# ELEG 5491 Introduction to Deep Learning

# Homework 1

**General Guidelines**:

- Please check the submission deadline on **Blackboard** and submit your solutions via **Blackboard**.
- Each homework's deadline has a grace period of 2 hours.
- Each student has one chance of late submission within 12 hours of the deadline.
- All other late submissions will be given 0 points with no exception.
- **Do not** close your browser of app before you have successfully uploaded your files. It is your own responsibility of keeping your file integrity.
- Show enough details of your solutions to earn full points
- Round your results to 4 decimal places or keep the fractional numbers.

1. (30 points) **Using L2 loss for binary logistic classification.** In the background chapter, the hypothesis $h$ of logistic regression for binary logistic classification is modeled as

$$h(x) = g\left(\theta^T x\right) = \frac{1}{1+e^{-\theta^T x}},$$

where $\theta = [\theta_0, \theta_1, \ldots, \theta_n]$ is the parameter vector. We have shown you that the binary cross entropy loss (which is *negative* log likelihood) is used to supervise the optimization of the parameters $\theta$ with a training set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$ of size $m$,

$$L_{\text{bce}}(\theta) = -\sum_{i=1}^{m} y^{(i)} \log h\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h\left(x^{(i)}\right)\right).$$

If we choose to use the following $L2$ loss function to supervise the optimization of parameters $\theta$

$$L_2(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)} - h(x^{(i)})\right)^2.$$

(a) (15 points) Find the $\dfrac{\partial L_2}{\partial \theta_j}$ for $j = 0, \ldots, n$.

(b) (15 points) Discuss why the $L2$ loss shouldn't be used for binary logistic classification.

2. (30 points) **Huber loss.** The Huber loss is usually used for regression tasks.

The forward inputs: $z^{(1)}, z^{(2)}, \ldots, z^{(N)}$ and ground truth $\hat{z}^{(1)}, \hat{z}^{(2)}, \ldots, \hat{z}^{(N)}$; forward output:

$$J = \frac{1}{N}\sum_{i=1}^{N} J^{(i)}, \quad J^{(i)} = \begin{cases} \frac{1}{2}(z^{(i)} - \hat{z}^{(i)})^2 & |z^{(i)} - \hat{z}^{(i)}| \le \delta, \\ \delta|z^{(i)} - \hat{z}^{(i)}| - \frac{1}{2}\delta^2 & \text{Otherwise.} \end{cases}$$

where $\delta$ is a constant, such as $\delta = 0.5$. Find backward output $\dfrac{\partial J}{\partial z^{(i)}}$.

3. (40 points) **Stochastic gradient descent for MLP.** Given the following training set $D = \{x^{(i)}, y^{(i)}\}_{i=1}^{4}$ for a three-class classification task:

$$x^{(1)} = [-0.7411, -0.5078, -0.3206]^T, y^{(1)} = [0, 1, 0]^T,$$
$$x^{(2)} = [0.0983, -0.0308, -0.3728]^T, y^{(2)} = [1, 0, 0]^T,$$
$$x^{(3)} = [0.0414, 0.2323, -0.2365]^T, y^{(3)} = [0, 1, 0]^T,$$
$$x^{(4)} = [-0.7342, 0.4264, 2.0237]^T, y^{(4)} = [0, 0, 1]^T.$$

Given a two-layer multi-layer perceptron with the following initial weights and biases for the first and second layer respectively.

$$W_1 = \begin{bmatrix} 1.6035 & -1.5062 & 0.2761 \\ 1.2347 & -0.4446 & -0.2612 \\ -0.2296 & -0.1559 & 0.4434 \end{bmatrix}, b_1 = [0.3919, -1.2507, -0.9480]^T$$

$$W_2 = \begin{bmatrix} 0.0125 & 1.2424 & 0.3503 \\ -3.0292 & -1.0667 & -0.0290 \\ -0.4570 & 0.9337 & 0.1825 \end{bmatrix}, b_2 = [-1.5651, -0.0845, 1.6039]^T$$

The ReLU function follows the first fully-connected layer, and the softmax function follows the second fully-connected. The network is trained with the cross-entropy loss.

If we use the **stochastic gradient descent** to update the parameters $W_1, W_2, b_1, b_2$.

(a) (15 points) What's the loss function value of the first iteration?

(b) (20 points) If you update the parameters for 1 iteration with learning rate 0.01, what's the parameters after updating for 1 iteration?

(c) (15 points) What's the loss function value of the 2nd iteration?