CSE465/L-09/16.02.2025/

⊛ Model Selection:

- model should be generalized
  - for all type data
    - train
    - test
    - validation ⇒ unseen data

- model must do well on both train as well as 'unseen' data.

⊛ - good on train set ~~but~~
- not good on unseen data

} over-fit model

⊛ -does not work well on both train and unseen data ⇒ under-fit model

⊛ we need to test our model with unseen data ~~to~~ for validation.

- divide the data set into two part
  - train set
  - test set / validation set

$\underbrace{\text{test set / validation set}}_{\text{most of the time same}}$

⇒ but its better to use three part

- train set ⇒ create model
- validation set ⇒ select model
- test set ⇒ evaluate model, test accuracy

$\boxed{\text{Slide-2}}$ ***

Multi Layer Perceptron (MLP)

MLP Notation:

⇒ Bias notation:

b_{ij} ⇒ i ⇒ layer number, starts from 1
j = index of neuron in layer 'i'

⇒ Output notation:

O_{ij} } i ⇒ layer number
j ⇒ index of neuron in layer 'i'

⇒ Weight notation:

W_{ij}^{k} } k ⇒ target layer number
k-1 to k
i ⇒ index of neuron in layer 'k-1'
j ⇒ index of neuron in layer 'k'

Number of trainable parameters:

Σ for each layer ⇒ weight = number of nodes in previous layer * current layer

bias = number of nodes in current layer

Input layer:

X_{ij} } i ⇒ row number
j ⇒ number of feature

L-10/ 18.02.2025/

For MLP:

combined probability = $\sum$ probability of each perceptron

$= z \Rightarrow$ most of the cases greater than 1.

$= \sigma(z) \Rightarrow$ apply sigmoid function

$\Rightarrow$ without weight & bias

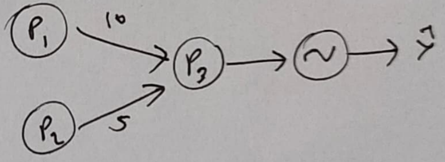
$$z = P_1(y) + P_2(y)$$
$$= 0.8 + 0.7$$
$$= 1.5 > 1$$

$$\sigma(z) = 0.82 \leq 1$$

$\Rightarrow$ with weight & bias:

$$z = \underbrace{10}_{\text{weight}}(0.8) + \overbrace{5}^{\text{weight}}(0.7) + \underbrace{3}_{\text{bias}}$$

$$\sigma(z) = \dots$$



$\boxed{\text{Slide- 17, 18}}$ ***

Loss Function

Loss function:

- for a single sample
- $L = (\hat{y}_i - y_i)^2$

Cost function:

- for entire datasets
- $J = \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2$

~~Loss~~ Loss function!

- mathematical function that quantifies how well a model's predictions match the actual target values.

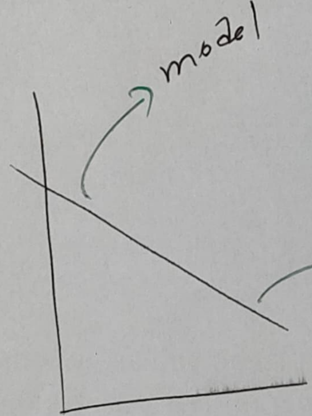
* Advantage & disadvantage of different loss function!

$\boxed{\text{slide - 4 - 10}}$ **

$\boxed{\text{Slide - 11}}$ ***

* keep focus on the error unit & data unit.

*



$\hat{y} = mx + c$

$\hat{y}_i = mx_i + c$

$L = (y_i - \hat{y}_i)^2$

$\Rightarrow \underbrace{L(m,c)}_{\text{learnable parameter}} = (y_i - mx_i - c)^2$

* For project, use:

  - ⇒ CNN
  - ⇒ GAN
  - ⇒ knowledge distillation

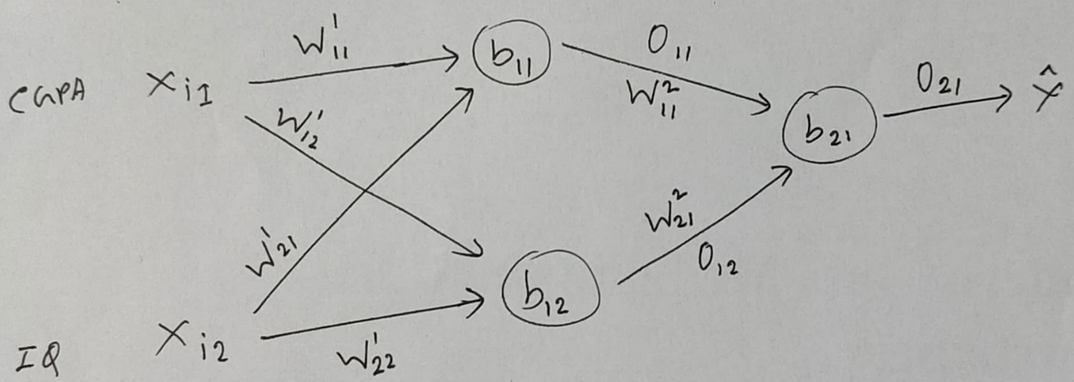
L-11 / 23.02.2025 /

Back-Propagation

CGPA	IQ	Salary	$\hat{Salary}$
3.75	80	8	18.96
3.00	90	8.3	5.2

18.96 ⇒ after first iteration, it overfit, change and test the next data

5.2 ⇒ underfit now, again change and test the next data.



⇒ activation function = Linear

$\hat{y} = O_{21} = O_{11} W_{11}^{2} + O_{12} W_{21}^{2} + b_{21}$

$L = (y - \hat{y})^2$

$W_{new} = W_{old} - \eta \frac{dL}{dW_{old}}$

$b_{new} = b_{old} - \eta \frac{dL}{db_{old}}$

Partial Derivative

$\therefore \frac{dL}{d\hat{y}} = \frac{d}{d\hat{y}} (y-\hat{y})^2 = -2(y-\hat{y})$

⇒ total 9 parameters find out partial derivative for all of them.

$O_{11} = x_{i1} W_{11}^{1} + x_{i2} W_{21}^{1} + b_{11}$

$O_{12} = x_{i1} W_{12}^{1} + x_{i2} W_{22}^{1} + b_{12}$

(i) $$\frac{dL}{dW_{11}^2} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dW_{11}^2} = \frac{dL}{dO_{21}} \cdot \frac{dO_{21}}{dW_{11}^2}$$

$$= -2(y-\hat{y}) \cdot O_{11}$$

(ii) $$\frac{dL}{dW_{21}^2} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dW_{21}^2} = -2(y-\hat{y}) \cdot O_{12}$$

(iii) $$\frac{dL}{db_{21}} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{db_{21}} = -2(y-\hat{y}) \cdot 1$$

(iv) $$\frac{dL}{db_{11}} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dO_{11}} \cdot \frac{dO_{11}}{db_{11}} = -2(y-\hat{y}) \cdot W_{11}^2 \cdot 1$$

(v) $$\frac{dL}{db_{12}} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dO_{12}} \cdot \frac{dO_{12}}{db_{12}} = -2(y-\hat{y}) \cdot W_{21}^2 \cdot 1$$

(vi) $$\frac{dL}{dW_{11}^1} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dO_{11}} \cdot \frac{dO_{11}}{dW_{11}^1} = -2(y-\hat{y}) \cdot W_{11}^2 \cdot x_{i1}$$

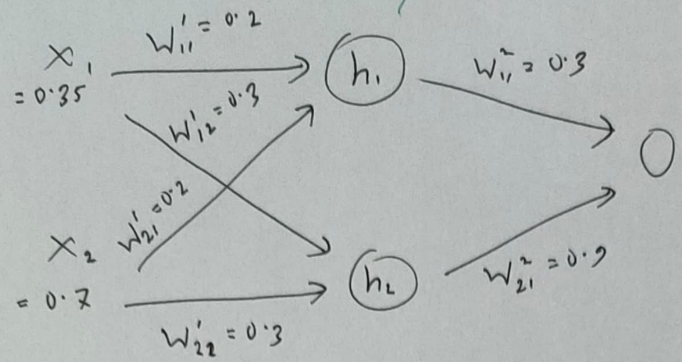
(vii) $$\frac{dL}{dW_{12}^1} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dO_{12}} \cdot \frac{dO_{12}}{dW_{12}^1} = -2(y-\hat{y}) \cdot W_{21}^2 \cdot x_{i1}$$

(viii) $$\frac{dL}{dW_{21}^1} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dO_{11}} \cdot \frac{dO_{11}}{dW_{21}^1} = -2(y-\hat{y}) \cdot W_{11}^2 \cdot x_{i2}$$

(ix) $$\frac{dL}{dW_{22}^1} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dO_{12}} \cdot \frac{dO_{12}}{dW_{22}^1} = -2(y-\hat{y}) \cdot W_{21}^2 \cdot x_{i2}$$

