COMP 433 Introduction to Deep Learning (Fall 2025)

Major Assignment #2

Due: 11:59PM, October 19th, 2025

Note Your will be submitting two separate files from this assignment as follows:

(a) **One(1) .pdf file:** containing answers to questions as well as reported results and snapshots from coding you develop. **Please write all the steps**. It is not necessary to type your answers. You may write them by hand in a legible manner and include photos of your solution as part of your submission.

(b) **One(1) .zip folder:** containing all developed Python codes. As mentioned, the snapshots of the outputs should be included in the PDF file.

(c) **Format:** For both files, please use a firstname_lastname_stuid formt. Example: John_Doe_1234567.pdf. If your first name or last name is composed of multiple parts, you can just write one part.

Theoretical Questions

Question 1 Consider a *fully-connected* neural network. The input matrix X has a dimension of NxD where N is the number of samples (observations) and D is the number of variables/features. Inputs are connected to a hidden layer with M neurons, all using the same activation function $g()$. The hidden layer is then connected to the output layer with K outputs all using ReLU activation function. Using a softmax layer at the output, the network performs K-class classification.

(a) In this fully connected network, how many parameters should be learned by the network (weights and biases)?

(b) Let's assume instead of backpropagation, we decide to use the forward path to calculate the gradients, using the two-sided gradient calculation formula on Slide 11 (Speed of Finite Difference). How many simple operations (addition and multiplications) are required? (Calculate and justify) (You can ignore the Softmax operation).

(c) Derive the feed-forward equation that maps the input to the output in a matrix and vector form.

Question 2 We have a dataset with one million samples. Compare the number of times that each parameter is updated if we train the network for five (5) epochs (even if the input is zero, it is still considered an update), if we use

- Stochastic gradient descent
- Mini-batch gradient descent with a batch size of 128
- (Full) gradient descent

Question 3 For each of the following equations, draw the architecture of the neural network. You need to label each of the nodes, edges, and activation functions.

(a)

$$y = \tanh\big(w_{31} \cdot \text{Leaky ReLU}(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b_1) + w_{32} \cdot \text{Leaky ReLU}(w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b_2) + b_3\big)$$

(b)

$$y = \sigma\Big(w_6 \cdot \sigma\big(w_4 \cdot \text{ReLU}\big(w_2 \cdot \text{ReLU}(w_1 x + b_1) + b_2\big) + w_5 \cdot \sigma\big(w_3 \cdot \text{ReLU}(w_2 \cdot \text{ReLU}(w_1 x + b_1) + b_2) + b_3\big) + b_4\big) + b_5\Big)$$

Question 4 Given the mean squared loss (MSE) $\mathcal{L}(X, w, y) = \frac{1}{2N}\|Xw - y\|^2$ for data matrix $X \in \mathbb{R}^{N \times D}$, weights $w \in R^{D \times 1}$, and outputs $y \in R^{N \times 1}$.

(a) Explain and justify the dimensions of the input, weights, and the output, i.e., what do N, D, stand for, and accordingly justify the dimension of w and y.

(b) Find the expression for gradient $\nabla_w \mathcal{L}(X, w, y)$ and minimizer of this loss, $\arg\min_w \mathcal{L}(X, w, y)$.

(c) Take w_0 as the initialization for gradient descent with step size α and show an expression for the first and second iterates w_1 and w_2 only in terms of α, w_0, X, y.

(d) Generalize this to show an expression for w_k in terms of α, w_0, X, y, k (k is the iteration).

(e) Write a pseudo code for calculating the w_k in terms of α, w_0, X, y, k.

Question 5 Consider the following neural network. The neurons in the hidden layer have a Relu activation. The output layer uses a Sigmoid activation for classification. The loss function is a *cross entropy loss function*.

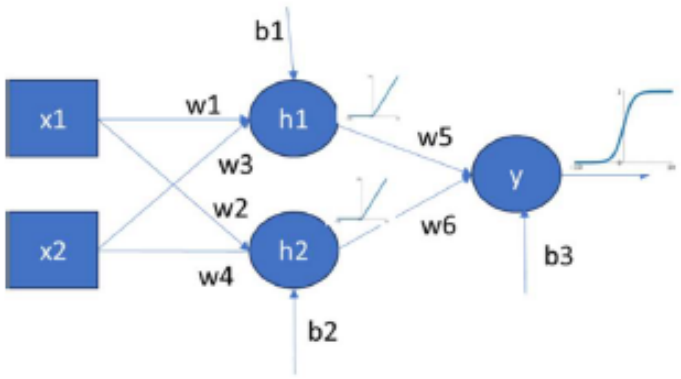


Figure 1: Network with 2 hidden neurons.

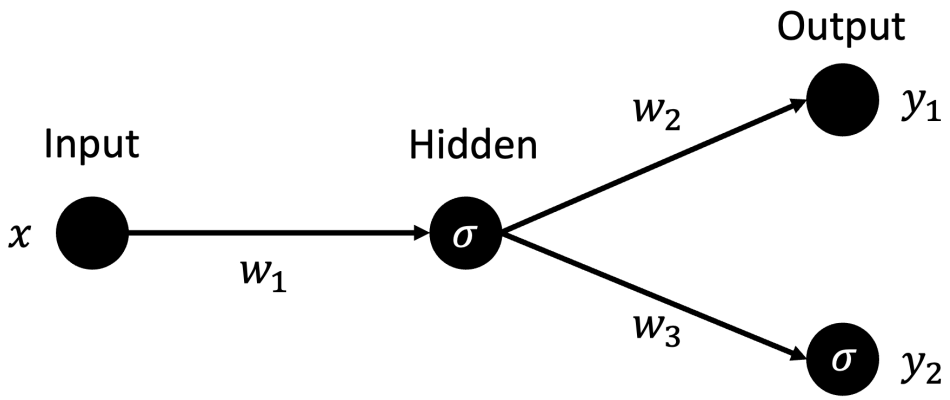
Parameter	Current value
w_1	0.1
w_2	0.2
w_3	0.3
w_4	0.4
w_5	0.5
w_6	0.6
b_1	0.1
b_2	0.2
b_3	0.3

Table 1: Initial (Current) values of parameters

- Calculate the output of y if the network is fed x = (1, 0) as input.

