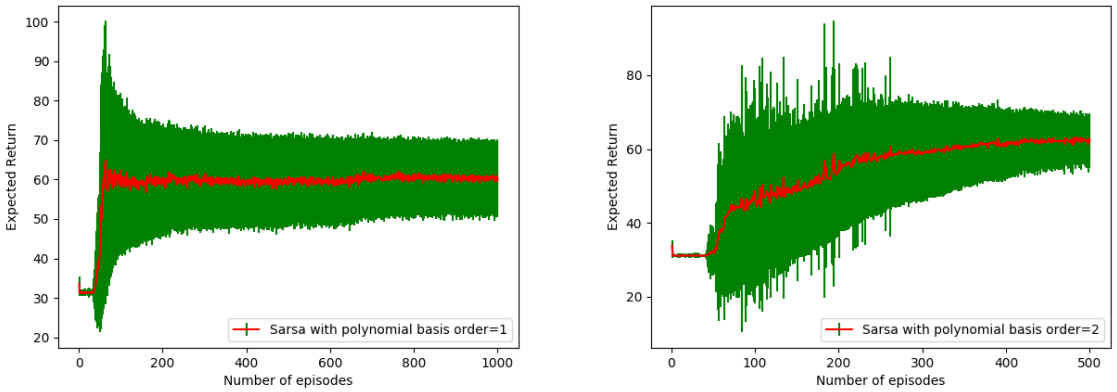

Figure 2.2: The learning curves obtained for cartPole domain with polynomial basis with order 1 and order 2 averaged over 100 trials using SARSA. The green area denotes standard deviation bars.

We observe that the results are still very poor compared to fourier basis. The main reason for hardness of optimization still is the fact that the expanded state has values which are very less. Hence, the parameters either don't get updated for smaller step size or diverge initially only for larger step sizes. Hence, the sweet spot for step size is almost negligible and is hence really hard to find too. Some trials atleast have learning reaching 200 return and are shown below. The algorithm doesn't stay at that top point due to these step size issues.

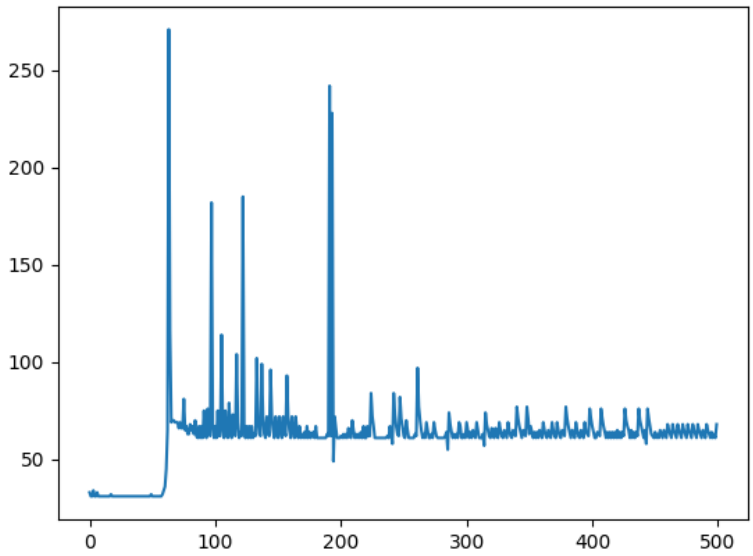

Figure 2.3: The learning curve for one of the trials for cartPole with polynomial basis using SARSA where the return has atleast explored 250. This illustrates the hardness of learning for polynomial basis compared to fourier basis. The X-axis is the number of episodes and Y-axis is the expected return.

Similarly, for Q-learning we found that $\epsilon = 0.01$ and initial learning rate $\alpha = 0.04$ are optimal for polynomial basis with order 2 through gridSearch. In the same way, we find $\epsilon = 0.01$ and initial learning rate $\alpha = 0.05$ are optimal through gridSearch for polynomial basis with order 1. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 50 epsiodes N is unchanged i.e. $N = 1$ and after 50 episodes N is incremented once for every epsiode. All the weights are initialized to 100. With these settings, we obtain the training curve as shown below.