Exam question will be like this

Sigmoid function:



⇒ Forward Pass:

$$O_{11} = \sigma[0.35 \times 0.2 + 0.7 \times 0.2]$$

$$= \frac{1}{1 + e^{-0.21}} = 0.55$$

$$O_{21} = \sigma[0.35 \times 0.3 + 0.7 \times 0.3]$$

$$= \frac{1}{1 + e^{-0315}} = 0.578$$

L-12 / 25.02.2025 /

Online Session
28.02.2025
6:30 pm

Backpropagation:

initialize weights and bias

for i=1 to epochs:

for j=1 to row:

select one row (random)

apply forward pass

calculate loss

update weights and bias:

$$W_n = W_o - \eta \frac{dL}{dW_o}$$

Gradient Descent:

(i) Batch GD:

- calculate the total error then update the weights and bias
- also non as ~~vani~~ vanila GD.
- time consuming but accurate.

(ii) stochastic GD:

- weights and biases are updated based on the loss due to single data-point.
- faster but not so accurate.

(iii) Mini batch GD:

- weights and biases are updated based on the loss due to mini batches.

with two hidden layer diagram with two features:

$$W_{new} = W_{old} - \eta \frac{dL}{dW_{old}}$$

$$\therefore W^1_{11} = W^1_{11} - \eta \frac{dL}{dW^1_{11}}$$

$$\therefore \frac{dL}{dW^3_{11}} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dW^3_{11}}$$

$$\therefore \frac{dL}{dW^2_{11}} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dW^3_{11}} \cdot \frac{dW^3_{11}}{dW^2_{11}}$$

$$\frac{dL}{dW_{11}^1} = \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dW_{11}^3} \cdot \frac{dW_{11}^3}{dW_{11}^2} \cdot \frac{dW_{11}^2}{dW_{11}^1} +$$

$$\frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dW_{21}^3} \cdot \frac{dW_{21}^3}{dW_{12}^2} \cdot \frac{dW_{12}^2}{dW_{11}^1}$$

* Vanishing GD Problem:

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

$$\frac{d}{dz}\sigma(z) = \frac{d}{dz}\left(\frac{1}{1+e^{-z}}\right)$$

$$= \frac{(1+e^{-z})\frac{d}{dz}1 - 1\frac{d}{dz}(1+e^{-z})}{(1+e^{-z})^2}$$

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

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

$$= \sigma(z)(1-\sigma(z))$$

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

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

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

* ReLU $= \max(0, z)$

$$\frac{d}{dz} = (0, 1)$$

Some more,

⇒ Leaky ReLU (No zero gradient)

⇒ dying ReLU problem

⇒ exploring gradient descent

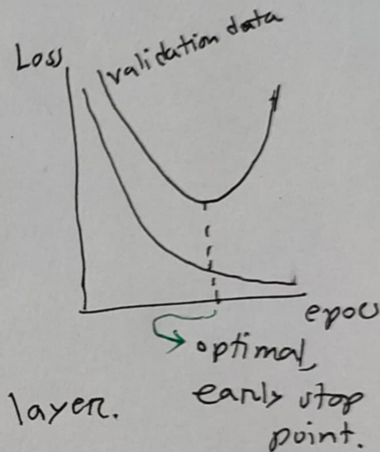
⇒ vanishing gradient descent problem.

L-13/02.03.2025 /

⊛ How to improve performance in a NN?

(i) Fine tuning the hyper-parameters in a NN.

- learning rate
- batch size ⇒ M.B.G.D. → small (8-32) ⇒ Generalize better; → large (8-10k)
- numbers of ~~e~~epochs ⇒ early stop
- number of hidden ~~la~~ layer
- activation function
- optimizer
- loss function
- number of neuron ~~per~~ per hidden layer.



(ii) By solving problem a NN may encountered.

- vanishing/exploding gradient descent
- not sufficient data ⇒ transparent data can be good choise.
- ~~sotw~~ slow training
- overfits

⊛ Problem with NN:

- vanishing/exploding gradient descent
- weights initialization
- activation function
- Batch normalization

- gradient clipping (exploding ~~gd~~ GD problem, solved)

- not sufficient data,
  - ⇒ data augmentation
  - ⇒ transparent learning

- slow training
  - optimizer
  - learning rate schedular

- Overfit:
  - L_1/L_2 regeneralizer

⊛ What is diverse data?

Midterm

07. 03. 2025

upto this