Comp579 A2

Q1: Coding

We first present the final training and testing performance in Figure 1 and Figure 2. The results are obtained following the assignment instructions with 10 independent trials each has 500 segments. For each segment, there are 10 episodes of training, followed by 1 testing episode.

For the training performance, we demonstrate that in temperatures 0.1 and 1, the performance of both Sarsa and Expected Sarsa are robust to a wide range of hyperparameters space of learning rates, i.e., [0.1, 1]. But the temperature at 3 shows worse performance with bigger variance in reward. The negative impact of a higher temperature on the performance of Sarsa and Expected Sarsa is primarily due to the increased emphasis on exploration over exploitation, leading to more frequent selection of suboptimal actions, difficulty in policy convergence, and higher sensitivity to other hyperparameters, resulting in worse performance and bigger variance in reward.

We also find that those two algorithms achieve worse performance in extremely low learning rates, such as 0.01 and high learning rates greater than 1. The algorithms exhibit poorer performance at extremely low learning rates (e.g., 0.01) because the updates to the value function are too small, leading to slow learning progress. Conversely, at high learning rates (greater than 1), the value function may overshoot the optimal values, causing instability and convergence issues, which also results in suboptimal performance.

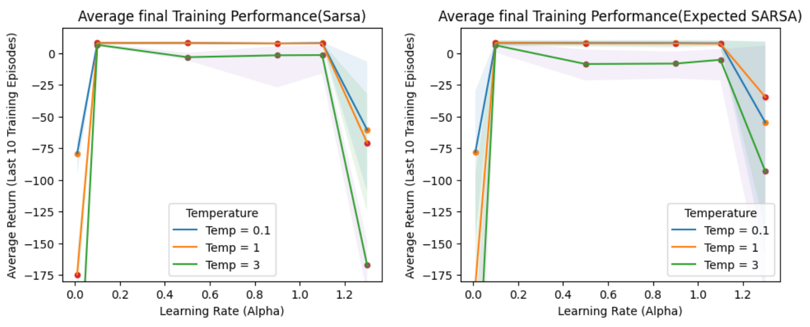

Figure 1: Final training performance Sarsa vs. Expected Sarsa.

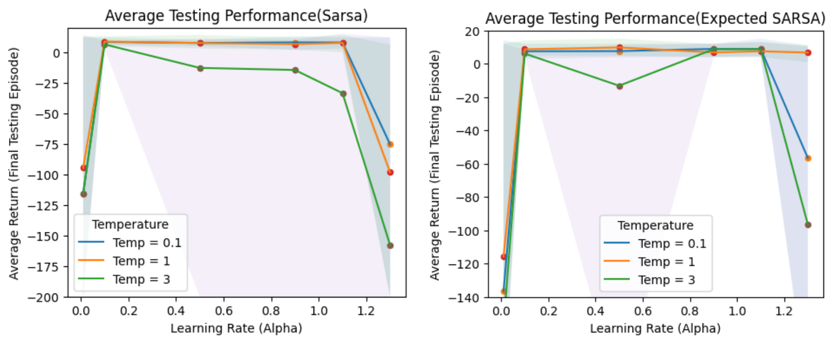

Figure 2: Final testing performance Sarsa vs. Expected Sarsa.

For the final testing performance, we demonstrate the consistent phenomenon to the final training performance. However, when the learning rate equals 0.5, we find that temp=3 will result in significantly lower performance regarding the expectation. Additionally, temp=3 exhibits high variance in the comparison to the other two options. From this ablation study, we are encouraged to choose a wide range of hyperparameters regarding the learning rate, i.e., greater than 0.1 within low temperature, i.e., smaller than 1. We chose our optimal hyperparameter combination of the learning rate and temperature as $(0.5, 0.1)$.

We present the learning curves of Sarsa and Expected Sarsa in Figure 3. In the Taxi Problem, Sarsa learns the optimal policy more rapidly than Expected-Sarsa, indicating it can more quickly adapt its strategy to maximize rewards. However, once both algorithms have converged to their optimal policies, Expected-Sarsa shows a more consistent performance with less fluctuation in its results, suggesting it is more stable in maintaining the learned policy under varying conditions or episodes.