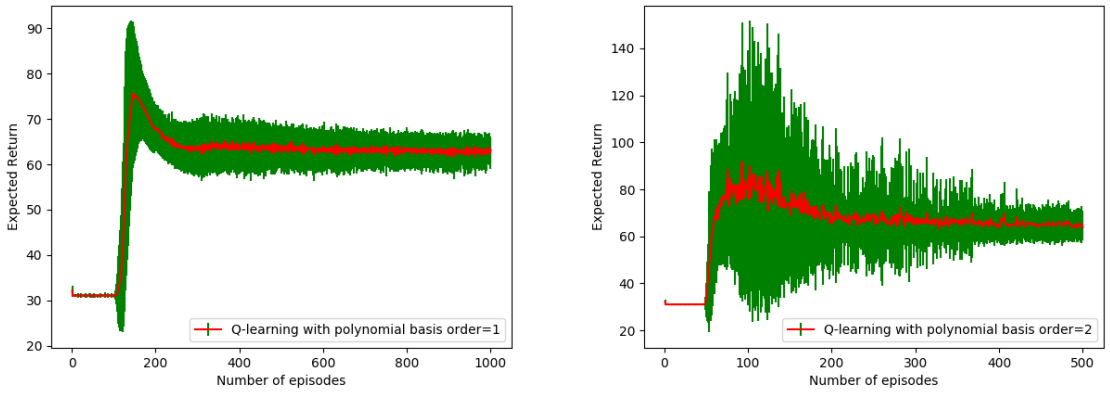

Figure 2.4: The learning curves obtained for cartPole domain with polynomial basis with order 1 and order 2 averaged over 100 trials using Q-learning. The green area denotes standard deviation bars.

2.2 Tile Basis

The tile basis consists of placing grids on the real state space where each grid is shifted by some amount from the original grid. Hence, the parameters consist of $tileWidth$ which defines the width of each tile in the grid, $tileShift$ which defines the the amount of unit shift for each shifted grid and $order$ which controls the number of such shifted grids. The c_i terms from fourier basis are used as the shift vectors in each direction and hence we will have $(order + 1)^4$ number of shifted grids.

For cartPole with SARSA and tile basis, we manually tune the parameters and find that $\epsilon = 0.015$, $\alpha = 0.0002$, $tileShift = 0.15$, $tileWidth = 0.25$ and $order = 2$ are the optimal parameters. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where N is incremented for every Q-learning update and with a maximum of 25 updates per episode. We similarly found that $\epsilon \leftarrow \epsilon/N$, where N is incremented for every SARSA update. We initialize all the weights to 1. We obtain the following learning curve for one of the best trials and the average curve.

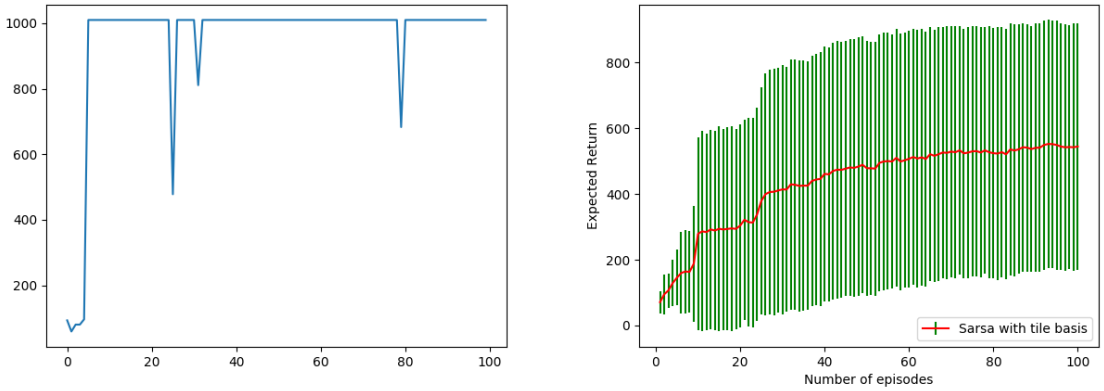

Figure 2.5: The learning curves obtained by running SARSA on cartPole. The left curve is obtained for some of the best trials while the right curve represents the average over 100 trials.

