

# Deep Learning Assignment-01

MTech (CS), IIIT Bhubaneswar  
March - 2025



Student ID: A124005 Student Name: Milan Patel
--

## Solutions of Deep Learning Assignment 01

1. : For a  $D$ -dimensional input vector, show that the optimal weights can be represented by the expression: 1

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{t}$$

What is the possible estimation of  $\mathbf{w}$ ?

### **Solution:**

To derive the optimal weights  $\mathbf{w}$  for a linear regression problem, we start with the least squares objective. Given a dataset with  $N$  samples, where  $\mathbf{X}$  is the  $N \times D$  design matrix (each row corresponds to a  $D$ -dimensional input vector),  $\mathbf{t}$  is the  $N \times 1$  target vector, and  $\mathbf{w}$  is the  $D \times 1$  weight vector, the goal is to minimize the sum of squared errors:

$$E(\mathbf{w}) = \|\mathbf{t} - \mathbf{X}\mathbf{w}\|^2$$

Expanding the squared error term:

$$E(\mathbf{w}) = (\mathbf{t} - \mathbf{X}\mathbf{w})^T (\mathbf{t} - \mathbf{X}\mathbf{w})$$

Taking the derivative of  $E(\mathbf{w})$  with respect to  $\mathbf{w}$  and setting it to zero for minimization:

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{X}^T (\mathbf{t} - \mathbf{X}\mathbf{w}) = 0$$

Rearranging the equation:

$$\mathbf{X}^T \mathbf{t} - \mathbf{X}^T \mathbf{X} \mathbf{w} = 0$$

Solving for  $\mathbf{w}$ :

$$\mathbf{X}^T \mathbf{X} \mathbf{w} = \mathbf{X}^T \mathbf{t}$$

Assuming  $\mathbf{X}^T \mathbf{X}$  is invertible, the optimal weight vector  $\mathbf{w}$  is:

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{t}$$

This is the least squares solution for the weight vector  $\mathbf{w}$ .

## Estimation of $\mathbf{w}$

The expression  $\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{t}$  provides the optimal weights that minimize the sum of squared errors between the predicted values  $\mathbf{X}\mathbf{w}$  and the target values  $\mathbf{t}$ . This is the best linear unbiased estimator (BLUE) under the assumptions of linear regression (e.g., no multicollinearity, homoscedasticity, and normally distributed errors).

Thus, the estimation of  $\mathbf{w}$  is given by:

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{t}$$

## 2. : OR Gate in single neural network

**Solution:**

## OR Gate Using a Single-Layer Neural Network

### Perceptron Model

A perceptron computes a weighted sum of its inputs and applies an activation function:

$$y = f(w_1 x_1 + w_2 x_2 + b)$$

Where:

- $x_1, x_2$  are the inputs
- $w_1, w_2$  are the weights
- $b$  is the bias
- $f(z)$  is the activation function (step function):

The OR gate can be implemented using a single-layer neural network (perceptron) with two inputs  $x_1$  and  $x_2$ , weights  $w_1$  and  $w_2$ , and a bias  $b$ . The output  $y$  of the perceptron is given by:

$$f(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

## Truth Table for OR Gate

The truth table for the OR gate is:

$x_1$	$x_2$	$t$
0	0	0
0	1	1
1	0	1
1	1	1

## Determining Weights and Bias

We need to find  $w_1$ ,  $w_2$ , and  $b$  such that the perceptron correctly classifies the inputs. Let us choose the following values:

$$w_1 = 1, \quad w_2 = 1, \quad b = -0.5$$

## Verification

Now, we verify the perceptron's output for each input combination:

1. For  $x_1 = 0, x_2 = 0$ :

$$w_1x_1 + w_2x_2 + b = (1)(0) + (1)(0) + (-0.5) = -0.5 < 0 \implies y = 0$$

2. For  $x_1 = 0, x_2 = 1$ :

$$w_1x_1 + w_2x_2 + b = (1)(0) + (1)(1) + (-0.5) = 0.5 \geq 0 \implies y = 1$$

3. For  $x_1 = 1, x_2 = 0$ :

$$w_1x_1 + w_2x_2 + b = (1)(1) + (1)(0) + (-0.5) = 0.5 \geq 0 \implies y = 1$$

4. For  $x_1 = 1, x_2 = 1$ :

$$w_1x_1 + w_2x_2 + b = (1)(1) + (1)(1) + (-0.5) = 1.5 \geq 0 \implies y = 1$$

## Conclusion

The weights  $w_1 = 1$ ,  $w_2 = 1$ , and bias  $b = -0.5$  correctly implement the OR gate using a single-layer neural network. The perceptron's output matches the truth table for all input combinations.

