Birla Institute of Technology & Science, Pilani
Work Integrated Learning Programmes Division
First Semester 2024-2025

Mid-Semester Test
(EC-2 Regular)

Course No.	: AIMLCZG511
Course Title	: Deep Neural Network
Nature of Exam	: Closed Book
Weightage	: 30%
Duration	: 2 Hours
Date of Exam	: 19-01-2025 (AN)

No. of Pages = 03
No. of Questions = 06

Note to Students:
1. Please follow all the *Instructions to Candidates* given on the cover page of the answer book.
2. All parts of a question should be answered consecutively. Each answer should start from a fresh page.
3. Assumptions made if any, should be stated clearly at the beginning of your answer.

Q.1. Given the following truth table for the NOR gate: **[5 Marks]**

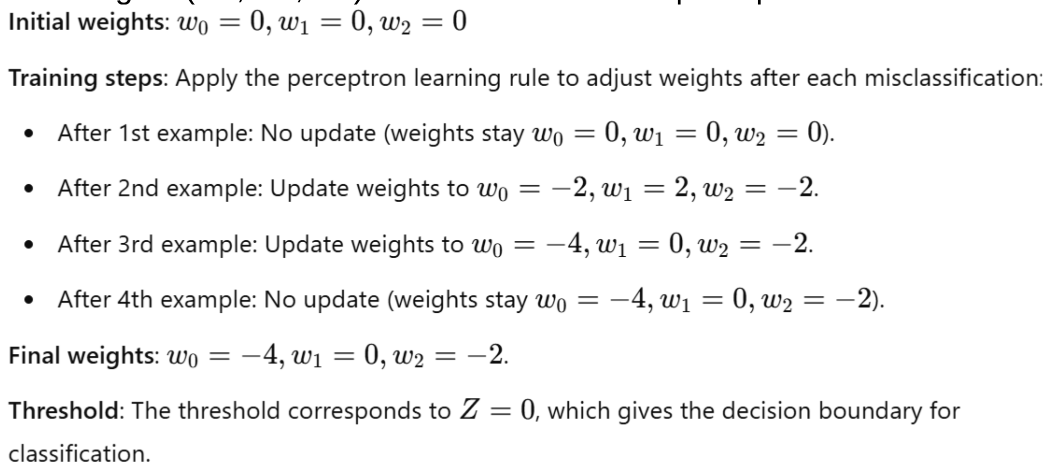
X_1	X_2	**Y** = NOR(X_1,X_2)
-1	-1	1
-1	1	-1
1	-1	-1
1	1	-1

Assume the initial weights $w_0=w_1=w_2=0$, learning rate η=1, the activation function is h=sign(Z), where Z = w_0 + $w_1(x_1)$ + $w_2(x_2)$, and the output Y is computed as Y=sign(Z).

1. Derive the weights (w_0,w_1,w_2), and the threshold for the perceptron model after applying the perceptron learning algorithm.
2. Verify that the perceptron correctly classifies all inputs
3. Provide a diagram of the perceptron to support your solution

Answer Key and Rubrics:

1. Derive the weights (w0, w1, w2) and threshold for the perceptron model:



Initial weights: $w_0 = 0, w_1 = 0, w_2 = 0$

Training steps: Apply the perceptron learning rule to adjust weights after each misclassification:

- After 1st example: No update (weights stay $w_0 = 0, w_1 = 0, w_2 = 0$).
- After 2nd example: Update weights to $w_0 = -2, w_1 = 2, w_2 = -2$.
- After 3rd example: Update weights to $w_0 = -4, w_1 = 0, w_2 = -2$.
- After 4th example: No update (weights stay $w_0 = -4, w_1 = 0, w_2 = -2$).

Final weights: $w_0 = -4, w_1 = 0, w_2 = -2$.

Threshold: The threshold corresponds to $Z = 0$, which gives the decision boundary for classification.

2. Verify that the perceptron correctly classifies all inputs:

Input 1: $X_1 = -1, X_2 = -1$, predicted output: $\hat{y} = -1$, expected: $Y = 1$ (misclassified).

Input 2: $X_1 = -1, X_2 = 1$, predicted output: $\hat{y} = -1$, expected: $Y = -1$ (correct).

Input 3: $X_1 = 1, X_2 = -1$, predicted output: $\hat{y} = -1$, expected: $Y = -1$ (correct).

Input 4: $X_1 = 1, X_2 = 1$, predicted output: $\hat{y} = -1$, expected: $Y = -1$ (correct).