Similarly, for cartPole with Q-learning and tile basis, we manually tune and find that $\epsilon = 0.01$, $\alpha = 0.0001$, $tileShift = 0.15$, $tileWidth = 0.25$ and $order = 2$ are the optimal parameters. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where N is incremented for every Q-learning update and with a maximum of 10 updates per episode. We initialize all the weights to 5. We obtain the following learning curve for one of the best trials and the average curve.

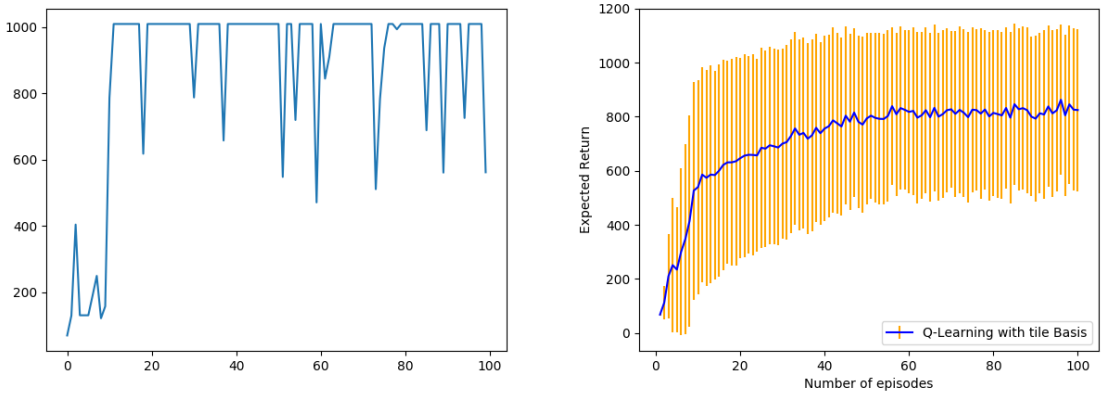

Figure 2.6: The learning curves obtained by running Q-learning on cartPole. The left curve is obtained for some of the best trials while the right curve represents the average over 100 trials.

Although the average results of tile coding are not as good as fourier basis, atleast some trials converge to 1010. Hence, it is better for cartPole domain compared to polynomial basis. Tile basis is very sensitive to the the hyperparameters like $tileWidth$, $tileShift$ and $order$. The main thing to note is that feature size is $2(order + 1)^4(1/tileWidth)^4$ which is very large and hence does not allow for larger $order$ and finer $tileWidth$ simultaneously. Also, the $tileShift$ should be such that the shifted tiles do not fall into exactly one tile of original grid and have overlap with multiple cells. Considering all these factors, this is also difficult to optimize compared to fourier basis which only has $order$ parameter.

3 Comparison with Cross Entropy

The below figures show the comparison of CE with all the other methods implemented in this assignment. We notice that Sarsa and Q-learning converge to optimal within 100 episodes unlike Cross-entropy. This is because in CE, we explore a population of policies evaluating each policy for few episodes to get an estimate of return. These additional computations make the number of episodes required to converge for CE to be large.

