HW4
15 Points

1 Instructions

You may choose whatever software (overleaf, Latex, Word, etc.) to write your answers. But make sure to save your file as PDF (the only acceptable format) and upload to Canvas by the deadline.

This homework is based on Neural Network lectures and primarily focuses on math and logic questions with a small programming component for the last question. To better familiarize yourself with the questions and the lecture materials, first go through the description section. The five HW problems are posted afterwards.

2 Description of Neural Networks

- Neural networks refer to a general class of models for making predictions on data. The key feature of neural networks is that the input data is processed in multiple stages, called *layers*. The parameters in each layer are *trained* or *learned* from examples. This multi-layer processing with trainable parameters in each layer is loosely inspired by biological neural systems.
- To describe the neural network, let $\mathbf{x}$ denote the input vector and let y denote the target variable that we wish to predict from $\mathbf{x}$.
- We consider networks for both **regression** or **classification** problems:
  - For regression problems, y is a scalar or vector and is typically continuous-valued. In this case, the neural network produces a prediction $\hat{y}$ of y of the same dimension as y.
  - For classification problems, the target variable $y \in \{1, \ldots, K\}$ and the neural network typically provides a soft prediction of the class label. Specifically, the networks makes a prediction of the probability $P(y = k|\mathbf{x})$ for each class label k given the input $\mathbf{x}$.
- For both regression and classification problems, we assume the input $\mathbf{x}$ is a vector of dimension N_i so that
$$\mathbf{x} = (x_1, \ldots, x_{N_i}).$$
- In this note, we look at neural networks with one hidden layer. In such a network, the neural network mapping is performed in two steps – each step is called a *layer*:
  - *Hidden layer* produces outputs $\mathbf{z}^{\mathrm{H}}$ and $\mathbf{u}^{\mathrm{H}}$ of dimension N_h.

  - *Output layer* produces outputs $\mathbf{z}^{\text{O}}$ and $\mathbf{u}^{\text{O}}$ of dimension N_o.

  The dimensions N_h and N_o will be discussed below.

- The equations for the two layers with a single input $\mathbf{x}$ are:

$$\text{Hidden layer:} \quad z_j^{\text{H}} = \sum_{k=1}^{N_i} W_{jk}^{\text{H}} x_k + b_j^{\text{H}}, \quad u_j^{\text{H}} = g_{\text{act}}(z_j^{\text{H}}), \quad j = 1, \ldots, N_h \tag{1a}$$

$$\text{Output layer:} \quad z_j^{\text{O}} = \sum_{k=1}^{N_h} W_{jk}^{\text{O}} u_k^{\text{H}} + b_j^{\text{O}}, \quad u^{\text{O}} = g_{\text{out}}(\mathbf{z}^{\text{O}}). \quad j = 1, \ldots, N_o. \tag{1b}$$

- In the hidden layer, the function $g_{\text{act}}(z)$ is called the *activation function.* There are three common choices:

  - Hard threshold:

$$g_{\text{act}}(z) = \begin{cases} 1, & \text{if } z \geq 0 \\ 0, & \text{if } z < 0. \end{cases} \tag{2}$$

  - Sigmoid: $g_{\text{act}}(z) = 1/(1 + e^{-z})$.
  - Rectified linear unit (ReLU): $g(z) = \max\{0, z\}$.

- The output map $g_{\text{out}}(z)$ and dimension N_o depends on the type of prediction problem we are using the neural network for:

  - Binary classification: In this case, the response is $y = \{0, 1\}$. To predict the response, we typically take $N_o = 1$ and the output u^{O} is the class probability:

$$P(y = 1|\mathbf{x}) = u^{\text{O}} = g_{\text{out}}(z^{\text{O}}) = \frac{1}{1 + e^{-z^{\text{O}}}}. \tag{3}$$

    The variable z^{O} is called the *logit* and the output mapping (3) is a sigmoid. We can also use z^{O} to make a hard decision by selecting the most likely class:

$$\hat{y} = \begin{cases} 1 & z^{\text{O}} \geq 0 \\ 0 & z^{\text{O}} < 0. \end{cases} \tag{4}$$

  - K-class classification: In this case, the target is a class label $y = 1, \ldots, K$. We typically take $N_o = K$ and use the soft-max function for the class probability:

$$P(y = k|\mathbf{x}) = u_k^{\text{O}} = g_{\text{out,k}}(z^{\text{O}}) = \frac{e^{z_k^{\text{O}}}}{\sum_{\ell=1}^{K} e^{z_\ell^{\text{O}}}}. \tag{5}$$

    Again, we can make a hard decision on the class label by selecting the highest logit score:

$$\hat{y} = \arg\max_{k=1,\ldots,K} z_k^{\text{O}}. \tag{6}$$

- Regression: In this case, one is trying to predict $\mathbf{y} \in \mathbb{R}^d$, where d is the number of variables to predict and each component y_j is typically continuous-valued. For this problem, we take $N_o = d$ and $\mathbf{u}^{\text{O}}$ is the prediction of $\mathbf{y}$,

  $$\hat{\mathbf{y}} = \mathbf{u}^{\text{O}} = g_{\text{out}}(\mathbf{z}^{\text{O}}) = \mathbf{z}^{\text{O}}. \tag{7}$$

  In (7), we will either say there is no activation or an *identity* activation.