Q.2. Answer the following: [**5 Marks]**

1. Given a fully connected neural network with an input dimension of 15, there are 4 hidden layers each with 8 hidden units, and a scalar output, compute the total number of trainable parameters in the network.
2. In the Adam optimizer, β1 and β2 are nonnegative weighting parameters, typically set to 0.9 and 0.999 respectively. Briefly explain the role of β1 and β2 in Adam and how they influence the first and second moment estimates.
3. In a binary classification task to distinguish between a dog and a cat, and a multi-class classification task to classify an image into one of 5 animal categories, what activation functions and loss functions would be used for each task? Explain your reasoning.
4. If a neural network has 6 nodes and each node has a 50% chance of being retained or dropped using the dropout technique, how many different thinned networks can be formed?
5. In a neural network, what are the consequences of using a too-large initialization and a too-small initialization for the weights? Explain how each affects the network's performance during training.

(**Note:** *Answer/Justify/Illustrate in no more than 30-50 words and precise response w.r.t given question only. Vague answers will be penalized. Augment simple example for your illustration)*

Answer key and rubrics

1. Compute the total number of trainable parameters in the given fully connected neural network.

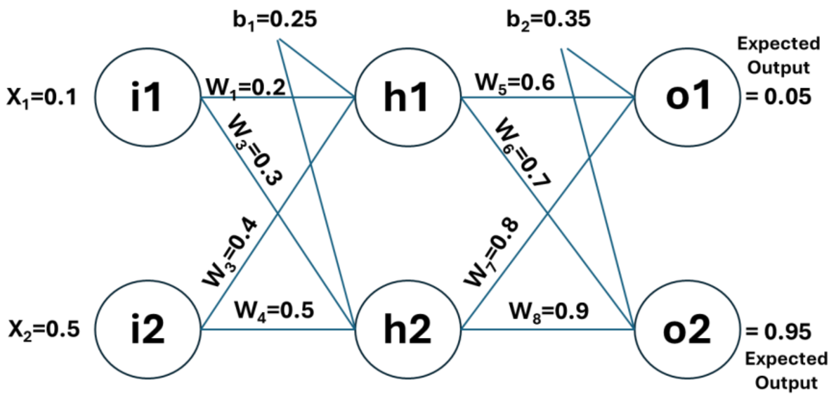
   Total trainable parameters:

$$128 + 72 + 72 + 72 + 9 = 353$$

2. Role of β1 and β2 in adam optimizer
   - β1 (typically 0.9) controls how much the running average of the gradients is influenced by past gradients.
   - **β2** (typically 0.999) controls how much the running average of squared gradients is influenced by past squared gradients.
3. Activation functions and loss functions for binary and multi-class classification.
   - Binary classification (dog vs. cat):
     - Activation function: Sigmoid
     - Loss function: Binary cross-entropy
   - Multi-class classification (5 animal categories):
     - Activation function: Softmax
     - Loss function: Categorical cross-entropy
4. Number of different thinned networks using dropout.
   - For each node, there are 2 choices (retained or dropped). Thus, the total number of possible thinned networks is: 64

5. Consequences of too-large and too-small weight initialization.
    - Too-large initialization: Causes exploding gradients, where the gradients become very large and cause the model's weights to grow excessively, resulting in unstable training.
    - Too-small initialization: Causes vanishing gradients, where the gradients become very small, making learning slow or halting.

Q.3. Consider the network below: **[5 Marks]**



1. Compute the total error that has to be backpropagated through the network using the sigmoid activation function.
2. Find the output of the network if a softmax layer is introduced after O_1 and O_2

Answer Key and Rubrics

1. Compute Total Error using Sigmoid Activation (2.5M)

$$H1 = \sigma(w1 \cdot x1 + w2 \cdot x2 + b1) = \sigma(0.2 \times 0.1 + 0.3 \times 0.5 + 0.25)$$

$$H2 = \sigma(w3 \cdot x1 + w4 \cdot x2 + b1) = \sigma(0.4 \times 0.1 + 0.5 \times 0.5 + 0.25)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$$H1 = 0.6035, \quad H2 = 0.6318$$

Then,

$$O1 = \sigma(w5 \cdot H1 + w6 \cdot H2 + b2)$$

$$O2 = \sigma(w7 \cdot H1 + w8 \cdot H2 + b2)$$

$$O1 = 0.7603, \quad O2 = 0.8024$$

2. Compute Error Using Mean Squared Error (MSE) **(2.5M)**

$$E = \frac{1}{2}[(0.05 - 0.7603)^2 + (0.95 - 0.8024)^2]$$

$$E = \frac{1}{2}(0.5045 + 0.0218) = \frac{0.5263}{2} = 0.2632$$

Q.4. Consider the following equation in a neural network,

g = (ax_1+bx_2) W_1+W_2,

Where a=1.5, b=2.5, x_1=0.2, x_2=0.4, w_1=0.1, and w_2=0.5. **[5 Marks]**

1. Draw a computational graph to illustrate the flow of computations involved in determining the output g.
2. Write down the equations for computing the output and the gradient with respect to w_1, w2, a, and b.
3. Illustrate how these gradients will be used in updating the weights during the training of a neural network.

