# CS 475 Machine Learning: Homework 5
# Deep Learning: Analytical
Due: Thursday November 12, 11:59pm
40 Points Total Version 1.0

Wenkai Luo (wluo14), Yuetong Liu (yliu390)

## Homework 5

For the remaining homeworks, we will combine all three homework types into single assignments. Both homework 5 and 6 will be worth 100 points. Homeworks 1-6 will be worth a total of 400.

Homework 5 has three parts totalling 100 points.

1. Analytical (40 points)
2. Programming (45 points)
3. Lab (15 points)

All three parts of the homework have the same due date. Late hours will be counted based on when the last part is submitted.

For this assignment you may only work with a partner on the analytical section.

## Instructions

We have provided this LaTeX document for turning in this homework. We give you one or more boxes to answer each question. The question to answer for each box will be noted in the title of the box.

**Other than your name, do not type anything outside the boxes. Leave the rest of the document unchanged.**

**Do not change any formatting in this document, or we may be unable to grade your work. This includes, but is not limited to, the height of textboxes, font sizes, and the spacing of text and tables. Additionally, do not add text outside of the answer boxes. Entering your answers are the only changes allowed.**

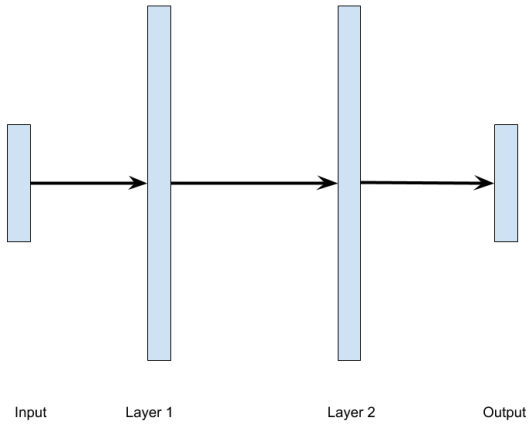
**We strongly recommend you review your answers in the generated PDF to ensure they appear correct. We will grade what appears in the answer boxes in the submitted PDF, NOT the original latex file.**

## Notation

| | |
|---|---|
| $\mathbf{x_i}$ | One input data vector. $\mathbf{x_i}$ is $M$ dimensional. $\mathbf{x_i} \in \mathbb{R}^{1\times M}$.<br>We assume $\mathbf{x_i}$ is augmented with a 1 to include a bias term. |
| $\mathbf{X}$ | A matrix of concatenated $\mathbf{x_i}$'s. There are $N$ input vectors, so $\mathbf{X} \in \mathbb{R}^{N\times M}$ |
| $y_i$ | The true label for input vector $\mathbf{x_i}$. In regression problems, $y_i$ is continuous.<br>In general ,$y_i$ can be a vector, but for now we assume it's a scalar: $y_i \in \mathbb{R}^1$. |
| $\mathbf{y}$ | A vector of concatenated $y_i$'s. There are $N$ input vectors, so $\mathbf{y} \in \mathbb{R}^{N\times 1}$ |
| $\mathbf{w}$ | A weight vector. We are trying to learn the elements of $\mathbf{w}$.<br>$\mathbf{w}$ is the same number of elements as $\mathbf{x_i}$ because we will end up computing the dot product $\mathbf{x_i} \cdot \mathbf{w}$.<br>$\mathbf{w} \in \mathbb{R}^{M\times 1}$. We assume the bias term is included in $\mathbf{w}$. |
| $h((x))$ | The true regression function that describes the data. |
| i.i.d. | Independently and identically distributed. |
| Bias-variance decomposition | We can write $E_D[(f(x, D) - h(x))^2] =$<br>$(E_D[f(x, D) - h(x))^2 + E_D[(f(x, D) - E_D[f(x, D)])^2]$<br>where the first term is the bias squared, and the second term is the variance. |
| Notes: | In general, a lowercase letter (not boldface), $a$, indicates a scalar.<br>A boldface lowercase letter, $\mathbf{a}$, indicates a vector.<br>A boldface uppercase letter, $\mathbf{A}$, indicates a matrix. |

# 1) Neural Network Construction (15 points)

MNIST is a popular dataset of images of hand-written digits. There are 10 digits (0 through 9) to classify. Each image in the dataset is 28x28 pixels, and the images are 1-channel grayscale images. Because of the simplicity of this dataset, it is possible to achieve good results by flattening images into 1-dimensional vectors and then training a multi-layer perceptron (MLP). Suppose we have a MLP with two hidden, fully connected layers:




Here, the input image is flattened into a 1-dimensional vector, and y is a multi-class categorical output with a probability assigned to each of the 10 digit classes (i.e. from a softmax activation). Assume each hidden layer has 100 neurons and denote the softmax activation functions as $\sigma$. The ReLU activation function is applied after Layer 1.