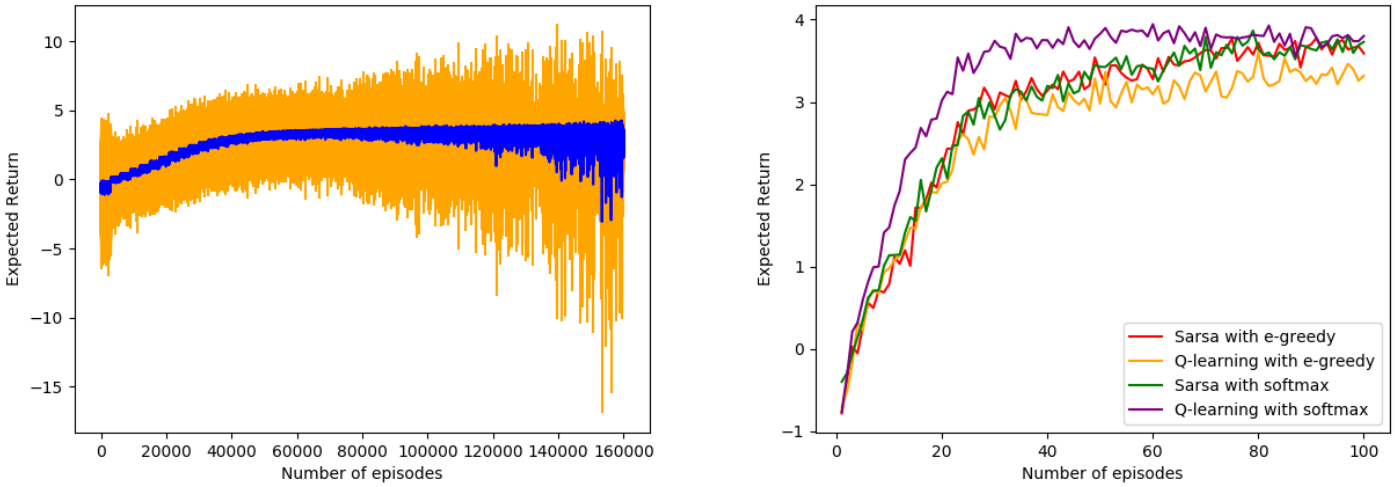

Figure 3.1: The above figure shows the comparison between Cross Entropy, Sarsa and Q-learning applied to gridWorld domain.

Similarly, we compare CE with all the other methods implemented for cartPole in the figure below. We again observe that Sarsa and Q-learning converge to optimal within 100 episodes for some basis like fourier. This is again because of the popuation exploration of CE.

