Weekly Homework 1

Dr. Sujit Das
Deep Learning

August 1, 2025

For this assignment, I watched the following videos:

Exercise 1. Matrix-Based Backpropagation Consider a small neural network with:
Input:

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Hidden layer (2 neurons):

$$\mathbf{W}_1 = \begin{bmatrix} 0.1 & 0.2 \\ 0.3 & 0.4 \end{bmatrix}, \quad \mathbf{b}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Output layer:

$$\mathbf{W}_2 = \begin{bmatrix} 0.5 & 0.6 \end{bmatrix}, \quad b_2 = 0$$

Activation function: Sigmoid
Target output:

$$y = 1$$

Tasks:

1. Implement the forward pass and compute the output.
2. Compute the loss using Mean Squared Error (MSE).
3. Perform backpropagation manually (or in code) to compute gradients:

$$\frac{\partial L}{\partial \mathbf{W}_2}, \quad \frac{\partial L}{\partial \mathbf{W}_1}, \quad \frac{\partial L}{\partial b_2}, \quad \frac{\partial L}{\partial \mathbf{b}_1}$$

Exercise 2. . **Softmax and Cross-Entropy Gradient**
Let a neural network output the logits:

$$\mathbf{z} = [2, \ 1, \ 0]$$

and the true class be represented as a one-hot vector:

$$\mathbf{y} = [1, \ 0, \ 0]$$

The softmax function is defined as:

$$\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

and the cross-entropy loss is:

$$L = -\sum_i y_i \log(\hat{y}_i)$$

Tasks:

1. Compute the softmax output $\hat{\mathbf{y}}$.
2. Compute the loss.
3. Derive the gradient $\frac{\partial L}{\partial z_i}$ for each class.
4. Explain why this combination (softmax + cross-entropy) simplifies the gradient expression.

Exercise 3. . ReLU Activation Gradient

A neuron receives input $x = 3$ with weight $w = 0.5$, bias $b = -2$. It uses the ReLU activation function:

$$\mathrm{ReLU}(z) = \max(0, z)$$

The target output is $y = 2$, and the loss is Mean Squared Error (MSE).

Tasks:

1. Compute the pre-activation z, the output of ReLU, and the loss.
2. Compute the gradient of the loss with respect to the weight w.
3. What happens to the gradient if the input $x = 2$ instead of 3?

Exercise 4. . Two Hidden Layers (2-2-1) – Sigmoid Activation

Input:

$$\mathbf{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Layer 1: 2 neurons Weights: Identity matrix

$$\mathbf{W}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{b}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Layer 2: 1 neuron Weights:

$$\mathbf{W}_2 = \begin{bmatrix} 1 & 1 \end{bmatrix}, \quad b_2 = 0$$

Activation: Sigmoid in both hidden and output layers

Target output:

$$y = 1$$

Tasks:

1. Perform forward propagation through the network.

2. Compute the final output and the loss (Mean Squared Error, MSE).

3. Perform backpropagation step-by-step to find:

$$\frac{\partial L}{\partial \mathbf{W}_2}, \quad \frac{\partial L}{\partial \mathbf{W}_1}$$

Exercise 5. . Binary Cross-Entropy Loss with Sigmoid

A neuron has:

Input:

$$x = 4, \quad w = 0.5, \quad \text{bias} = 0$$

Activation: Sigmoid

Target output:

$$y = 0$$

Loss: Binary cross-entropy

$$L = -\left[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\right]$$

Tasks:

1. Perform forward pass to compute $\hat{y}$.

2. Compute the loss.

3. Derive the gradient $\frac{\partial L}{\partial w}$ manually using the chain rule.

Guidelines for the students

Objective

To reinforce your understanding of forward and backward propagation, gradients, and neural network computations using simple neural network structures and loss functions.

Instructions

1. Attempt all questions unless specified otherwise.

2. Show all steps clearly:

   - Forward pass calculations.
   - Loss computation.
   - Backpropagation with proper chain rule application.

3. Include all formulas used in each step.
4. Use neat handwriting (if handwritten) or typeset (LaTeX preferred, optional).
5. You may use `NumPy` for code-based questions, but all steps must still be explained clearly.
6. **No direct ChatGPT or AI-generated answers allowed. Use your own understanding.**
7. Submit the assignment as a single PDF file (scanned if handwritten).

Submission Deadline

[4-08-2025]
Late submissions will incur a penalty of 2 marks per day unless prior approval is given.

Marking Scheme Per Question (10 Marks Each)

Q1: Forward Pass Computation

Step-by-step calculation of weighted sum	2
Activation function application (if any)	2
Correct loss computation	2
Gradient with respect to weights or bias	3
Final boxed answer with unit/notation	1
Total	**10**

Q2: Hidden Layer Gradient Calculation

Forward pass through hidden & output layers	3
Use of activation functions (e.g., sigmoid)	1
Correct application of chain rule	3
Final gradients for weights in both layers	2
Neatness and logical flow	1
Total	**10**

Q3: Matrix-Based Backpropagation (NumPy)

Matrix forward pass computations	2
Loss evaluation using predicted output	2
Manual or code-based gradient derivation	4
Explanation of each gradient step	1
Organized output (comments or explanations)	1
Total	**10**

Q4: Softmax + Cross-Entropy Derivation

Correct softmax computation	2
Cross-entropy loss derivation	2
Use of derivative: $\hat{y} - y$ trick	3
Explanation of why it simplifies	2
Clarity and notation	1
Total	**10**

Q5: ReLU and Custom Loss Backpropagation

ReLU application and handling gradient cases	2
Custom loss formula and correct derivation	3
Gradient calculation for weight/bias	3
Gradient update with learning rate	1
Boxed final answer and steps shown	1
Total	**10**

Overall Homework Total: 50 Marks

Criteria	**Description**	**Marks**
Q1	Forward pass + weight update	10
Q2	Multi-layer backpropagation	10
Q3	Matrix-based computation	10
Q4	Softmax + cross-entropy	10
Q5	ReLU + custom loss gradients	10
Grand Total		**50**

Academic Honesty

Plagiarism or copying from peers will result in **zero marks** for the assignment. You may discuss concepts, but **write your own steps and reasoning**.

Submission Format Example

- **Page 1:** Name, Roll No., Assignment Title
- **Pages 2–n:** Answers with clearly labeled question numbers
- Highlight important steps or box final answers
- If coding: attach output screenshots and explain in 2–3 lines