(a) (5 points) Assuming there are no bias terms, express the forward pass of an image through this MLP. Denote the weights of layers 1 and 2 as $W_1$ and $W_2$, respectively. Let X denote the vector of inputs. You can express the forward pass using multiplication of matrices.

If we define C is the output of the first layer, and each row of weight $W_1$ and $W_2$ represent corresponding neuron. Then we can express C in the following term:

$$C = \begin{bmatrix} C_1 \\ C_2 \\ .. \\ C_{100} \end{bmatrix} = \begin{bmatrix} max(0, W_{1,1} * X) \\ max(0, W_{1,2} * X) \\ .. \\ max(0, W_{1,100} * X) \end{bmatrix} = Relu(W_1 X)$$

$$O = \sigma(W_2 C) = \sigma[W_2 * Relu(W_1 X)]$$

(b) (5 points) How many parameters are there in each layer of the network?

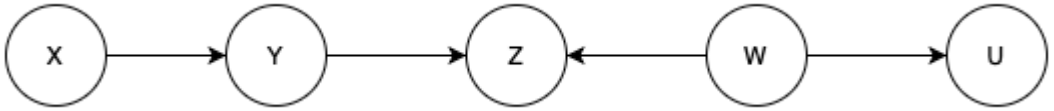
Based on the fact that layer 1 has 100 neurons, input size is 28*28 = 784.
Layer 1 is fully connected, so layer 1 has 784*200 = 78400 parameters.
With layer 2 is fully connected, then layer 2 has 100*10 = 1000 parameters.

(c) (5 points) Suppose we switch to a different dataset where the images are provided as high quality images (2000 x 2000 pixels) and images are now in color, where each pixel is represented by a RBG value (a triplet of integers between 0 and 255). Do you think the MLP used in the previous parts of the question will do well on this dataset? Why or why not?

With a higher dimension data, it might require larger network with more neurons to learn the useful feature.
Color image is a more complicated data form. For a specific pixel color, it can have different combination of RGB value. Meanwhile, by flatting a color image into one-dimension data form, it might not be able to encode such relative-position information into flatten data form.
Thus, I think previous MLP won't do well.

## 2) D-Separation (10 points)

The given diagram represents a Bayesian Network or DAG over random variables $X, Y, Z, W, U$




Using the properties of d-separation, which of the following independence statements are true in the DAG above? Give a one sentence explanation for each answer. The notation $X \perp Z|Y$ means that $X$ is independent of $Z$ given $Y$.

(a) $X \perp Z$

(b) $Z \perp U$

(c) $X \perp U$

(d) $X \perp U \mid Z$

(e) $Z \perp U \mid W$

(f) $X \perp Z \mid Y$

(g) $X \perp W \mid \{Y, Z\}$

(a) False, X and Z to Y is head-to-tail connection, path is unblocked and Z depend on X.

(b) False, Z and U to W is tail-to-tail connection, path is unblocked, both depend on W.

(c) True, X and U are generated without any common parents.

(d) False, Z is a head-to-tail node, path from X to U is unblocked.

(e) True, W is a tail-to-tail node, path from Z to U is blocked.

(f) True, Y is a head-to-tail node, path from X to Z is blocked.

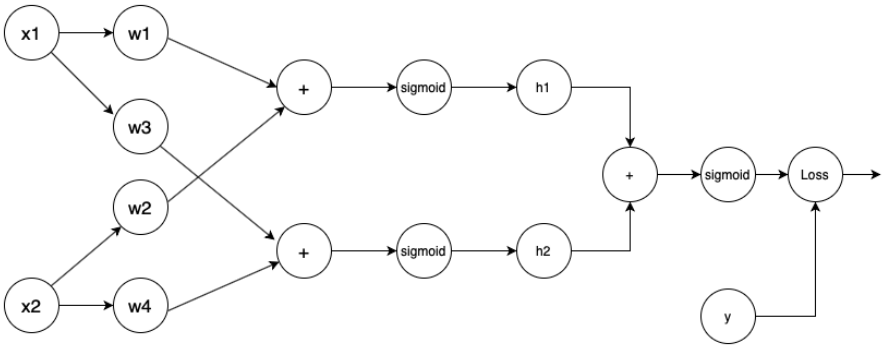
(g) False, Z is a head-to-head node, Y is a descendant of X, path is unblocked.

