Tetris

navidm

April 2021

Task 1a

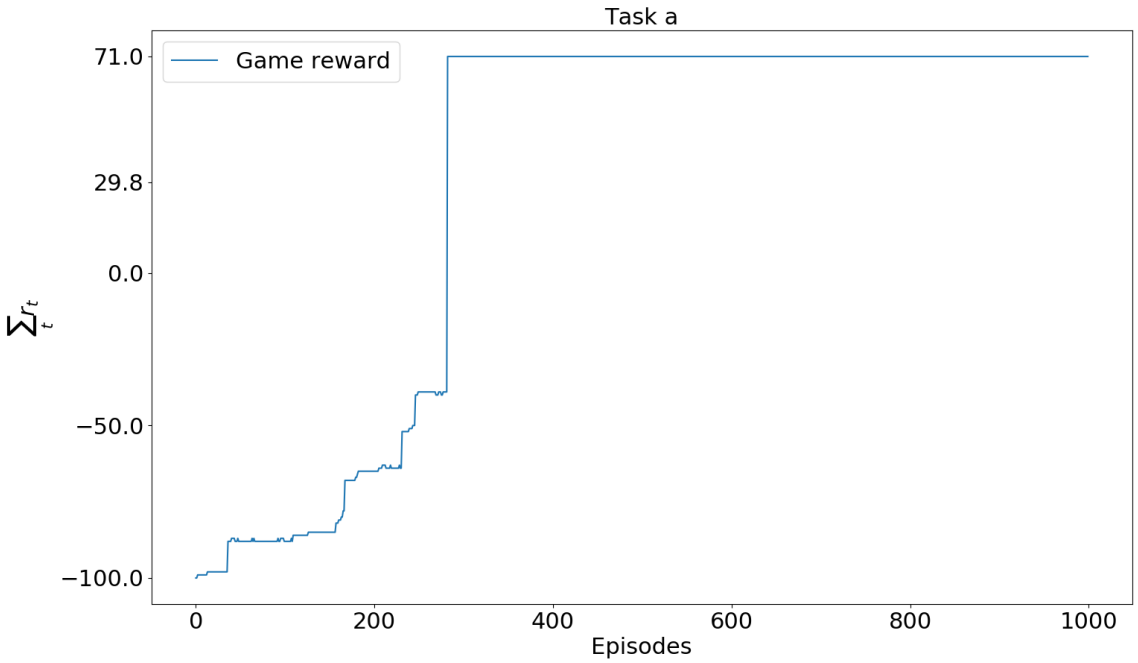


Figure 1: Training with deterministic sequence of tiles with $\epsilon = 0$

Task 1b

With ϵ-greedy training, agent will explore the action space better which leads to a higher reward sum at the end. Also there are fluctuations in the reward due to random non-optimal actions taken for the sake of exploration. This makes the average reward being smaller than case 1a, but the maximum reward obtained is higher.

