Exam Advanced Machine Learning
28 October 2021, 18.45–21.30

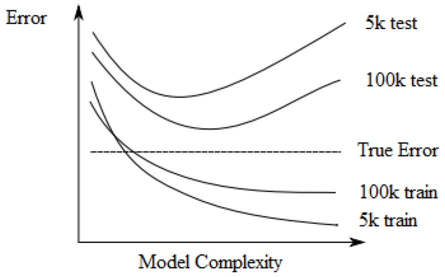
This exam consists of 6 problems, each consisting of several questions. All answers should be motivated, including calculations, formulas used, etc. The use of a calculator is not allowed.

Question 1: Short questions

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

(a) Suppose that we have two datasets that are sampled from the same distribution. Dataset A has 5,000 observations, and dataset B has 10,000 observations. We randomly construct a train set with 80% of the data, and a test set with the remaining 20%. Draw, for each dataset, a curve for the training error and a curve for the test error as a function of the model complexity. Thus, you will have 4 curves where the x-axis represents the model complexity and the y-axis the error. Clearly mark all your curves.

See figure below



(b) You want to solve a classification task. You first train your model on 20 samples. The training converges, but the training loss is very high. You then decide to train the model on 10,000 samples. Will your approach fix the problem? If yes, explain the most likely results when training with 10,000 samples. If not, give a solution to fix the problem.

The model is suffering from a bias problem. Increasing the amount of data reduces the variance, and is not likely to solve the problem. A better approach would be to create a model with more flexibility, i.e., learnable parameters.

(c) Explain the purpose of using an 1×1 convolution in a convolutional neural network.

The purpose of a 1×1 convolution is to reduce the number of multiplications needed in a convolutional neural network.

(d) If your input image is $64 \times 64 \times 16$, how many parameters are there in a single 1×1 convolution filter (including bias)?

There are 17 parameters.

(e) You come across a non-linear activation function g that passes 1 if its input is non-negative, else evaluates to 0. Thus,

$$g(z) = \begin{cases} 1, & z \geq 0, \\ 0, & z < 0. \end{cases} \tag{1}$$

A friend recommends you to use this non-linearity in your convolutional neural network with the Adam optimizer. Would you follow his advice? Explain your answer.

No, the function is not continuous but more importantly the gradient is zero everywhere.

(f) A popular model used for sentiment classification is an LSTM model. This model inputs word vectors to the LSTM model at each time step, and uses the last hidden state vector to predict the sentiment label. An alternative method for sentiment classification is the bag-of-vectors model. In this model, the average of all the word vectors in a sentence is used to predict the sentiment label. Name at least one benefit of the LSTM model over the bag-of-vector model for sentiment classification.

The LSTM network takes the word order into account.

Question 2: Neural networks

The following neural network in Figure 1 has 3 units. The neural network operates almost as a regular neural network. Each unit takes a linear combination of the units of the previous layer, *subtracts* a bias term, and then applies an activation function g to obtain the activated units (i.e., a_1, a_2, or a_3). The activation function is a negative sigmoid function, i.e., $g(z) = -1/(1+e^{-z})$, which differs from the standard sigmoid function by a minus sign. Additionally, this network uses a *non-standard error* function $E = \frac{1}{2}(2y - 2t)^2$, with t the target value and $y = a_3$ the output of the neural network.

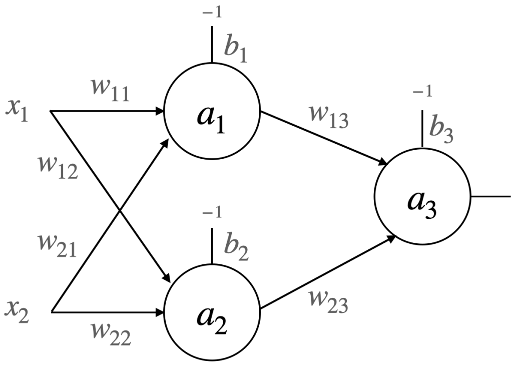


Figure 1: Neural network architecture.

(a) Using the initial weights provided below, and the input vector $(x_1, x_2) = (2, 0.5)$, compute the output at each neuron a_1, a_2, and a_3 after forward propagation. Use the negative sigmoid values given in the table at the end of this question.

weight	b_1	w_{11}	w_{12}	w_{13}	b_2	w_{21}	w_{22}	w_{23}	b_3
value	0.5	1	1	0.5	0.5	1	1	0.25	0.5

$a_1 = g(2) = -0.88,$
$a_2 = g(2) = -0.88,$
$a_3 = g(-1.16) \approx -0.24.$

(b) Calculate the derivative of $g(z)$ with respect to z.