## 3) Backpropagation (15 points)

Suppose you have the following function

$$p = \sigma(h_1 \cdot \sigma(w_1 x_1 + w_2 x_2) + h_2 \cdot \sigma(w_3 x_1 + w_4 x_2))$$
$$L(p, y) = -y \log p - (1 - y) \log(1 - p)$$

Here $\sigma$ denotes the sigmoid activation function and $y$ is a binary label. Consider this computation graph that represents the above computation




We will use backpropagation to compute the gradients of the loss with respect to $w_1, w_2, w_3, w_4, h_1, h_2$. To demonstrate, let's compute $\frac{\partial L}{\partial h_1}$. To simplify notation, we will denote terms as follows

$$z = h_1 \cdot \sigma(w_1 x_2 + w_2 x_2) + h_2 \cdot \sigma(w_3 x_1 + w_4 x_2)$$
$$l_{12} = w_1 \cdot x_2 + w_2 \cdot x_2$$
$$l_{34} = w_3 \cdot x_1 + w_4 \cdot x_2$$

Using the chain rule, this can be written as $\frac{\partial L}{\partial h_1}$

$$\frac{\partial L}{\partial h_1} = \frac{\partial L}{\partial p} \cdot \frac{\partial p}{\partial z} \cdot \frac{\partial z}{\partial h_1}$$

And so, $\frac{\partial L}{\partial h_1}$ can be written as the product of gradients computed at individual nodes in the computational graph.

(a) (4 Points) Write out the expressions for the following derivatives.

(i) $\frac{\partial}{\partial q} - z \log q - (1-z)\log(1-q)$ (ii) $\frac{\partial}{\partial g_1}(g_1 * a_1 + g_2 * a_2)$

(iii) $\frac{\partial}{\partial z}\sigma(z)$ (iv) $\frac{\partial}{\partial g_1}(g_1 + g_2)$

(i) $\frac{\partial}{\partial q} - z \log q - (1-z)\log(1-q) = -z \cdot \frac{1}{q} + (1-z)\frac{1}{1-q} = \frac{q-z}{q \cdot (1-q)}$

(ii) $\frac{\partial}{\partial g_1}(g_1 * a_1 + g_2 * a_2) = \frac{\partial}{\partial g_1} g_1 * a_1 = a_1$

(iii) $\frac{\partial}{\partial z}\sigma(z) = \frac{\partial}{\partial z}\frac{1}{1+e^{-z}} = \frac{-1}{(1+e^{-z})^2} \cdot (-e^{-z}) = \frac{1}{1+e^{-z}} \cdot \frac{e^{-z}}{1+e^{-z}} = \sigma(z) \cdot (1-\sigma(z))$

(iv) $\frac{\partial}{\partial g_1}(g_1 + g_2) = \frac{\partial}{\partial g_1}(g_1) = 1$

(b) (4 points) Using the chain rule of calculus, write out the following expressions in terms of the partial derivatives, but do not evaluate the actual backpropagation updates.

$$\text{(i)}\ \frac{\partial}{\partial h_1}L \qquad \text{(ii)}\ \frac{\partial}{\partial h_2}L \qquad \text{(iii)}\frac{\partial}{\partial w_1}L$$

$$\text{(iv)}\ \frac{\partial}{\partial w_2}L \qquad \text{(v)}\ \frac{\partial}{\partial w_3}L \qquad \text{(vi)}\ \frac{\partial}{\partial w_4}L$$

From (a), we can compute that $\frac{\partial L}{\partial p} = \frac{p-y}{p(1-p)}$ and $\frac{\partial p}{\partial z} = \sigma(z)\cdot(1-\sigma(z))$. Thus we can use chain rule to compute the following partial derivatives.