$$w_1 = 1, \quad w_2 = 1, \quad b = -0.5$$

3. :for given graph give the following solutions

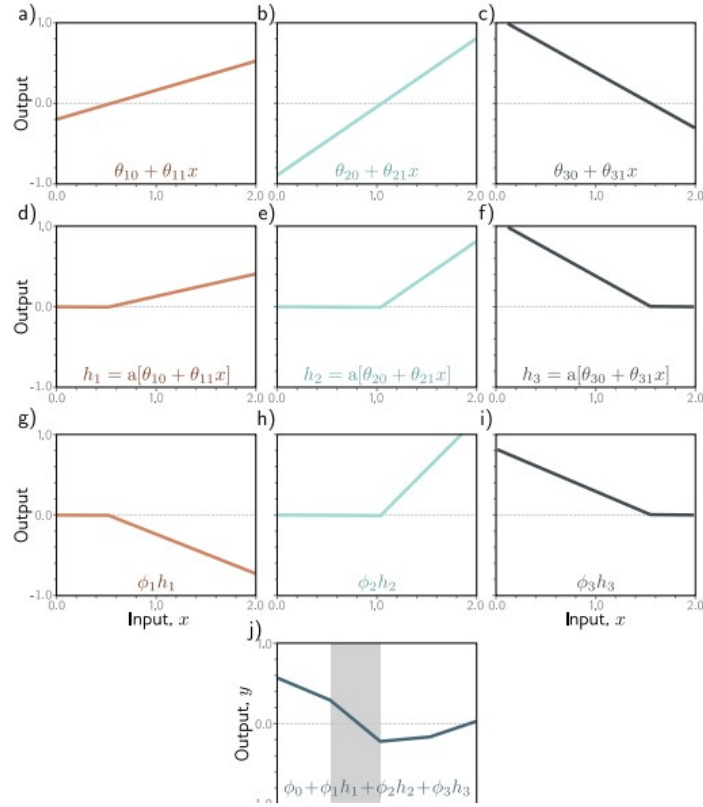


Figure 1: generalization of intersection

(a) :Generalized Point of Intersection for Shallow Neural Networks for input space parameterized by spherical coordinates  $\theta$  and  $\phi$

**Solution:**

**Generalizing the Point of Intersection in Terms of  $\theta$  and  $\phi$  for Shallow Neural Networks**

**Step 1: Structure of a Shallow Neural Network**

Consider a shallow neural network with:

- Input dimension:  $d$
- Number of hidden neurons:  $m$
- Activation function:  $\sigma$
- Weight vectors:  $\mathbf{w}_i \in \mathbb{R}^d$
- Bias terms:  $b_i \in \mathbb{R}$
- Output weights:  $a_i \in \mathbb{R}$

The output of the network is given by:

$$f(\mathbf{x}) = \sum_{i=1}^m a_i \sigma(\mathbf{w}_i^T \mathbf{x} + b_i)$$

**Step 2: Weight Vectors in Angular Coordinates**

In spherical coordinates:

$$\mathbf{w} = \|\mathbf{w}\| \begin{bmatrix} \sin(\theta) \cos(\phi) \\ \sin(\theta) \sin(\phi) \\ \cos(\theta) \end{bmatrix}$$

**Step 3: Decision Boundary Condition**

For each neuron, the decision boundary satisfies:

$$\mathbf{w}_i^T \mathbf{x} + b_i = 0,$$

which in spherical coordinates becomes:

$$\|\mathbf{w}_i\| [x_1 \sin(\theta_i) \cos(\phi_i) + x_2 \sin(\theta_i) \sin(\phi_i) + x_3 \cos(\theta_i)] + b_i = 0$$

**Step 4: Intersection of Decision Boundaries**

If two neurons intersect, we solve the system:

$$\mathbf{w}_i^T \mathbf{x} + b_i = 0, \quad \mathbf{w}_j^T \mathbf{x} + b_j = 0,$$

which translates to:

$$\|\mathbf{w}_i\| \mathbf{x} \cdot \mathbf{v}(\theta_i, \phi_i) + b_i = 0, \quad \|\mathbf{w}_j\| \mathbf{x} \cdot \mathbf{v}(\theta_j, \phi_j) + b_j = 0$$

**Step 5: General Solution**

The point of intersection  $\mathbf{x}$  can be computed by solving the linear system:

$$\mathbf{x} = \mathbf{A}^{-1} \mathbf{b},$$

where  $\mathbf{A}$  is the matrix formed by the weight directions in spherical coordinates, and  $\mathbf{b}$  is the bias vector.

- (b) Give the equation of 4 line segments in the graph in terms of  $\theta_1, \theta_2, \theta_3$ , etc., for the figure.