- Assume that the expected output (true value) for the input x = (1, 0) is supposed to be 1. Calculate the loss for the case above.
- Calculate the updated weights for w1, w4, w5, and w6, b1 and b3, after the backpropagation of the error for this sample (we are using stochastic gradient descent). Assume that the learning rate is equal to 0.1.
- What would be the updated value of w5 if instead of SGD, we use Momentum with a momentum term equal to 0.9 and current velocity of 0.5.

Question 6 Consider the following simple network:



Note that σ is the sigmoid activation function. This is a unrealistic simplified example of a multi-task neural network (a network trained to solve multiple tasks at once, sharing the same hidden layers), performing both regression and classification tasks.

(a) Derive an expression for the output*s* of the neural network in terms of the input and the weights.

(b) Assume that you are using the following loss function for regression, $y1$, and a cross-entropy loss function for $y2$:

$$l_{reg} = \sum_{i}^{n} \frac{1}{2}(y1_i - t1_i)^2 \tag{1}$$

$$l_{cls} = \sum_{i}^{n} [t2_i ln(y2_i) + (1 - t2_i) ln(1 - y2_i) \tag{2}$$

$$loss = l_{reg} + l_{cls} \tag{3}$$

where $t1$ and $t2$ are the targets for regression and classification respectively, and n is the number of outputs for a given batch. For this neural network, compute $\frac{\partial l}{\partial w_1}$. (hint: use the chain rule)

(c) Given the table below of inputs and weights, compute the output of the network. Write all the steps.

	Input	Weights
Network 1	$\mathbf{x} = 0.5$	$\mathbf{w} = [0.5, 0.2, 0.1]$

Implementation Questions Use the random seed of 433 anytime needed.

Question 1 (a) We will build a network to classify fashion items available in FashionMNIST available in PyTorch. We will study different ways to initialize the model and different optimizers.
First create a class of type `nn.module`. The input will be 28x28 images from the FashionMNIST dataset (784). The module should allow the construction of a network as follows:

```
net = my_nn_model()
```

Hint: You may use the `nn.ModuleList` construction. Use a depth of 8 and width of 16 hidden units. The input layer of your network should take minibatches of data sized $B \times 784$ where B is the batch size and the output should have 10 values (10 fashion classes). Use ReLU activation function.

(b) Write a function to initialize your model where weights are distributed uniformly between $-d$ ans d, i.e., $w \sim U(-d, d)$ and biases to zero. We will study four (4) cases $d = 0.01, 0.1, 1.0, \sqrt{\frac{6}{n_i+n_o}}$, where n_i, n_o are the number of input and output connections to a unit. Note the final value corresponds to Xavier initialization with n_i, n_o, the number of input and output connections to a unit.

You will perform parts (c) and (d) for all 4 values of d and a depth of 8. This function can be a member of your model class. You may make use of the built-in `torch.nn.init` functions to achieve this.

(c) Using a cross-entropy loss at the end of the network: forward and backward a minibatch of 128 MNIST digits through the network with depth 8. Compute and visualize the gradient norm at each layer. Specifically this refers to the $\left\|\frac{\partial L}{\partial a}\right\|$, where a are the post-activation outputs. Your plots should have a layer on the x-axis and gradient norm on the y-axis. Note that to get the gradient norms at each layer you can use `retain_grad` on the layer outputs in the forward pass to keep the gradient buffer from clearing at each layer on the backward. Perform this for each of the four (4) initializations to obtain four (4) curves. Note: in this question, you do not need to train or update the models.

(d) For each of the initialization settings train the model for 5 epochs on FashionMNIST, using the cross-entropy loss. You may use SGD with a learning rate of 0.01 and minibatch sizes of 128. Record the training accuracy and testing accuracy after each epoch and plot them versus epochs.

(e) Briefly (max 1 paragraph) discuss your findings: how do depth and initialization affect the activations, gradients, and convergence?

(f) Only for the Xavier initialization, train the model for 5 epochs on Fashion-MNIST, using the cross-entropy loss. Use three different optimizers and minibatch sizes of 128: SGD, RMSProp, and Adam, all with a learning rate of 0.01. Record the training accuracy and testing accuracy after each epoch and plot them versus epochs. Briefly (max 1 paragraph) discuss your findings: how does the optimizer affect the performance (if any)?

Question 2 In this task, we will work on a regression task with PyTorch , using *fetch_california_housing* dataset available in *scikit-learn*. Therefore, you need to convert it to Torch format.

(a) Load, convert, and split your dataset into train and test sets with a 75:25 ratio using methods from PyTorch and the random see of 433.

(b) Explore the dataset and apply any required preprocessing. Yo do not need to provide any EDA visualizations.

(c) Implement and train your neural network. Note and plot the performance in terms of batch vs loss across 100 epochs. Analyze the Mean Squared Error (MSE) and Mean Average Error (MAE) on the test set.

(d) It is time to fine-tune your model with the aim of enhancing its performance. Implement a grid search considering the learning rate and batch size parameters. Try variations of learning rates (i.e. 0.1, 0.01, 0.001) and batch sizes (16, 32, 64, etc). Compare the results in terms of training curves and testing performance.