Midterm Exam

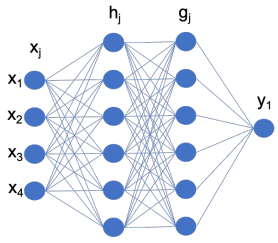
AI75823 Deep Learning, Spring 2022
School of BioMedical Convergence Engineering, PNU
Due: Apr. 19. 9:00 (AM)

I. REMARK

- This is an open book exam.
- There are a total of 100 points in the exam. Each problem specifies its point total.
- You must SHOW AND DESCRIBE YOUR WORK in answer sheets to get full credit.
- You must SUBMIT ALL CODE FILES so that TA can run the codes.
- If you DISOBEY THE DEADLINE, you will get only 50 % of your full score!!!!
- If you JUST COPY YOUR COLLEAGUE'S, you will get 0 point.
- If you do not use COLAB, please describe how TA runs your code files.
- Answer using Korean or English.

II. PROBLEM SET

1) The figure below is a fully-connected network. Assume a simple gradient descent method is used. Also, assume that every layer has no ReLU and the last layer has no an output function (unit), but every node has a bias. Assume that batch size is 1 and the loss function is MSE (between estimate y_1 and ground-truth $\bar{y}_1$). Describe a back-propagation algorithm for the network in detail. It should be pseudo-code like Algorithm 6.2 in Chapter 6 in the textbook. [10 points]



2) Answer the following questions [10 points].

   a) Derive that binary cross-entropy is the optimal loss function for binary classification with respect to maximum likelihood.

   b) Why is the sigmoid function required as an output unit for the classification problem.

3) The table describes the input-output set of the 3-input Exclusive NOR function. [10 points]

Input A	Input B	Input C	Output Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

   a) Make an estimator using linear regression.

   b) Make an estimator using deep learning structure (neural networks). Use "MSE" to minimize error between estimate and ground-truth (output Y). Try to make as simplest structure as possible.

   c) Compare the results of linear regression with that of neural networks.

4) The task is to investigate that neural nets can compute any function. One function is given as $f(x,y) = 0.2 + 0.8x^2 + 0.6y^2 + 0.5xy + \sin(15x) + \cos(40y)$ where $0 \leq x \leq 1$ and $0 \leq y \leq 1$. [10 points]

   a) Plot the functions $f(x,y)$, $\frac{\partial f(x,y)}{\partial x}$, $\frac{\partial f(x,y)}{\partial y}$ and $\frac{\partial^2 f(x,y)}{\partial x \partial y}$.

   b) Implement a deep-learning code for representing the functions. The input of the network is $[x,y]^T$ using 2 nodes and the output of the network is $[f(x,y), \frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y}, \frac{\partial^2 f(x,y)}{\partial x \partial y}]^T$ using 4 nodes. The network must be updated through minimizing the error between the estimate and ground-truth. Describe your networks and show your code. Make training data via picking (x,y) randomly in the given ranges. The number of pairs is up to you. Train your network using the

data.

c) For testing, input the pairs in $\{(x_i, y_j)|x_i = 0.01i, y_j = 0.01j, 0 \leq x_i \leq 1, 0 \leq y_i \leq 1\}$ where i and j are integers. Show four graphs representing the percentage errors between the (deep-learning) estimates and ground-truths on the xy domain for the functions $f(x, y)$, $\frac{\partial f(x,y)}{\partial x}$, $\frac{\partial f(x,y)}{\partial y}$ and $\frac{\partial^2 f(x,y)}{\partial x \partial y}$.

5) The task is to create a deep-learning algorithm to distinguish dogs from cats. Follow the directions. [30 points].

a) Go to the 'http://kaggle.com/c/dogs-vs-cats/data' . Download and unzip the file 'train.zip' ,

b) Load the image files. Randomly select 10,000 dog figures and 10,000 cat figures. Every figure title has a label indicating either dog or cat. Use the 8,000 dog figures and 8,000 cat figures for training, and use the remain for validation.

c) Preprocess the images for deep-learning if you need. You can resize the image if it is too big under your processor conditions. You can use any preprocessing method for facilitating learning process.

d) Design "TWO" deep-learning structures. The first one is fully-connected network model and the second one is CNN model. Describe your models as diagrams. Also, count the number of trainable weights for every model and explain how to get the number in detail from the number of nodes, layers, channels or kernel size extra...

e) For every model, set up the optimization algorithm, learning rate, batch size and epoch numbers, and describe them. Graph the training loss and validation loss over epoch. Explain the graph results.

f) What is your accuracy (% success rate) for the validation data set? Give a shot to obtain maximum accuracy for every model. It must be at least 65%. Discuss your results. Save your trained models using files so that you can load the files when you test the models.

g) Download and unzip the file 'test1.zip'. Label 100 images ('1.jpg', '2.jpg', ... '100.jpg') by yourself. Use the images as test data and obtain classification results through loading your model files and running the models. What is your accuracy? It must be also at least 65%.

6) The task is to create a deep-learning algorithm to nullify noises and artifacts from brain activation signals. You need to follow the directions. [30 points].

a) Download the data files from Plato. You may read the files using "numpy.load". "tRinpData" and "tRoutData" have corrupted brain signals and clear brain signals, respectively. "tRrefData" have reference signals to facilitate the learning. The size of every signal is 1024 (which corresponds to 102.4 sec). The total number of samples is 50,000. Pick some samples and plot corrupted signals and clear signals.

b) Make two-channel input data (1024×2) using "tRinpData' and "tRrefData". Use "tRoutData" as ground-truth. Design and implement a deep-learning structure. During training, the model should minimize the error between its output (estimated signal) and ground-truth (clear brain signal). Plot training and validation errors over epoch. Describe whole procedures as detail as possible.

c) For testing, use "tEinpData" and tErefData". Compute MSE using the estimates and "tEoutData". Show a graph displaying MSE values from all samples. Pick some samples and plot corrupted signal, estimated signal and ground-truth signal.