

Exam Advanced Machine Learning

7 January 2020, 18.30–21.15

This exam consists of 6 problems, each consisting of several questions. All answers should be motivated, including calculations, formulas used, etc. The use of a calculator is not allowed.

Question 1: Short questions

Please provide an argument for your answer on the following questions.

- (a) Is the following statement true or false? A convolutional neural network (CNN) for image analysis can be trained for unsupervised learning tasks, whereas an ordinary neural network cannot.
- (b) Is the following statement true or false? A perceptron model can achieve zero training error on *any* linearly separable dataset.
- (c) How does the bias-variance decomposition of a ridge regression estimator compare with that of ordinary least squares regression?
- (d) Is the following statement true or false? Logistic regression is equivalent to a neural network without hidden units and using the cross-entropy loss.
- (e) What are some practical problems with the sigmoidal activation functions in neural networks?

Question 2: Neural networks

- (a) Suppose that you build a neural network for a specific purpose. You train your network with cost function $J = \frac{1}{2} \|\mathbf{y} - \mathbf{z}\|^2$.
 - When the input \mathbf{x} is given to the network, first a weight matrix V mapping the input layer to the hidden layer is applied to yield $\mathbf{g} = V\mathbf{x}$.
 - The vector of hidden unit values \mathbf{g} are then activated using a ReLU activation function r , yielding $\mathbf{h} = r(\mathbf{g})$.
 - Finally, the activated values are transformed to the output \mathbf{z} by a weight matrix W given by $\mathbf{z} = W\mathbf{h}$.

Derive $\partial J / \partial W_{ij}$ and $\partial J / \partial V_{ij}$ for this network.

- (b) Suppose that you have a 3-dimensional input $\mathbf{x} = (x_1, x_2, x_3) = (2, 2, 1)$ fully connected to 1 neuron with activation function g_i . The forward propagation can be written as

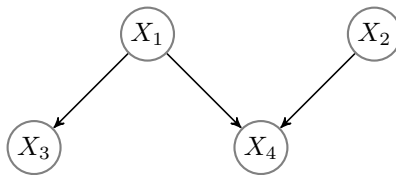
$$a_i = g_i(w_1x_1 + w_2x_2 + w_3x_3 + b).$$

After training this network, the values of the weights and bias are $\mathbf{w} = (w_1, w_2, w_3) = (0.5, -0.2, 0)$ and $b = 0.1$. You try 4 different activation functions g_1, \dots, g_4 , which respectively output the values $a_1 = 0.67$, $a_2 = 0.70$, $a_3 = 1.0$, and $a_4 = 0.70$. What is a valid guess for the activation functions g_1, \dots, g_4 ? You can choose for each activation function from the set of sigmoid, tanh, linear, ReLU, leaky ReLU, and indicator functions.

- (c) Explain what effect the following operations generally have on the bias and variance of your model.
- (1) Regularizing the weights;
 - (2) Increasing the size of the layers (more hidden units per layer);
 - (3) Using dropout to train a deep neural network;
 - (4) Getting more training data (from the same distribution as before).

Question 3: Graphical models

The following figure shows a graphical model over four binary valued variables X_1, \dots, X_4 . We do not know the parameters of the probability distribution associated with the graph.



- (a) Write the expression for the joint probability $\mathbb{P}(X_1, X_2, X_3, X_4)$ of the network in its *reduced* factored form.
- (b) Would it typically help to know the value of X_3 so as to gain more information about X_2 ?
- (c) Assume we already know the value of X_4 . Would it help in this case to know the value of X_3 to gain more information about X_2 ?
- (d) List three different conditional independence statements between the four variables that can be inferred from the graph. You can include marginal independence by saying “given nothing”.

Question 4: Hidden Markov Models (HMMs)

Consider a two-state hidden Markov model specified by (π, A, φ) that can output 4 possible values. Thus, the hidden states $z_i \in \{1, 2\}$, and the output values $x_i \in \{1, 2, 3, 4\}$. The further specification of the hidden Markov model is given as follows:

$$\pi = (0.5, 0.5), \quad A = \begin{pmatrix} 0.99 & 0.01 \\ 0 & 1 \end{pmatrix}, \quad \varphi = \begin{pmatrix} 0 & 0.199 & 0.8 & 0.001 \\ 0.1 & 0 & 0.7 & 0.2 \end{pmatrix}.$$

- (a) Give an example of an output sequence of length 2 that cannot be generated by the hidden Markov model specified above.
- (b) We generated a sequence of 2020^{2020} observations from the hidden Markov model, and found that the last observation in the sequence was 3. What is the most likely hidden state corresponding to that last observation?
- (c) Consider an output sequence with a 3 followed by another 3. What is the most likely sequence of hidden states corresponding to these observations?
- (d) Consider an output sequence with a 3 followed by another 3 followed by a 4. What are the *first two states* of the most likely sequence of hidden states corresponding to these observations?

Question 5: Reinforcement learning

We are using Q-learning to learn a policy in a system with two states s_1 and s_2 . State s_1 has two actions (a and b), and state s_2 has only one action (c). Suppose that the discount factor is γ , and the learning rate is α . We initialize the Q-table with all zero values.

- On the first transition, you start in state s_1 , apply action a , receive an immediate reward of 1, and then land in state s_2 . What is the resulting Q-table? Give your answer as an algebraic expression that may include one or both of the symbols γ and α .
- On the second transition, you apply action c , receive an immediate reward of 0, and then land in state s_1 . What is the resulting Q-table? Give your answer as an algebraic expression that may include one or both of the symbols γ and α .
- On the third transition, you apply action b , receive an immediate reward of 1, and then land in state s_2 . What is the resulting Q-table? Give your answer as an algebraic expression that may include one or both of the symbols γ and α .
- What is the optimal policy that Q-learning has learned so far?

Question 6: Regularized linear regression

Recall that the objective function for the L_2 regularized linear regression is given by

$$J(\mathbf{w}) = \|X\mathbf{w} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_2^2,$$

where X is the design matrix (the rows of X are the data points). During the lecture, we derived that the global minimizer of J is given by

$$\mathbf{w}^* = (X^T X + \lambda I)^{-1} X^T \mathbf{y}.$$

Now, let us consider running Newton's method to minimize J . Thus, let \mathbf{w}_0 be an arbitrary initial guess for Newton's method. One update step is then given by

$$\mathbf{w}_1 = \mathbf{w}_0 - [H(J(\mathbf{w}))]^{-1} \nabla_{\mathbf{w}} J(\mathbf{w}),$$

where H is the Hessian, and ∇ is the gradient.

- Show that \mathbf{w}_1 , the value of the weights after one Newton step, is equal to \mathbf{w}^* .

partial grade	1	2	3	4	5	6
(a)	1	3	1	1	1	3
(b)	1	2	1	1	1	
(c)	1	2	1	1	1	
(d)	1		1	1	1	
(e)	1					

Final grade is: (sum of partial grades) / 9.0 + 1.0