- The number, N_h of hidden variables (also called the number of *hidden units*) represents the model complexity. Higher values of N_h can express more complex mappings, but will need more data to train.

- When $N_o = 1$ (single output unit), we drop the subscript j in (1b) and write

  $$\text{Output layer:} \quad z^{\text{O}} = \sum_{k=1}^{N_h} W_k^{\text{H}} u_k^{\text{H}} + b^{\text{O}}, \quad \hat{y} = g_{\text{out}}(z^{\text{O}}). \tag{8}$$

- Matrix form of the neural network with single sample input $\mathbf{x}$:

  $$\begin{aligned} \mathbf{z}^{\text{H}} &= \mathbf{W}^{\text{H}}\mathbf{x} + \mathbf{b}^{\text{H}}, \quad \mathbf{u}^{\text{H}} = g_{\text{act}}(\mathbf{z}^{\text{H}}), \\ \mathbf{z}^{\text{O}} &= \mathbf{W}^{\text{O}}\mathbf{u}^{\text{H}} + \mathbf{b}^{\text{O}}, \quad \mathbf{u}^{\text{O}} = g_{\text{out}}(\mathbf{z}^{\text{O}}). \end{aligned}$$

- Batch form: Often we have a batch of input-output samples, $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, N$ (e.g. a mini-batch in training or test). In this case, we need to add an index i to each sample and rewrite (1) as:

  $$\text{Hidden layer:} \quad z_{ij}^{\text{H}} = \sum_{k=1}^{N_i} W_{jk}^{\text{H}} x_{ik} + b_j^{\text{H}}, \quad u_{ij}^{\text{H}} = g_{\text{act}}(z_{ij}^{\text{H}}), \quad j = 1, \ldots, N_h \tag{9a}$$

  $$\text{Output layer:} \quad z_{ij}^{\text{O}} = \sum_{k=1}^{N_h} W_{jk}^{\text{O}} u_{ik}^{\text{H}} + b_j^{\text{O}}, \quad \mathbf{u}_j^{\text{O}} = g_{\text{out}}(\mathbf{z}_i^{\text{O}}), \quad j = 1, \ldots, N_o. \tag{9b}$$

- Matrix form of batch processing in neural networks: Define the matrices,

  $$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^{\mathsf{T}} \\ \vdots \\ \mathbf{x}_N^{\mathsf{T}} \end{bmatrix} = \begin{bmatrix} x_{11} & \cdots & x_{1,N_i} \\ \vdots & \vdots & \vdots \\ x_{N1} & \cdots & x_{N,N_i} \end{bmatrix}, \quad \mathbf{Z}^{\text{H}} = \begin{bmatrix} (\mathbf{z}_1^{\text{H}})^{\mathsf{T}} \\ \vdots \\ (\mathbf{z}_N^{\text{H}})^{\mathsf{T}} \end{bmatrix} = \begin{bmatrix} z_{11}^{\text{H}} & \cdots & z_{1,N_h}^{\text{H}} \\ \vdots & \vdots & \vdots \\ z_{N1}^{\text{H}} & \cdots & z_{N,N_h}^{\text{H}} \end{bmatrix},$$

  Similarly define

  $$\mathbf{U}^{\text{H}} = \begin{bmatrix} (\mathbf{u}_1^{\text{H}})^{\mathsf{T}} \\ \vdots \\ (\mathbf{u}_N^{\text{H}})^{\mathsf{T}} \end{bmatrix} = \begin{bmatrix} u_{11}^{\text{H}} & \cdots & u_{1,N_h}^{\text{H}} \\ \vdots & \vdots & \vdots \\ u_{N1}^{\text{H}} & \cdots & u_{N,N_h}^{\text{H}} \end{bmatrix}. \quad \mathbf{U}^{\text{O}} = \begin{bmatrix} (\mathbf{u}_1^{\text{O}})^{\mathsf{T}} \\ \vdots \\ (\mathbf{u}_N^{\text{O}})^{\mathsf{T}} \end{bmatrix} = \begin{bmatrix} u_{11}^{\text{O}} & \cdots & u_{1,N_o}^{\text{H}} \\ \vdots & \vdots & \vdots \\ u_{N1}^{\text{O}} & \cdots & u_{N,N_o}^{\text{H}} \end{bmatrix}.$$