$g'(z) = g(z)(1 + g(z)).$

(c) Calculate $\partial E/\partial w_{13}$.

$\partial E/\partial w_{13} = 4(a_3 - t)g(z_3)(1 + g(z_3))a_1.$

(d) Calculate $\partial E/\partial w_{11}$.

$\partial E/\partial w_{11} = 4(a_3 - t)g(z_3)(1 + g(z_3))w_{13}g(z_1)(1 + g(z_1))x_1.$

z	$g(z)$	z	$g(z)$
-3.00	-0.05	0.25	-0.56
-2.75	-0.06	0.50	-0.62
-2.50	-0.08	0.75	-0.68
-2.25	-0.10	1.00	-0.73
-2.00	-0.12	1.25	-0.78
-1.75	-0.15	1.50	-0.82
-1.50	-0.18	1.75	-0.85
-1.25	-0.22	2.00	-0.88
-1.00	-0.27	2.25	-0.90
-0.75	-0.32	2.50	-0.92
-0.50	-0.38	2.75	-0.94
-0.25	-0.44	3.00	-0.95
0.00	-0.50	3.25	-0.96

Question 3: Graphical models

The following figure shows a graphical model over nine binary-valued variables $A, \ldots, I$. We do not know the parameters of the probability distribution associated with the graph.

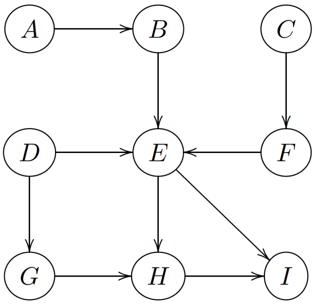


Figure 2: Graphical model.

(a) Write the expression for the joint probability $\mathbb{P}(A,B,C,D,E,F,G,H,I)$ of the network in its *reduced* factored form.

$\mathbb{P}(A)\mathbb{P}(B|A)\mathbb{P}(C)\mathbb{P}(D)\mathbb{P}(E|B,D,F)\mathbb{P}(F|C)\mathbb{P}(G|D)\mathbb{P}(H|E,G)\mathbb{P}(I|E,H)$.

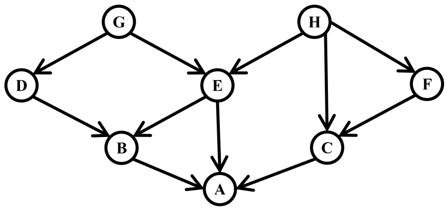
(b) Which of the following conditional independence assertions are true?

i) $A \perp\!\!\!\perp B$
ii) $A \perp\!\!\!\perp C$
iii) $A \perp\!\!\!\perp D \,|\, E$
iv) $A \perp\!\!\!\perp I \,|\, E$
v) $B \perp\!\!\!\perp C \,|\, I$
vi) $D \perp\!\!\!\perp I \,|\, \{E,G,H\}$

i) False
ii) True
iii) False
iv) False
v) False
vi) True

(c) Draw the graphical model that corresponds to $\mathbb{P}(A,B,C,D,E,F,G,H)$ given in its reduced form by: $\mathbb{P}(A\,|\,B,C,E)\mathbb{P}(B\,|\,D,E)\mathbb{P}(C\,|\,F,H)\mathbb{P}(D\,|\,G)\mathbb{P}(E\,|\,G,H)\mathbb{P}(F\,|\,H)$ $\mathbb{P}(G)\mathbb{P}(H)$.

See figure below



Question 4: Hidden Markov Models (HMMs)

A magician has two coins, each of which has an unknown type. The coins can either be fair coins (50/50 odds of heads versus tails), or trick coins that either (1) have heads on both sides, or (2) have tails on both sides. A priori, each coin is equally likely to be any of the three possible types. At every time step, the magician randomly picks a coin (without showing you which one was selected), flips it, and shows you the result. However, unfortunately, the magician only shows you the coin very briefly, and 10% of the time you make a mistake when you observe the true side of the coin (e.g., you see heads when it was actually tails).

(a) Model this process as a Hidden Markov Model. Specify all the parameters (π, A, φ). Think carefully about the states in the model.

Take as states the coins that the magician has picked. This gives 5 states: (F,F), (F,T), (F,H), (H,H), and (T,T), where F = fair coin, T = a biased tail coin, and H = a biased head coin. This gives: $\pi = (1/5, 1/5, 1/5, 1/5, 1/5)$, $A = I$, and

$$\varphi^\top = \begin{pmatrix} 1/2 & 3/10 & 7/10 & 9/10 & 1/10 \\ 1/2 & 7/10 & 3/10 & 1/10 & 9/10 \end{pmatrix}.$$

where the first row represents observing H and the second observing T. The columns represent the states ordered by (F, F), (F, T), (F, H), (H, H), and (T, T).