Q2: Bellman equation and dynamic programming

a

The Bellman equation for an action-value function $q^\pi(s, a)$ under a policy π is given by:

$$q^\pi(s, a) = \mathbb{E}[r_{t+1} + \gamma q^\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s, A_t = a]$$

The action-value functions for r_1 and r_2 under policy π are defined as:

$$q_1^\pi(s, a) = \mathbb{E}[r_{1,t+1} + \gamma q_1^\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s, A_t = a]$$

$$q_2^\pi(s, a) = \mathbb{E}[r_{2,t+1} + \gamma q_2^\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s, A_t = a]$$

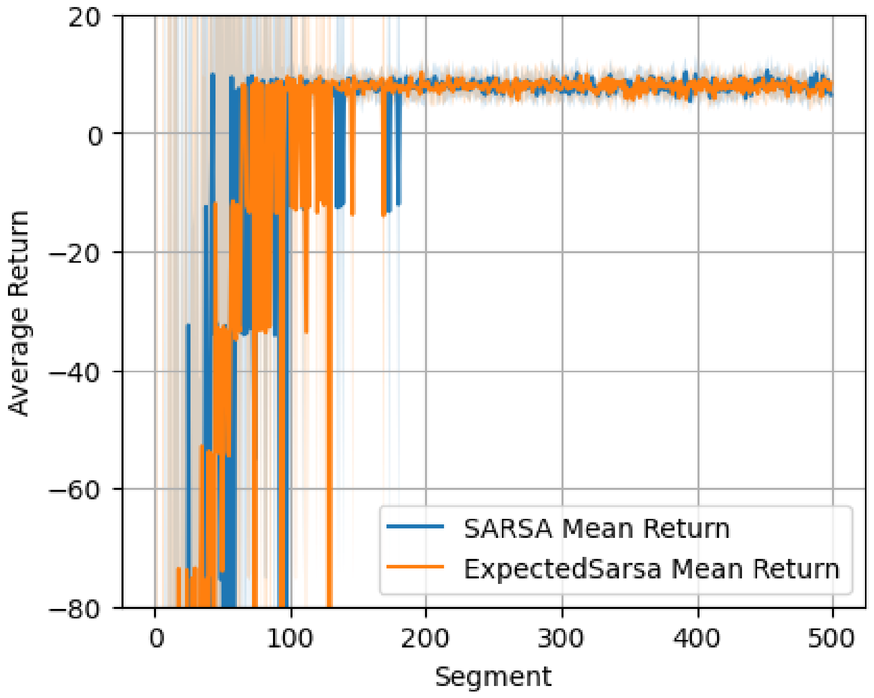

Figure 3: Learning curves.

We combine the two value functions of the form:

$$q^{\pi}(s,a) = \alpha q_1^{\pi}(s,a) + \beta q_2^{\pi}(s,a)$$

thus:

$$q^{\pi}(s,a) = \alpha\mathbb{E}[r_{1,t+1} + \gamma q_1^{\pi}(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a] + \beta\mathbb{E}[r_{2,t+1} + \gamma q_2^{\pi}(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$$

This simplifies to:

$$q^{\pi}(s,a) = \mathbb{E}[\alpha r_{1,t+1} + \beta r_{2,t+1} + \gamma(\alpha q_1^{\pi}(S_{t+1}, A_{t+1}) + \beta q_2^{\pi}(S_{t+1}, A_{t+1})) \mid S_t = s, A_t = a]$$

Given $r_{t+1} = \alpha r_{1,t+1} + \beta r_{2,t+1}$ and $q^{\pi}(s_{t+1}, a_{t+1}) = \alpha q_1^{\pi}(s_{t+1}, a_{t+1}) + \beta q_2^{\pi}(s_{t+1}, a_{t+1})$, we can rewrite the equation as:

$$q^{\pi}(s,a) = \mathbb{E}[r_{t+1} + \gamma(q^{\pi}(s_{t+1}, a_{t+1})) \mid S_t = s, A_t = a]$$

This demonstrates that, under the assumption of linear combination of rewards, the action-value functions can also be combined linearly with the same coefficients to solve the MDP for the combined reward function.

b

The Bellman optimality equation for an action-value function $q^*(s,a)$ is given by:

$$q^*(s,a) = \mathbb{E}[r_{t+1} + \gamma \max_{a'} q^*(S_{t+1}, a') \mid S_t = s, A_t = a]$$