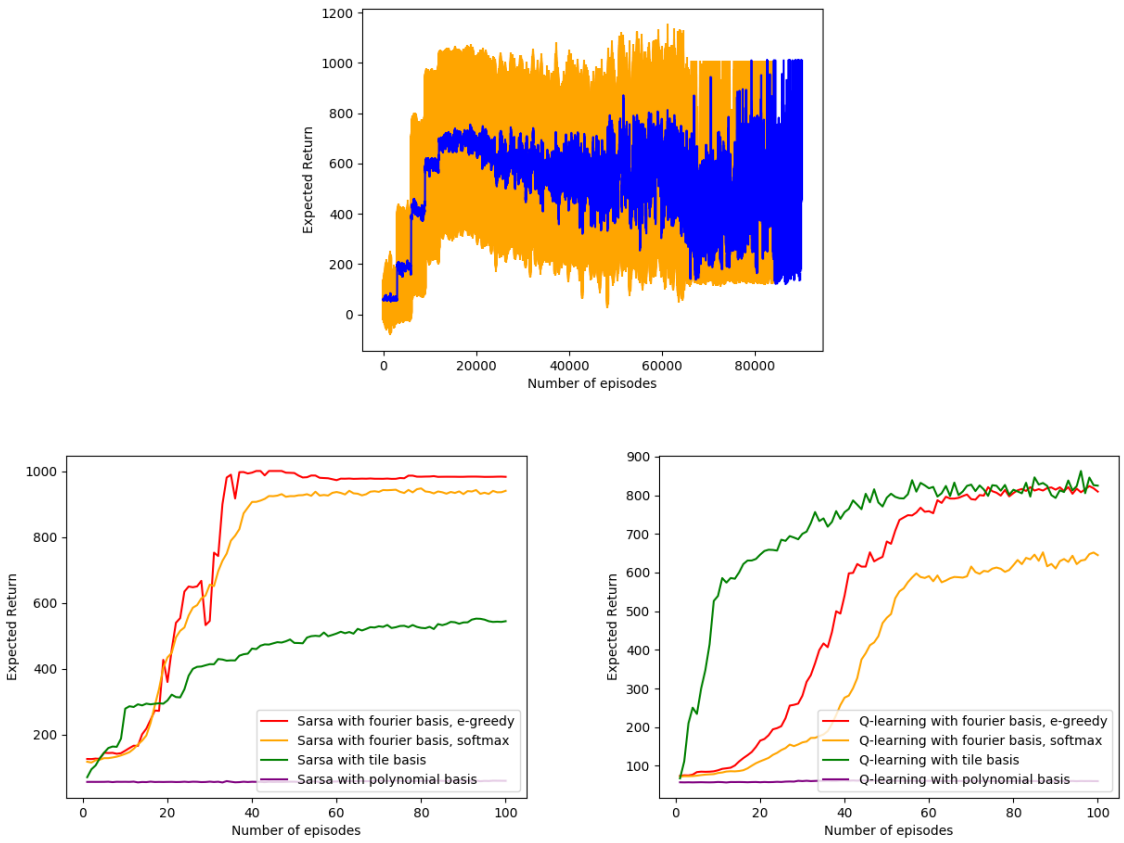

Figure 3.2: The above figure shows the comparison between Cross Entropy, Sarsa and Q-learning applied to cartPole domain with different basis

Both the methods were equally hard to tune, as both of them are sensitive to hyperparameters. But SARSA and Q-learning with fourier basis is slightly easier to tune as it has only ϵ and α which need to be tuned and hence has number of hyperparameters lesser than CE.

We have earlier noted (in hw3) that when the original state representation is very large i.e. it has large number of components, then the basis expanded state has even more components and hence the learning rate beyond which the algorithms (SARSA,

Q-learning) diverge will decrease. (This comes from divergence of TD) Hence, to optimize SARSA and Q-learning we need to have very small learning rates. But smaller learning rates lead to no or very small updates and hence we may not explore solutions and be stuck in local minima. Hence, in these cases Cross Entropy method is better as it does not have divergence issues.

When the state representation is small enough, then SARSA and Q-learning are preferred as they converge within hundreds of episodes while CE takes tens of thousands of episodes to converge. For example in gridWorld domain, SARSA and Q-learning are preferred over CE.

4 Softmax

Softmax action selection with temperature parameter σ has been implemented. For gridWorld, the optimal parameters have been found through gridSearch. For SARSA, the optimal hyperparameters found are $\sigma = 0.2$, $\alpha = 0.5$ and all the weights are initialized to 4. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 20 epsiodes N is incremented by only one after each episode and after 20 episodes N is incremented for every SARSA update. We similarly found that $\epsilon \leftarrow \epsilon/N$, where N is incremented for every SARSA update.

For Q-learning, the optimal hyperparameters are $\sigma = 0.02$, $\alpha = 0.75$ and all the weights initialized to 4. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 20 epsiodes N is incremented by only one after each episode and after 20 episodes N is incremented for every Q-learning update. We obtain the following learning curves.