(b) As you watch the magician, you observe the following sequence A = (Heads, Tails, Heads, Tails) in the first four trials. On a different occasion, you observe sequence B = (Heads, Heads, Heads, Heads) in the first four trials. Which sequence is more likely to occur? Provide an argument/calculation to your answer.

Sequence B is more likely. It contains the term $(9/10)^4$ in the expectation. This is much higher than the terms in the expected value of sequence A.

(c) Consider the Markov model that would result if you ran the process above and observed heads for the first 20 times. What is the long-run probability that you will observe heads and tails in this case? Support your answer with an argument.

The long-run probability is $(9/10, 1/10)$.

Question 5: Logistic regression

You are building a classification model to distinguish between labels from a synthetically generated dataset. More specifically, you are given a dataset $(x^{(i)}, y^{(i)})$, where $x^{(i)} \in \mathbb{R}$ and $y^{(i)} \in \{0, 1\}$ for $i = 1, \ldots, m$. The data is generated with the following scheme.

$$X|Y = 0 \sim \mathcal{N}(2, 2)$$
$$X|Y = 1 \sim \mathcal{N}(0, 3)$$

Thus, $X|Y = 0$ is generated according to a Normal distribution with mean 2 and variance 2, where $X|Y = 1$ is generated according to a Normal distribution with mean 0 and variance 3. You can assume that the dataset is perfectly balanced between the two classes.

(a) As a baseline, you decide to use a logistic regression model to fit the data. Since the data is synthesized easily, you can assume you have infinitely many samples (i.e., $m \to \infty$). Can your logistic regression model achieve 100% training accuracy? Explain your answer.

No, the standard deviation is high enough for considerable overlap.

(b) After training on a large training set of size M, your logistic regressor achieves a training accuracy of T. Can the following technique, *applied individually* improve over this *training accuracy*? Please justify your answer.

(i) Adding a regularizing term to the binary cross-entropy loss function for the logistic regressor.

(ii) Standardizing all training samples to have mean zero and unit variance.

(iii) Using a 5-hidden layer feedforward network without non-linearities in place of logistic regression.

(iv) Using a 2-hidden layer feedforward network with ReLu in place of logistic regression.

i) No
ii) No
iii) No
iv) Yes

Question 6: Gradient descent

We are considering a supervised machine learning problem where $t \in \mathbb{R}$ denotes the labels, and $\mathbf{x} \in \mathbb{R}^d$ denotes the data. In our model, we denote by $\mathbf{w} \in \mathbb{R}^d$ the weight vector. We have m data points in total, and we use regularization with parameter $\lambda > 0$. Define the loss function $\mathcal{L}(\mathbf{w})$ by

$$\mathcal{L}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} l(t_i - \mathbf{w}^\top \mathbf{x}_i) + \frac{\lambda}{2}||\mathbf{w}||^2, \text{ with } l(\xi) = \begin{cases} \frac{1}{2}\xi^2, & \text{if } |\xi| < 1, \\ |\xi| - \frac{1}{2}, & \text{otherwise.} \end{cases} \tag{2}$$

The error function l is known as the Huber function, and is less strict than the squared error function.

(a) Calculate the gradient of $\mathcal{L}(\mathbf{w})$ with respect to $\mathbf{w}$.

$\partial \mathcal{L}(\mathbf{w})/\partial \mathbf{w} = 1/m \sum_{i=1}^{m} g_i + \lambda \mathbf{w}$, where $g_i = (\mathbf{w}^\top x_i - t_i)x_i$ if $|\mathbf{w}^\top x_i - t_i| < 1$, and $\operatorname{sgn}(\mathbf{w}^\top x_i - t_i)x_i$ otherwise.

(b) Write out a stochastic gradient descent learning algorithm for minimizing $\mathcal{L}(\mathbf{w})$.

$\mathbf{w} \leftarrow \mathbf{w} - \eta_t(g_i + \lambda \mathbf{w})$.

(c) After some careful thoughts, you decide to use gradient descent with momentum. Explain how momentum speeds up learning compared to standard gradient descent.

The oscillations are dampened by $\hat{g} = \beta\hat{g} + (1-\beta)g$, so that the update becomes $\mathbf{w} \leftarrow \mathbf{w} - \alpha\hat{g}$.

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

Final grade is: (sum of partial grades) / 4.0 + 1.0