The optimal action-value functions under an optimal policy π^* for r_1 and r_2 are:

$$q_1^*(s,a) = \mathbb{E}[r_{1,t+1} + \gamma \max_{a'} q_1^*(S_{t+1}, a') \mid S_t = s, A_t = a]$$

$$q_2^*(s,a) = \mathbb{E}[r_{2,t+1} + \gamma \max_{a'} q_2^*(S_{t+1}, a') \mid S_t = s, A_t = a]$$

We examine if a linear combination $q^*(s,a) = \alpha q_1^*(s,a) + \beta q_2^*(s,a)$ can yield the optimal action-value function for the combined reward function. Substituting the definitions of q_1^* and q_2^* into this proposition results in:

$$q^*(s,a) = \alpha\mathbb{E}[r_{1,t+1} + \gamma \max_{a'} q_1^*(S_{t+1}, a') \mid S_t = s, A_t = a] + \beta\mathbb{E}[r_{2,t+1} + \gamma \max_{a'} q_2^*(S_{t+1}, a') \mid S_t = s, A_t = a]$$

this simplifies to:

$$q^*(s,a) = \mathbb{E}[r_{t+1} + \gamma(\max_{a'} q_1^*(S_{t+1}, a') + \max_{a'} q_2^*(S_{t+1}, a')) \mid S_t = s, A_t = a]$$

However, The challenge in linearly combining q_1^* and q_2^* lies in the max operator used in the Bellman optimality equation. The optimal action maximizing q_1^* may not be the same as that maximizing q_2^*, indicating that $\max_{a'} q^*(S_{t+1}, a')$ cannot be directly decomposed into $\alpha \max_{a'} q_1^*(S_{t+1}, a') + \beta \max_{a'} q_2^*(S_{t+1}, a')$ due to the non-linearity of the max operator. Hence, it is impossible to linearly combine q_1^* and q_2^* to achieve q^* for the combined reward function r.

Q3: An alternative learning algorithm

a

This learning algorithm exhibits characteristics of both on-policy and off-policy methods due to its specific target formula for updating the Q-function:

$$R + \gamma((1-\alpha)\max_a Q(s', a) + \alpha \sum_a \pi(s', a) Q(s', a)) \tag{1}$$

- **On-Policy Component**: The term $\alpha \sum_a \pi(s', a)Q(s', a)$ incorporates the expected return under the current policy π, which is an ϵ-greedy policy derived from Q. This part of the target makes the learning process partly on-policy since it directly uses the current behavior policy (π) to influence the Q-value updates.
- **Off-Policy Component**: The term $(1-\alpha)\max_a Q(s', a)$ represents the maximum expected return from the next state s' under any policy, not necessarily the one currently being followed. This characterizes off-policy learning, where the policy used to generate behavior (the exploration policy) is different from the policy being evaluated and improved (the greedy policy, in this case).

Given the mixture of these components, the algorithm cannot be strictly categorized as either on-policy or off-policy. Instead, the hyper-parameter α balances the learning from the current policy's expected returns (on-policy characteristic) and the best possible actions regardless of the current policy (off-policy characteristic).

b

The degree to which the algorithm is on-policy or off-policy is modulated by the hyper-parameter α. When α is closer to 1, the learning process leans more towards being on-policy, emphasizing the current policy's expected returns. Conversely, when α is closer to 0, it behaves more like a traditional off-policy learner, focusing on the maximum possible returns from any action.

- **Bias**:
  - Low α: More like Q-learning, with less bias towards the current policy but may inherit maximization bias.
  - High α: More characteristics of SARSA, potentially increasing bias with suboptimal policies but aligning closer to the expected returns under the current policy.
- **Variance**:
  - Low α: Higher variance due to the maximization process.

  - High α: Reduced variance by averaging over the actual distribution of actions taken.