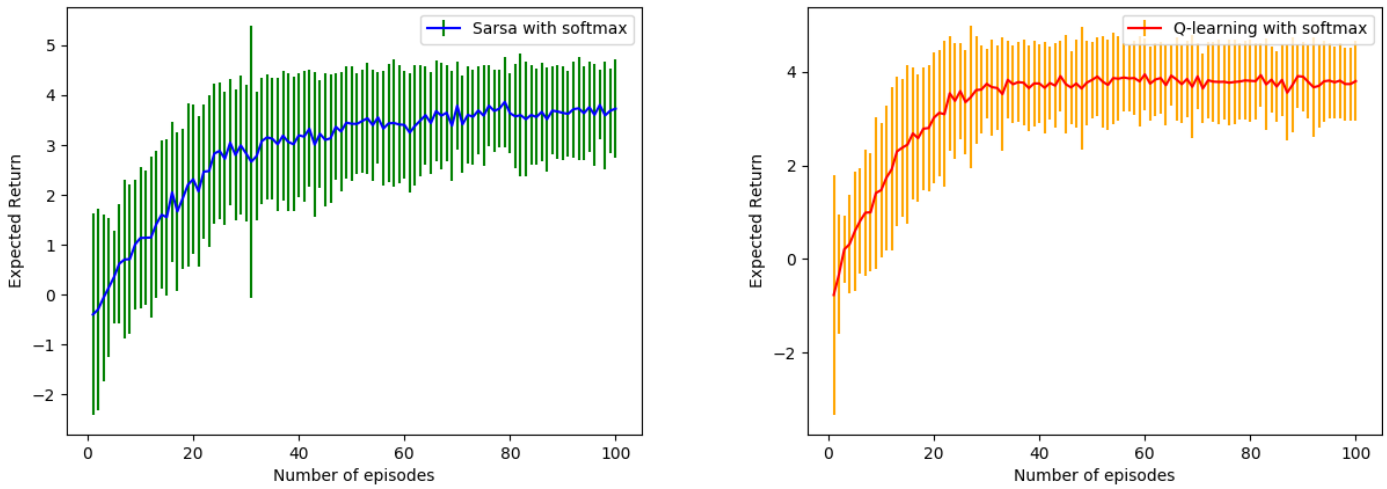

Figure 4.1: The above figure shows SARSA and Q-learning applied on gridWorld with softmax. The left curve is for SARSA while the right one is for Q-learning.

For CartPole, the optimal parameters have been found through gridSearch. For SARSA, the optimal hyperparameters found are $\sigma = 0.1$, $\alpha = 0.002$ and all weights are initialized to 1. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 40 epsiodes N is unchanged i.e. $N = 1$ and after 40 episodes N is incremented for every Q-learning update and with a maximum of 10 updates per episode. We similarly found that $\epsilon \leftarrow \epsilon/N$, where N is incremented for every SARSA update.

For Q-learning, the optimal hyperparameters are $\sigma = 0.1$, $\alpha = 0.00175$ and all weights are initialized to 1. For the step size routine, we found that $\alpha_{curr} \leftarrow \alpha/\sqrt{N}$ as the optimal decay process, where for the first 50 epsiodes N is unchanged i.e. $N = 1$ and after 50 episodes N is incremented for every Q-learning update and with a maximum of 5 updates per episode. We obtain the following learning curves.

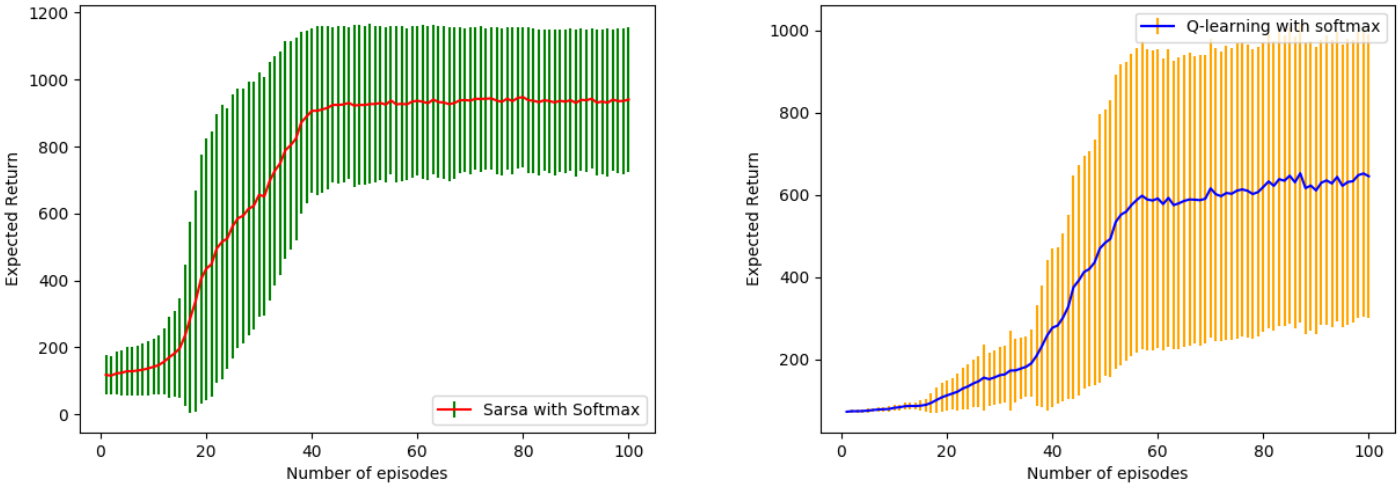

Figure 4.2: The above figure shows SARSA and Q-learning applied on cartPole with softmax. The left curve is for SARSA while the right one is for Q-learning.