Answer Key and Rubrics

Equations for computing the output and gradients **(2M)**

Output Equation:

$$g = (ax_1 + bx_2)W_1 + W_2$$

Gradients Computation (Using Chain Rule):

1. **Gradient with respect to W_1:**

$$\frac{\partial g}{\partial W_1} = (ax_1 + bx_2) = 1.3$$

2. **Gradient with respect to W_2:**

$$\frac{\partial g}{\partial W_2} = 1$$

3. **Gradient with respect to a:**

$$\frac{\partial g}{\partial a} = x_1W_1 = 0.2 \times 0.1 = 0.02$$

4. **Gradient with respect to b:**

$$\frac{\partial g}{\partial b} = x_2W_1 = 0.4 \times 0.1 = 0.04$$

Updating the weights using gradients **(1M)**

Weights are updated using **gradient descent** with a learning rate α:

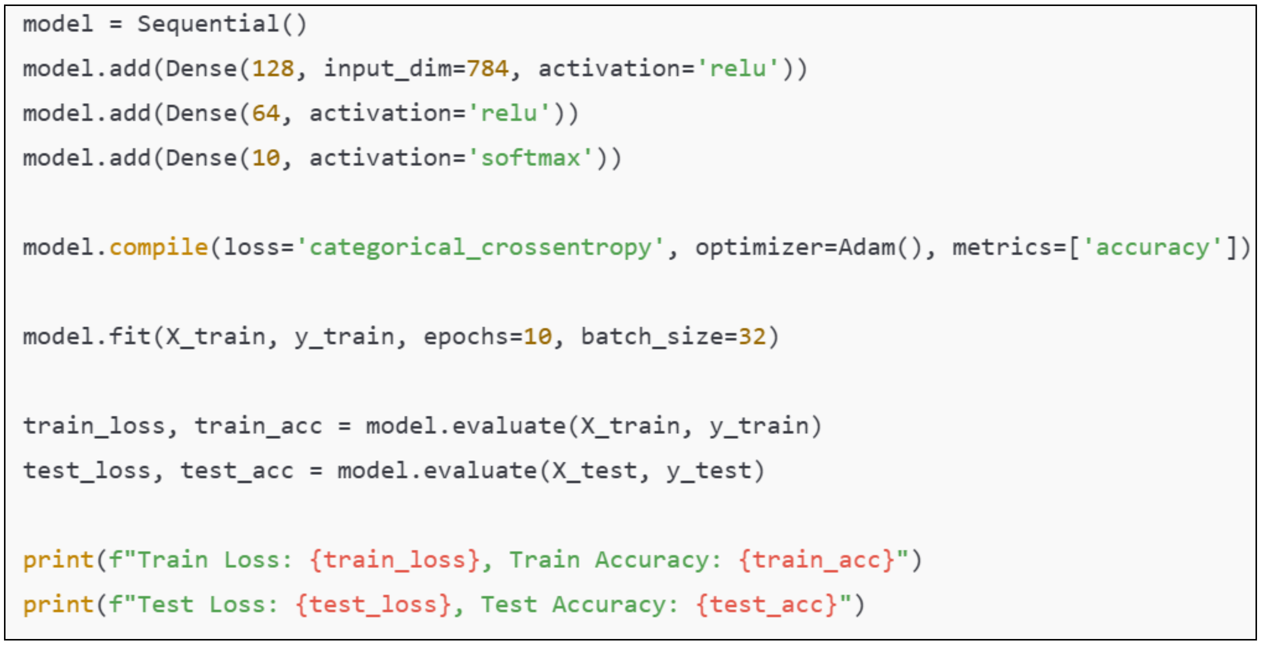
$$W_1^{new} = W_1 - \alpha\frac{\partial g}{\partial W_1}$$

