PAGE :
DATE : / /

Module 2

Q.10. In a perceptron learning algorithm, what is the initial value of the weights before the algorithm starts learning?

All the weights are assigned random values before the algorithm starts learning.

Q.11. What is the condition for convergence of a perceptron learning algorithm?

The data should be linearly seperable for convergence of a perceptron learning algorithm

Q.12. Consider you are given a Boolean function with 5 inputs. It is represented by a network of perceptrons containing one hidden layer and one output layer with one perceptron. How many perceptrons are there in the hidden layer?

Number of input $(n) = 5$

No. of perceptrons in hidden layer $= 2^n = 2^5 = 32$

Q.19. Comment on the update formula at the ith update in the Momentum-Based Gradient Descent.

Momentum-based Gradient descent, where 'w' and 'b' are updated not just based on the current updates (derivative) but also the past updates (derivatives).

$V_{dW} = \beta V_{dW} + (1-\beta) dW$, where V_{dW} is momentum

$V_{db} = \beta V_{db} + (1-\beta) db$

$W = W - \alpha V_{dW}$; $b = b - \alpha V_{db}$

Q. 21. Assume you have 1,50,000 data points, mini-batch size being 25000, one epoch implies one pass over the data, and one step means one update of the parameters. What is the number of steps in one epoch for mini-batch gradient descent.

Mini-batch size $(k) = 25000$

No. of data points $(n) = 1{,}50{,}000$

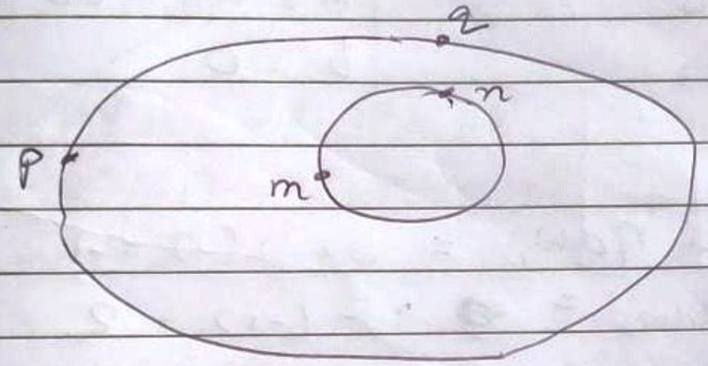
$\therefore$ No. of mini-batch $= \frac{150000}{25000} = 6$

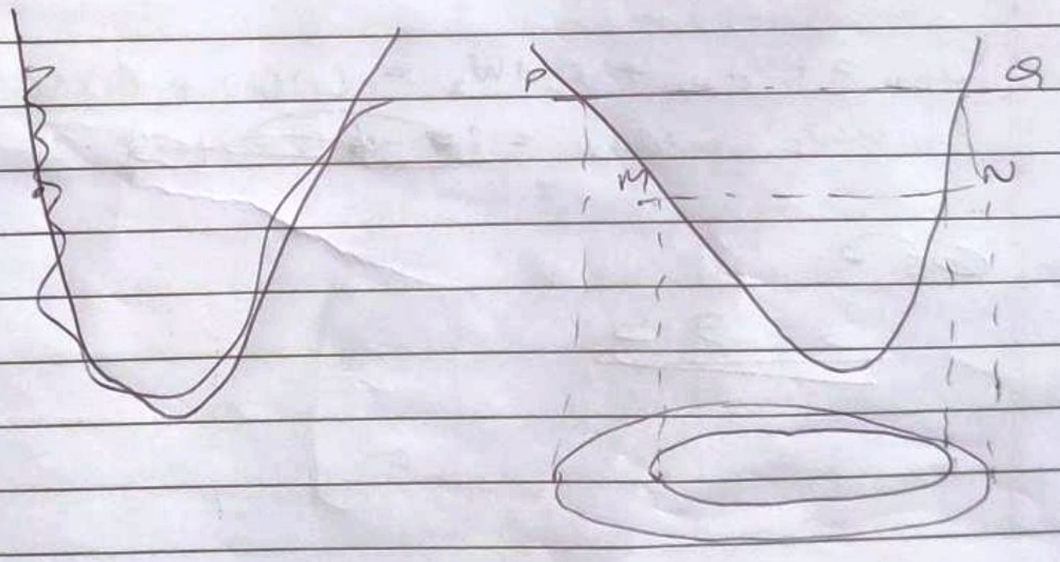
$\therefore$ No. of steps in one epoch $= 6$

Q.22 How do you reduce the oscillations and improve the stochastic estimates of the gradient that is estimated from one data point at a time?

By using mini-batch gradient descent, ~~it helps red~~

Q.20. Error at point p = Error at point q = 0.49





Q.26. Purpose of data augmentation

① More data = Better learning
② Works well for image recognition / object recognition taks
③ It also works well for speech
④

Q.29. Valid ensemble method?

Ensemble methods combine the output of different models to reduce generalization error.
Bagging and boosting are valid forms of ensemble methods

Q.42. Which one of the following happens when learning rate is too large for back-propagation?

