

1 Foreword

Author : Anderson Chau

Disclaimer : The notes are written only for my understanding and memorization purpose after I have self-studied those online lecture notes.

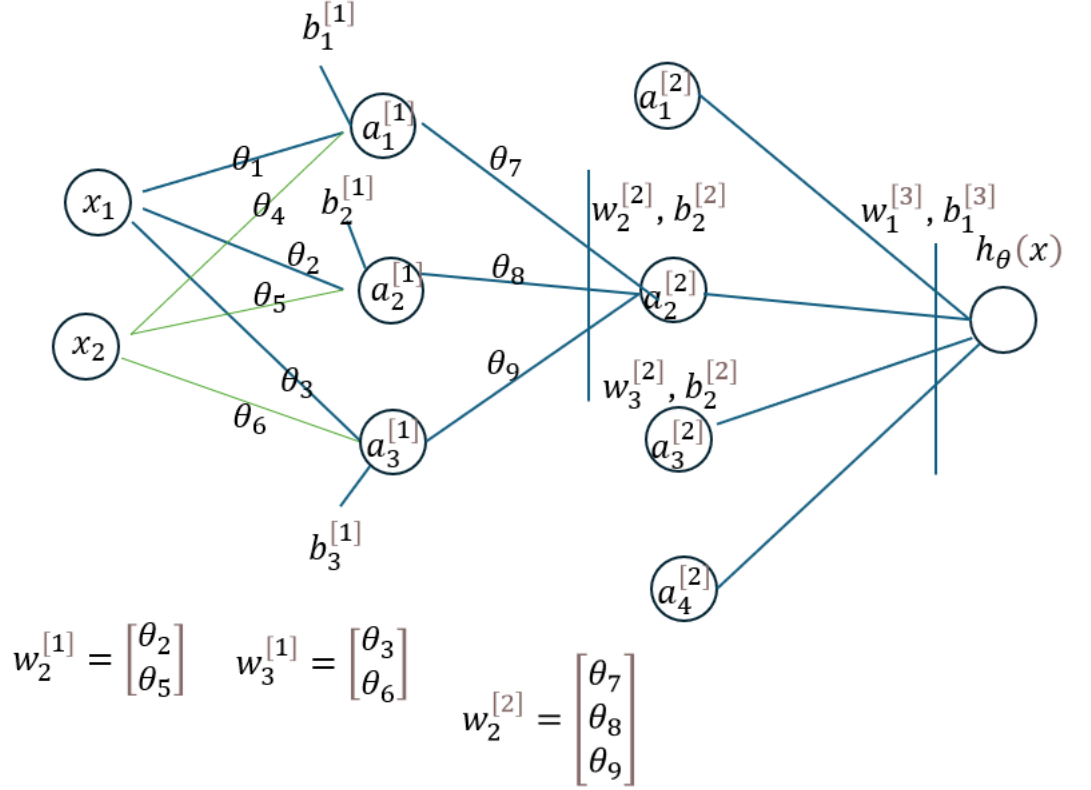
2 Main Idea

The old trick : (i) Forward feed (ii) Compute Loss (NN output vs training data) (iii) Backpropagation with Gradient Descent to tune parameters

2. Use activation functions to avoid the model to be a pure linear model, which is useless (just $ax+b$)

3. Examples of activation functions : Sigmoid , (Leaky) ReLU, tanh etc.

3 ANN Structure



w's and b's are parameters : Totally ther are Hidden Layer 1 (layer 1) (2x3 + 3) + Hidden Layer 1 (layer 2) (3x4 + 4) + Output Layer (layer 3)(4x1 +1) number of parameters

4 Forward Feed (see above NN diagram)

$$a_1 = \text{ReLU}(\theta_1 x_1 + \theta_4 x_2 + b_1^{[1]})$$

$$a_2 = \text{ReLU}(\theta_2 x_1 + \theta_5 x_2 + b_2^{[1]})$$

Let i be i^{th} layer and j be j^{th} neuron (count vertifically) this layer of NN
 Rewriting the notation : $w_j^{[i]}$ as **VECTOR** of input θ 's for layer i and j^{th} neuron.

With the above, Layer 1 of NN can be expressed as:

For all $j \in [1, \dots, m]$: (m=3, count vertically, is the number of number of

neuron layer 1)

$$z_j = \mathbf{w}_j^{[1]\top} \mathbf{x} + b_j^{[1]} \quad \text{where} \quad \mathbf{w}_j^{[1]} \in R^d, b_j^{[1]} \in R$$

(d = 2 , count vertically, is the number of number of previous layer , layer 0)

$$a_j = \text{ReLU}(z_j),$$

$$\mathbf{a} = [a_1, \dots, a_m]^\top \in R^m$$

$$\bar{h}_\theta(\mathbf{x}) = \mathbf{w}^{[3]\top} \mathbf{a} + b^{[3]} \quad \text{where} \quad \mathbf{w}^{[3]} \in R^n, b^{[3]} \in R$$

(n = 4, count vertically, is the number of number of neuron layer, layer 2)

5 Vectorization of Forward Feeding equations

6 Vectorized Backpropagation

It is just matrix calculus + chain rule + gradient descent combined.