$$\begin{aligned}
\frac{\partial L}{\partial h_1} &= \frac{\partial L}{\partial p}\cdot\frac{\partial p}{\partial z}\cdot\frac{\partial z}{\partial h_1} = \frac{p-y}{p(1-p)}\cdot\sigma(z)\cdot(1-\sigma(z))\cdot\sigma(l_{12})\\
\frac{\partial L}{\partial h_2} &= \frac{\partial L}{\partial p}\cdot\frac{\partial p}{\partial z}\cdot\frac{\partial z}{\partial h_2} = \frac{p-y}{p(1-p)}\cdot\sigma(z)\cdot(1-\sigma(z))\cdot\sigma(l_{34})\\
\frac{\partial L}{\partial w_1} &= \frac{\partial L}{\partial p}\cdot\frac{\partial p}{\partial z}\cdot\frac{\partial z}{\partial l_{12}}\cdot\frac{\partial l_{12}}{\partial w_1}\\
&= \frac{p-y}{p(1-p)}\cdot\sigma(z)\cdot(1-\sigma(z))\cdot h_1\left[\sigma(l_{12})\cdot(1-\sigma(l_{12}))\right]\cdot x_1\\
\frac{\partial L}{\partial w_2} &= \frac{\partial L}{\partial p}\cdot\frac{\partial p}{\partial z}\cdot\frac{\partial z}{\partial l_{12}}\cdot\frac{\partial l_{12}}{\partial w_2}\\
&= \frac{p-y}{p(1-p)}\cdot\sigma(z)\cdot(1-\sigma(z))\cdot h_1\left[\sigma(l_{12})\cdot(1-\sigma(l_{12}))\right]\cdot x_2\\
\frac{\partial L}{\partial w_3} &= \frac{\partial L}{\partial p}\cdot\frac{\partial p}{\partial z}\cdot\frac{\partial z}{\partial l_{34}}\cdot\frac{\partial l_{34}}{\partial w_3}\\
&= \frac{p-y}{p(1-p)}\cdot\sigma(z)\cdot(1-\sigma(z))\cdot h_2\left[\sigma(l_{34})\cdot(1-\sigma(l_{34}))\right]\cdot x_1\\
\frac{\partial L}{\partial w_4} &= \frac{\partial L}{\partial p}\cdot\frac{\partial p}{\partial z}\cdot\frac{\partial z}{\partial l_{34}}\cdot\frac{\partial l_{34}}{\partial w_4}\\
&= \frac{p-y}{p(1-p)}\cdot\sigma(z)\cdot(1-\sigma(z))\cdot h_2\left[\sigma(l_{34})\cdot(1-\sigma(l_{34}))\right]\cdot x_2
\end{aligned}$$

(c) (4 points) Using the backpropagation algorithm and your answers from part (a) and (b), evaluate the value of the partial derivatives in (b) at $x_1 = 1, x_2 = -1, y = 1, w_1 = 3, w_2 = 2, w_3 = -1, w_4 = 5, h_1 = 2.5, h_2 = 1.5$.

With the provided value, we can compute $l_{12} = 1$, $l_{34} = -6$, $z \approx 1.831$, $p \approx 0.862$ and $L \approx 0.149$, and thus evaluate of partial derivatives in (b):

$$\frac{\partial L}{\partial h_1} \approx -0.101$$
$$\frac{\partial L}{\partial h_2} \approx -0.0003$$
$$\frac{\partial L}{\partial w_1} \approx -0.0679$$
$$\frac{\partial L}{\partial w_2} \approx 0.0679$$
$$\frac{\partial L}{\partial w_3} \approx -0.0005$$
$$\frac{\partial L}{\partial w_4} \approx 0.0005$$

(d) (3 points) Instead of sigmoid activation, assume the intermediate layers use ReLU activations. How do the values of the gradients change?

Instead of sigmoid activation, assume the intermediate layers use ReLU activation. Compared with the sigmoid activation, ReLU activation can solve the vanishing gradient problem due to the sigmoid function. Thus what we expect to see is that those gradient magnitude(regardless the sign) relating to the ReLu activation will be larger than the one using sigmoid activation. For those gradient related to sigmoid activation, it also depend on the gradient of other variable, it cannot have a specific conclusion.

With the theorem that $\frac{\partial ReLu(x)}{\partial x} = 1, x > 0 \quad (0, otherwise)$, we can re-compute the gradient of each partial derivatives:

$\frac{\partial L}{\partial w_1} \approx -0.190$ $\frac{\partial L}{\partial w_2} \approx 0.190$ $\frac{\partial L}{\partial w_3} \approx -0.0$ $\frac{\partial L}{\partial w_4} \approx 0.0$

We can see that it aligns with our expectation.