Principles of Robot Autonomy II
Exam 1 with Solutions
February 10, 2023

Name:

SUNet ID:

Instructions:

- Time allowed: 60 minutes.
- Total Points: 48.
- The exam consists of **four** equally weighted problems, with equally weighted subproblems.
- Please read all questions carefully before answering. Correct answers will receive full credit.
- **To get partial credit for an incorrect answer, you should explain your reasoning.** Please write all your answers in the provided answer sheet.

Good luck!

1. **Supervised Learning (12 points)**

(i) To speed up computation, stochastic gradient descent updates the model parameters with the gradient of the loss on a random *subset* of the training set at each iteration.

True **False**

Explain:

(ii) Consider a perceptron with a weight W and a bias b. Given that the input is $x \in \mathbb{R}^N$ and the output is $y \in \mathbb{R}^{10}$, the number of parameters that need to be learned is $N \times 10 + 10$.

True **False**

Explain:

(iii) You find after training a two-layer neural network classifier that test set accuracy is lower than the training set accuracy. Which one of the following could help improve test set performance? (Select one option)

Switch to a mean squared error training loss

Increase the regularization weight

Remove the final softmax layer as this is a nonlinear function

Explain:

Solutions:

(i) True. The gradient of the loss on the full training set can be expensive to compute, especially if the training set is large. Instead, stochastic gradient descent approximates the gradient with a small batch of data.

(ii) True. From each input to each output, there is a weight parameter. This makes $N \times 10$. For each output, there is also a bias term. In sum, the number of parameters that need to be learned is $N \times 10 + 10$.

(iii) B. The problem is overfitting. Increasing the regularization weight could help with overcoming this problem.

2. **Dynamic Programming (12 points)**

(i) Consider the *deterministic* decision making problem with the system model $x_{k+1} = f_k(x_k, u_k)$ for $k = 1, \dots, N-1$ and cost functions g_k for $k = 1, \dots, N$,

$$J^*(x_0) = \min_{u_k \in U_k, k=0,1,\dots,N-1} g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k).$$

Starting from x_0, let $\{u_0^*, u_1^*, \dots, u_{N-1}^*\}$ be an optimal control sequence that generates the resulting optimal state sequence $\{x_0^*, x_1^*, \dots, x_N^*\}$. Then, if starting from x_k^*, the tail sequence $\{u_k^*, \dots, u_{N-1}^*\}$ is also an optimal control sequence that minimizes the cost from time k to N.

True **False**

Explain:

(ii) Even in a discrete MDP with a finite state space and finite action space, policy iteration may *not* terminate in a finite number of iterations.

True **False**

Explain:

(iii) Consider a Markov decision process for the TurtleBot robot where there are four candidate actions at each state (i.e. $|\mathcal{A}| = 4$): LEFT, RIGHT, FORWARD, BACKWARD. If there are ten states in this environment (i.e. $|\mathcal{S}| = 10$), then what is the number of deterministic policies for this problem?

4

40

4^{10}

Explain:

Solutions:

(i) True. This is the principle of optimality for deterministic systems. Suppose there exists a different control sequence $\{\tilde{u}_k^*, \dots, \tilde{u}_{N-1}^*\}$ such that $J(x_k, \tilde{u}_k^*, \dots, \tilde{u}_{N-1}^*) < J(x_k, u_k^*, \dots, u_{N-1}^*)$. Then, $J(x_0, u_0^*, \dots, u_{k-1}^*) + J(x_k, \tilde{u}_k^*, \dots, \tilde{u}_{N-1}^*) < J(x_0, u_0^*, \dots, u_{k-1}^*) + J(x_k, u_k^*, \dots, u_{N-1}^*) = J^*(x_0)$, which is a contradiction.

(ii) False. Each iteration is an improvement over the previous policy and there is a finite number of policies.

(iii) The number of deterministic policies for an MDP is $|\mathcal{A}|^{|\mathcal{S}|}$, so 4^10 for this problem.

3. **Learning-based Control (12 points)**

(i) For a given state x, the optimal tabular Q-function value, $Q^*(x,u)$, evaluated at the optimal action u^* has exactly the same numerical value as the optimal value function, $V^*(x)$, at that state.

True **False**

Explain:

(ii) In Q-learning, each time we update our estimate of the Q function we **must** use an ϵ-greedy policy based on the current estimate of the Q function to collect data.

True **False**

Explain:

(iii) Consider a Markov decision process (MDP) with two states $\{A,B\}$ and two actions $\{0,1\}$. Suppose we have been applying value iteration on this MDP for k iterations and obtained $V_k^*(A)=10$, $V_k^*(B)=4$. Given the reward values and transition probabilities below, perform one more iteration of the value iteration algorithm with a discount factor of $\gamma=0.8$ to find V_{k+1}^* values.

$R(A,0)=1, \quad R(A,1)=2, \quad R(B,0)=2, \quad R(B,1)=2$
$P(A \mid A,0)=1, \quad P(A \mid A,1)=0.5$
$P(A \mid B,0)=0, \quad P(A \mid B,1)=0.5$

What is $V_{k+1}^*(A)$?

8
9
10
11
12

Explain:

Solutions:

(i) True. The optimal action Q-value is equivalent to the value function value at every state.

(ii) False. Q-learning is *off-policy*, and can be applied to transitions collected from an arbitrary policy, not necessarily from a policy corresponding to the current estimate of the Q function.

(iii) The answer is 9, because

$$\begin{aligned} V_{k+1}^*(A) &= \max \big\{ R(A,0) + \gamma \left[V_k^*(A) P(A \mid A,0) + V_k^*(B) P(B \mid A,0) \right], \\ &\qquad R(A,1) + \gamma \left[V_k^*(A) P(A \mid A,1) + V_k^*(B) P(B \mid A,1) \right] \big\} \\ &= \max \left\{ 1 + 0.8 \left[10 + 0 \right], 2 + 0.8 \left[5 + 2 \right] \right\} \\ &= 9 \end{aligned}$$

4. **Learning-based Perception (12 points)**

    (i) A filter in a convolutional layer can be written as an affine function $f(x) = w^\top x + b$, where w is the vectorized representation of the weight parameters, x is the vectorized version of image pixels covered by the filter, and b is a scalar bias term.

    True **False**

    Explain:

    (ii) Pooling layers generally lead to an increase in the number of parameters in a CNN.

    True **False**

    Explain:

    (iii) Consider a CNN that was trained to classify whether the input image contains a cat. Given an image containing a cat, we can predict the position of the cat in the image by applying the CNN to smaller sections of the image.

    True **False**

    Explain:

Solutions:

(i) True. The convolution operation is a dot product between the weight parameters of the filter and the image pixels covered by the filter, followed by addition of the scalar bias term.

(ii) False. Pooling layers do not introduce any new parameters, and they often decrease the dimensionality of data, leading a decrease in the number of parameters in the following layers.

(iii) True. This type of approach, called sliding window, applies a pre-trained CNN on smaller sections ("windows") of the image.