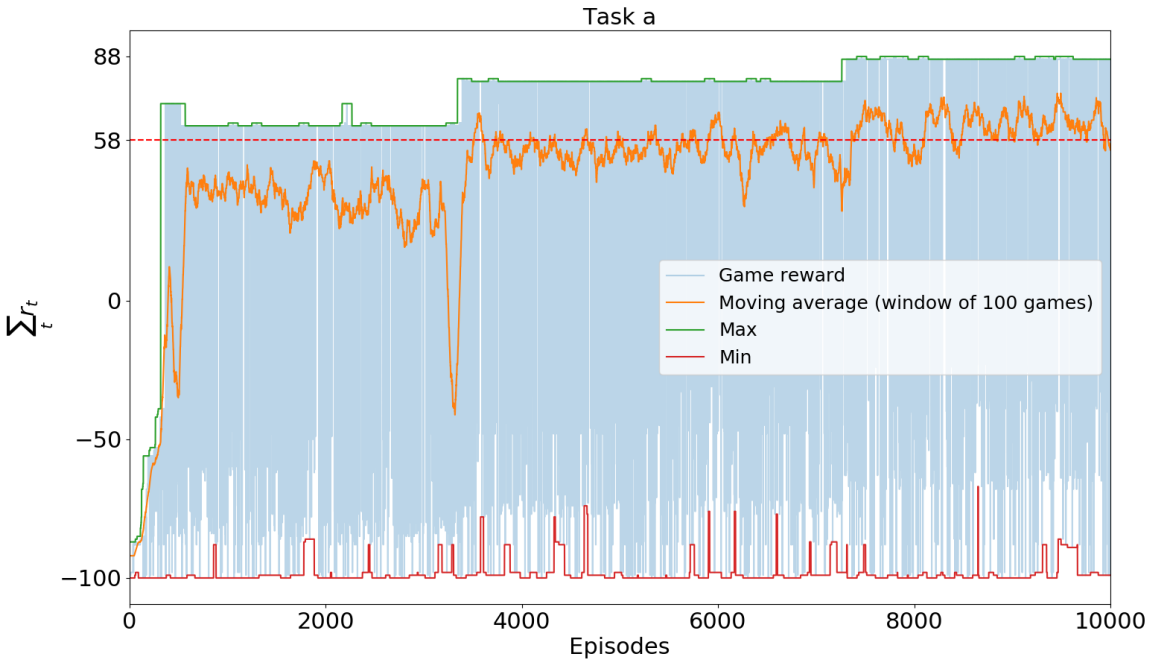
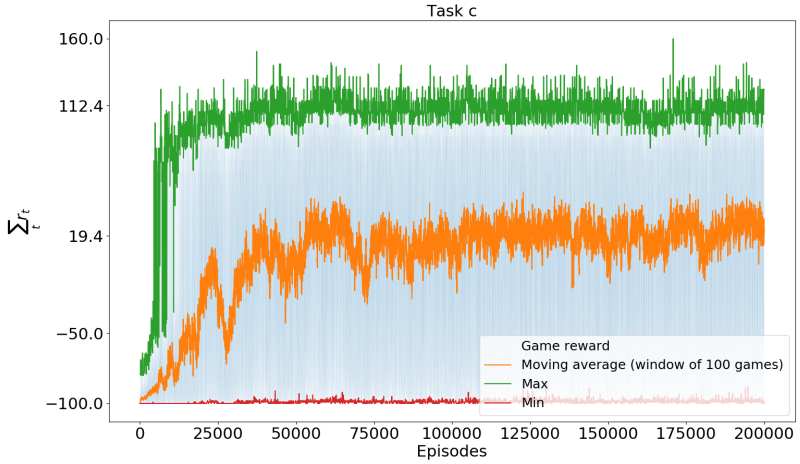


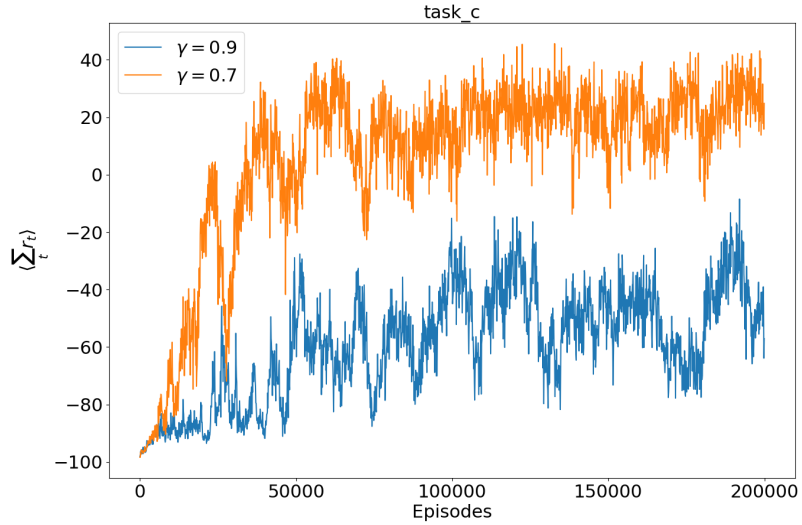
Figure 2: Training with deterministic sequence of tiles with $\epsilon = 10^{-3}$

Task 1c

In this case with γ close to 1, agent will not learn to finish a game without loosing using tabular Q-learning. This is because of large number of states that become possible due to stochastic tile sequence ($\mathcal{O}(2^{18})$). Therefor the reward sum will be always negative. However using smaller discount factor like $\gamma = 0.7$ will help the agent to optimize for a smaller time window which prevents from investing on risky combinations in order to obtain high rewards. In contrast, the agent will try to remove the rows earlier which gives at least a positive average reward sum, which is of course much smaller compared to the result from tasks a and b.



(a) Training with discount factor $\gamma = 0.7$.



(b) Comparison of the average rewards for $\gamma = 1$ and $\gamma = 0.7$

Figure 3: Rewards during training for stochastic tile sequence.

Task 1d

Having an 8×8 environment for the game and all the 7 classical Tetris tiles, with a rough approximation will give a state space of size $\mathcal{O}(7 \times 2^{64} \sim 10^{20})$, which will be impossible to do with tabular Q-learning considering the memory needed to save the Q-table. Even if we take into account that no row should be completely filled, the state space will have a size of $(2^8 - 1)^8 \sim 10^{19}$.