**Solution:**

Consider a shallow neural network with three hidden units and ReLU activations. Let the output  $y$  of the network be defined by the following equation:

$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$

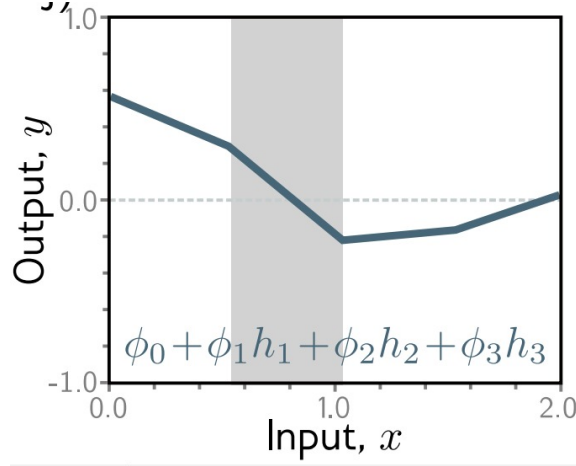


Figure 2: 4 line equations

where each hidden unit  $h_i$  is given by the ReLU activation function:

$$h_i = a(\theta_{i0} + \theta_{i1}x) = \max(0, \theta_{i0} + \theta_{i1}x)$$

The output  $y(x)$  is composed of four linear segments, which can be written as:

$$y(x) = \begin{cases} \phi_0, & x < x_1 \\ \phi_0 + \phi_1(\theta_{10} + \theta_{11}x), & x_1 \leq x < x_2 \\ \phi_0 + \phi_1(\theta_{10} + \theta_{11}x) + \phi_2(\theta_{20} + \theta_{21}x), & x_2 \leq x < x_3 \\ \phi_0 + \phi_1(\theta_{10} + \theta_{11}x) + \phi_2(\theta_{20} + \theta_{21}x) + \phi_3(\theta_{30} + \theta_{31}x), & x \geq x_3 \end{cases}$$

Explicitly, the four line segments are:

- First segment:  $y = \phi_0$
- Second segment:  $y = \phi_0 + \phi_1(\theta_{10} + \theta_{11}x)$
- Third segment:  $y = \phi_0 + \phi_1(\theta_{10} + \theta_{11}x) + \phi_2(\theta_{20} + \theta_{21}x)$
- Fourth segment:  $y = \phi_0 + \phi_1(\theta_{10} + \theta_{11}x) + \phi_2(\theta_{20} + \theta_{21}x) + \phi_3(\theta_{30} + \theta_{31}x)$

The activation thresholds  $x_1$ ,  $x_2$ , and  $x_3$  where each hidden unit is activated are given by:

$$x_i = -\frac{\theta_{i0}}{\theta_{i1}}, \quad \text{for each neuron.}$$

The output function combines the contributions of all active hidden units according to their weights and is expressed in the above piecewise form.

4. Let  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$  be independent and identically distributed (i.i.d.) vectors from a multivariate normal distribution:

$$\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

where  $\boldsymbol{\mu}$  is the unknown mean vector and  $\Sigma$  is the known covariance matrix.

**Solution:**

## Maximum Likelihood Estimate of Unknown Mean Vector

Let  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$  be independent and identically distributed (i.i.d.) vectors from a multivariate normal distribution:

$$\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

where  $\boldsymbol{\mu}$  is the unknown mean vector and  $\Sigma$  is the known covariance matrix.

The probability density function (PDF) of  $\mathbf{x}_i$  is given by:

$$f(\mathbf{x}_i|\boldsymbol{\mu}) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}_i - \boldsymbol{\mu})\right)$$

## Likelihood Function

Given the independence of the samples, the likelihood function is the product of the individual densities:

$$L(\boldsymbol{\mu}) = \prod_{i=1}^n f(\mathbf{x}_i|\boldsymbol{\mu})$$

Taking the natural logarithm of the likelihood function (log-likelihood):

$$\log L(\boldsymbol{\mu}) = -\frac{np}{2} \log(2\pi) - \frac{n}{2} \log |\Sigma| - \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i - \boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}_i - \boldsymbol{\mu})$$

## Maximizing the Log-Likelihood

To find the MLE of  $\boldsymbol{\mu}$ , we differentiate  $\log L(\boldsymbol{\mu})$  with respect to  $\boldsymbol{\mu}$  and set the result to zero:

$$\frac{\partial \log L}{\partial \boldsymbol{\mu}} = \sum_{i=1}^n \Sigma^{-1}(\mathbf{x}_i - \boldsymbol{\mu}) = 0$$

Simplifying:

$$\sum_{i=1}^n (\mathbf{x}_i - \boldsymbol{\mu}) = \mathbf{0} \Rightarrow n\hat{\boldsymbol{\mu}} = \sum_{i=1}^n \mathbf{x}_i$$

**Result: MLE of Mean Vector**

$$\hat{\boldsymbol{\mu}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$$

Thus, the maximum likelihood estimate of the unknown mean vector is the sample mean.