$$W_2^{new} = W_2 - \alpha\frac{\partial g}{\partial W_2}$$

$$a^{new} = a - \alpha\frac{\partial g}{\partial a}$$

$$b^{new} = b - \alpha\frac{\partial g}{\partial b}$$

Q.5. Consider the following Python code snippet for implementing a Deep Neural Network (DNN) using the Keras library on the MNIST dataset: **[5 Marks]**



```
model = Sequential()
model.add(Dense(128, input_dim=784, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(10, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer=Adam(), metrics=['accuracy'])

model.fit(X_train, y_train, epochs=10, batch_size=32)

train_loss, train_acc = model.evaluate(X_train, y_train)
test_loss, test_acc = model.evaluate(X_test, y_test)

print(f"Train Loss: {train_loss}, Train Accuracy: {train_acc}")
print(f"Test Loss: {test_loss}, Test Accuracy: {test_acc}")
```

1. Define the architecture of the neural network used in the given code. What is the role of each layer.
2. After training the model, the train accuracy is very high, but the test accuracy is significantly lower. What does this indicate? How would you define training error and generalization error in this context?
3. Suppose the activation function in the output layer is mistakenly changed to sigmoid instead of softmax. What impact would this have on the model's performance, and how would you correct it?
4. Suggest two possible improvements or modifications you could make to the model.
5. Justify the choice of the Adam optimizer in this code, and what other optimizers could be used as alternatives?

(**Note:** *Answer/Justify/Illustrate in no more than 30-50 words and precise response w.r.t given question only. Vague answers will be penalized. Augment simple example for your illustration)*

Answer Key and Rubrics

1. Define the Architecture of the Neural Network **(1M)**

- Input Layer: 784 neurons (corresponding to flattened 28×28 MNIST images).
- Hidden Layer 1: 128 neurons, ReLU activation – captures complex patterns.
- Hidden Layer 2: 64 neurons, ReLU activation – further feature extraction.
- Output Layer: 10 neurons, Softmax activation – outputs class probabilities for digits (0-9)

2. Interpretation of High Train Accuracy but Low Test Accuracy **(1M)**
    - It indicates **overfitting**, where the model memorizes training data but fails to generalize to unseen test data.
    - **Training error**: Difference between predicted and actual labels in the training set.
    - **Generalization error**: Difference between predictions on unseen test data and actual labels.
3. Effect of Changing Output Activation from Softmax to Sigmoid **(1M)**
    - Issue: Sigmoid outputs independent probabilities for each class instead of a probability distribution summing to 1.
    - Impact: The model may struggle with multiclass classification, leading to poor performance.
    - Correction: Use Softmax, which normalizes outputs into probabilities across all 10 classes.
4. Two Possible Model Improvements **(1M)**
    - Add Dropout (e.g., 20-50%) in hidden layers to reduce overfitting.
    - Use Batch Normalization to stabilize training and improve convergence speed.
5. Justification of Adam Optimizer & Alternatives **(1M)**
    - Why Adam? Combines momentum & adaptive learning rates, leading to faster convergence.
    - Alternatives:
        - SGD – Slower but may generalize better.
        - RMSprop – Good for non-stationary data.

Q.6. Answer the following: [**5 Marks]**

1. What is the purpose of the bias term in a neural network? How does the bias term contribute to the network's ability to learn and make predictions? Provide a brief explanation of how biases are incorporated into the network's computations.
2. Explain the concept of vanishing gradients in the context of neural networks. Discuss the potential consequences of vanishing gradients during training and how this issue can impact the performance of a neural network. Provide at least one technique or architectural modification that can be used to mitigate the vanishing gradient problem.

(**Note:** *Answer/Justify/Illustrate in no more than 30-50 words and precise response w.r.t given question only. Vague answers will be penalized. Augment simple example for your illustration)*

Answer key and Rubrics

1. Purpose of Bias Term in Neural Networks **(2.5 M)**
    - The bias term allows a neural network to shift the activation function, enabling better learning of patterns that do not pass through the origin. It helps the model fit data more flexibly by allowing activation thresholds to be adjusted.

$$y = f(w_1x_1 + w_2x_2 + ... + w_nx_n + b)$$

2. Vanishing Gradients in Neural Networks **(2.5 M)**
    - Vanishing gradients occur when backpropagation causes gradients to shrink as they propagate through deep layers, slowing or stopping weight updates.

- This leads to poor learning, especially in deep networks with sigmoid or tanh activation functions.
- Example Issue: In a deep network using sigmoid activation, if outputs are near 0 or 1, derivatives become very small, reducing weight updates.
- Solution: Using ReLU activation f(x)=max(0,x)) mitigates vanishing gradients as it maintains gradients in positive regions.

****