- **Maximization Bias**:
  - With higher α, the algorithm reduces maximization bias by relying more on the current policy's average returns.
  - Lower α values maintain a risk of maximization bias, similar to Q-learning.

c

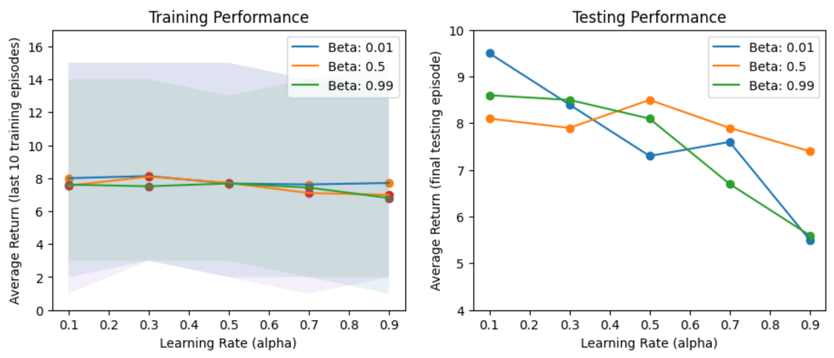

Figure 4: Final training performance for different α in Equation 1. Without abusing of notations, we use β to represent α in Equation 1.

We show that there is no clear comparison of the final training performance with varying α in Equation 1 under different learning rates regarding the final training performance. However, there is an explicit down-stream pattern with increasing learning rates in terms of the final test performance. In large learning rates, i.e., $[0.5, 0.9]$, $\alpha = 0.5$ achieves better performance than two other options, which demonstrates the benefits of the moderate choice of α, while the learning rate is near-optimal in the default setting. In this case, it both aims to pay attention to the bias and variance reduction and thus achieve the best final testing performance.

However, from the training performance in Figure 4 (left), we cannot find the alignment phenomenon regarding our prior statements of variances and bias over different α. We hypothesize the reason for that is the task is easy compared to other tasks, which are sufficient to represent the variance and bias over different α.

All Code in Assignment 2 is avaiable at this link.

appendix

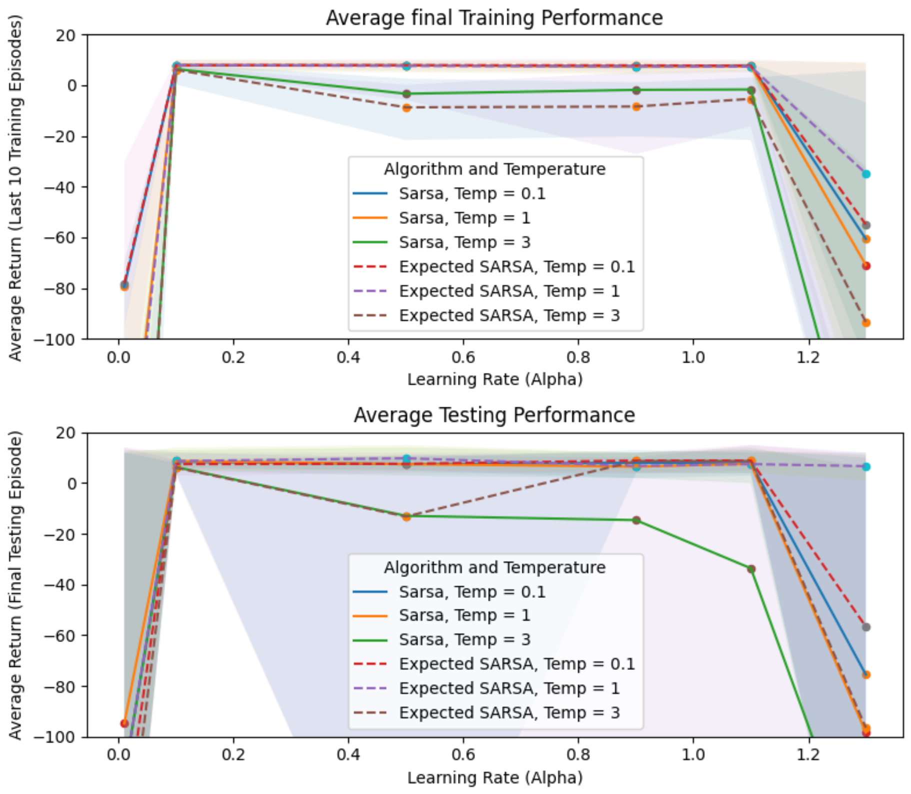

Figure 5: performance plot combined 2 algorithms