  Hence each row contains all the components for the i-th sample. Then, we can write

  $$\mathbf{Z}^{\text{H}} = \mathbf{X}\left[\mathbf{W}^{\text{H}}\right]^{\mathsf{T}} + \mathbf{B}^{\text{H}}, \quad \mathbf{U}^{\text{H}} = g_{\text{act}}(\mathbf{Z}^{\text{H}}) \tag{10a}$$

  $$\mathbf{Z}^{\text{O}} = \mathbf{U}^{\text{H}}\left[\mathbf{W}^{\text{O}}\right]^{\mathsf{T}} + \mathbf{B}^{\text{O}} \quad \mathbf{U}^{\text{O}} = g_{\text{out}}\left(\mathbf{Z}^{\text{O}}\right), \tag{10b}$$

where $\mathbf{B}^{\text{H}}$ and $\mathbf{B}^{\text{O}}$ repeat the bias vectors on the N rows

$$\mathbf{B}^{\text{H}} = \begin{bmatrix} (\mathbf{b}^{\text{H}})^{\mathsf{T}} \\ \vdots \\ (\mathbf{b}^{\text{H}})^{\mathsf{T}} \end{bmatrix} \quad \mathbf{B}^{\text{O}} = \begin{bmatrix} (\mathbf{b}^{\text{O}})^{\mathsf{T}} \\ \vdots \\ (\mathbf{b}^{\text{O}})^{\mathsf{T}} \end{bmatrix} \tag{11}$$

Problems

1. Consider the neural network (1) with a scalar input x and parameters

$$W^{\text{H}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad b^{\text{H}} = \begin{bmatrix} -1 \\ -3 \end{bmatrix} \quad W^{\text{O}} = [-1, 2], \quad b^{\text{O}} = 0.5,$$

using a hard threshold activation function (2) and threshold output function (4).

(a) What is N_h, the number of hidden units? What is N_o, the number of output units?

(b) Write $\mathbf{z}^{\text{H}}$ in terms of x. Show the functions for each component z_j^{H}.

(c) Write $\mathbf{u}^{\text{H}}$ in terms of x. Show the functions for each component u_j^{H}.

(d) Write z^{O} in terms of x.

(e) What is $\hat{y}$ in terms of x?

2. Consider the data set for four points with scalar features x_i and binary class labels $y_i = 0, 1$.

x_i	0	1	3	5
y_i	0	0	1	0

(a) Find a neural network with $N_h = 2$ units, $N_o = 1$ output units such that $\hat{y}_i = y_i$ on all four data points. Use a network similar in structure to the previous problem. Also, you want to find features that can distinguish between $x = 3$ and $x = \{0, 1, 5\}$. Since there are many features, use two features: whether $x \geq 2$ and $x \geq 4$. Use a hard threshold activation function (2) and threshold output function (4). State the weights and biases used in each layer.

(b) Compute the values of $\hat{y}_i$ and all the intermediate variables $\mathbf{z}_i^{\text{H}}$, $\mathbf{u}_i^{\text{H}}$ and z_i^{O} for each sample $x = x_i$.

(c) Now suppose we are given a new sample, $x = 3.5$. What does the network predict as $\hat{y}$?

3. *Two-dimensional example:* Consider a neural network on a 2-dimensional input $\mathbf{x} = (x_1, x_2)$ with weights and biases:

$$W^{\text{H}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}, \quad b^{\text{H}} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \quad W^{\text{O}} = [1, 1, -1], \quad b^{\text{O}} = -1.5.$$

Assume the network uses hard a threshold activation function (2) and threshold output function (4).

(a) Write the components of $\mathbf{z}^{\mathrm{H}}$ and $\mathbf{u}^{\mathrm{H}}$ as a function of (x_1, x_2). For each component j, indicate where in the (x_1, x_2) plane $u_j^{\mathrm{H}} = 1$.

(b) Write z^{O} as a function of (x_1, x_2). In what region is $\hat{y} = 1$?

4. *Architecture choices for different problems:* For each problem, state possible selections for the dimensions N_i, N_h, N_o and the functions $g_{\mathrm{act}}(\cdot)$ and $g_{\mathrm{out}}(\cdot)$. Indicate which parameters are free to choose.

   (a) One wants a neural network to take as an input a 20×20 gray scale image and determine which letter ('a' to 'z') the image is of.

   (b) One extracts 120 features of a sample of a speech recording (like the MFCCs). Based on the audio samples, the network is to determine if the speech is male or female.

   (c) One wants a neural network to predict the stock price based on the average stock price of the last five days.

5. *Implementation in python:* Write python code for implementing the following steps for a batch of samples:

   (a) The hidden layer step (10a).

   (b) The output layer step (10b) for binary classification with a sigmoid output (3).

   (c) The output layer step (10b) for K-class classification with a softmax output (5).

   For all examples, avoid for-loops and instead use Python broadcasting.