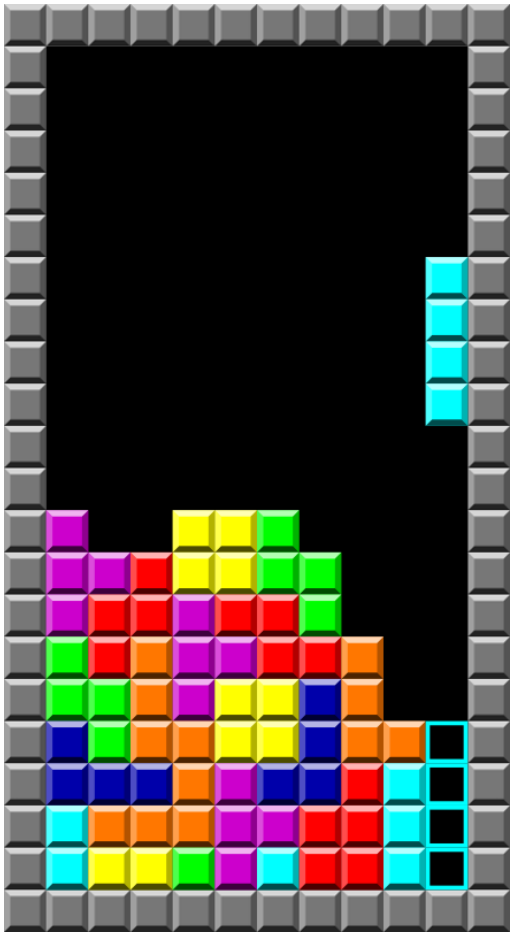
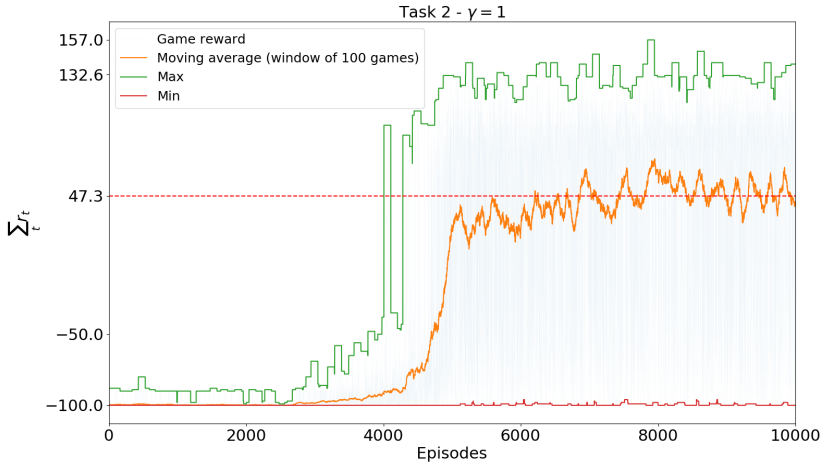


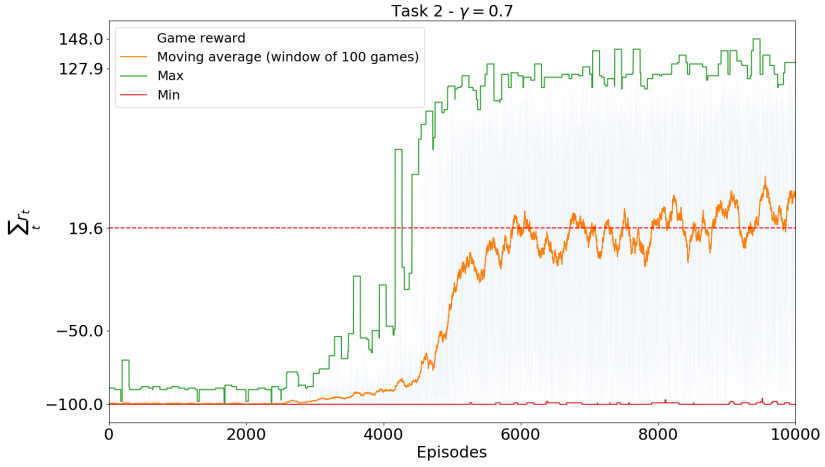
Figure 4: A typical Tetris game (Courtesy: Wikipedia)

Task 2a

Using the suggested architecture for the network, the Q-table were replaced with a function approximator that do not need the huge memory needed for the Q-table. In this way the agent can deal with very large state spaces like the classical Tetris with the environment size of 20×10. As a consequence, the agent can even predict a good action for the states that it has never seen before.
In contrast with task 1c, agent learns to obtain an average reward of 47.3, with an average maximum of 132.6 after training. Also using discount factor $\gamma = 0.7$ will yield almost the same result as tabular learning.



(a) Training with discount factor $\gamma = 0.7$.



(b)

Figure 5: Rewards during training for stochastic tile sequence.