A large value of learning rate may speed up the convergence but might result in overshooting and overstep the optimal value. It leads to rapid learning but there is oscillation of weights.

Q.43 What happens to the gradient when the steepness of error surface increases?

When the curve is steep, the gradient becomes large.

Q-37. Significance of inclusion of learning parameter (η) during gradient descent based parameter update.

The learning rate (η) is used to control the amount of weight adjustment at each step of training. The learning rate, ranging from 0 to 1, determines the rate of learning at each time step.
This parameter determines how fast or slow we will move towards the optimal weights.
If the learning rate is very large, we will skip the optimal solution. If it is too small, we will need too many iterations to converge to the best values. So using a good learning rate is crucial.

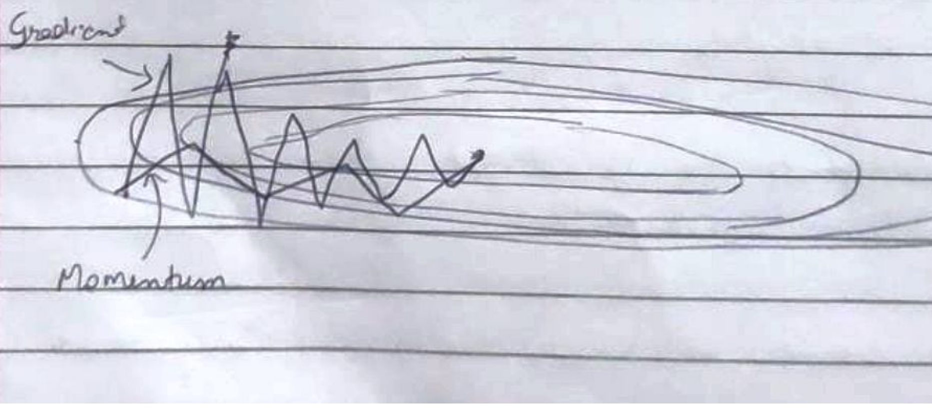
Q.44. Very small distance between the contours sliced at two different vertical positions on the error surface (RMSE) indicates which of the following:

A very small distance between the contours sliced at two different vertical position indicates a steep sloope along that direction

Q.45 To overcome the slow convergence of gradient descent, momentum based gradient descent uses which factor?

In momentum based gradient descent, 'w' and 'b' are not updated based on the current updates (derivative), but ~~st~~ also the past updates (derivative).

$$V_{dW} = \beta V_{dW} + (1-\beta) dW$$
$$V_{db} = \beta V_{db} + (1-\beta) db$$
$$W = W - \alpha V_{dW} \; ; \; b = b - \alpha V_{db}$$



PAGE :
DATE : / /

Q.46. In Nesterov Accelerated Gradient descent, the oscillations are always less as compared to momentum based gradient descent?

It moves the history component first, then calculates the derivative and updates later.
Looking ahead helps NAG in correcting its course quicker than momentum based gradient descent.
Hence, the oscillations are smaller and chances of escaping the minima valley also smaller.

Q.53

Iteration	ω	$-\nabla_\omega E$
0	1	1.0
1	2	0.5
2	?	0.25

$\beta = 0$, $V_{dw_0} = 0.6$

$V_{dw_t} = \beta V_{dw} + (1-\beta) dW_t$

Here, $1-\beta = \eta$

$\therefore V_w = \beta V_w + \eta dW$

$w = w - V_w$

Iteration 1:-

$\beta = 0.5$, $\eta = 1$, $V_\omega = 0$, $-\nabla = -dW = 1$, $\omega = 1$

$V_\omega = (0.5 \times 0) + 1 \cdot (-1)$

$V_\omega = -1$

$\omega = \omega - V_\omega = 1 - (-1) = 2$

$\underline{\omega = 2}$

Iteration 2:

$\beta = 0.5$, $\eta = 1$, $V_\omega = -1$, $-dW = 0.5$, $\omega = 2$

$V_\omega = (0.5 \times -1) + (1 \times -0.5)$

$V_\omega = -1$

$\omega = \omega - V_\omega = 2 - (-1) = 3$

$\underline{\underline{\omega = 3}}$

Q. 98 Implement perceptron training rule for a network with the following data

$(x_1 = [2 \quad 1 \quad -1],\ d_1 = -1)$;

$(x_2 = [0 \quad -1 \quad -1],\ d_2 = 1)$.

Initial weights $w_1 = [0 \quad 1 \quad 0]$

Repeat the training sequence $(x_1, d_1), (x_2, d_2)$ until two correct responses in a row are achieved.

Assume $c = 1$ and $f(net) = sgn(net)$.

Epoch 1:

Consider $X = x_1$

Step 1:

$$net = \sum w^T x$$

$$= \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} [0 \quad 1 \quad 0] \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}$$

$$= \{0 + 1 + 0\}$$

$$net = 1$$

$y_1 = f(net) = 1$

$y_1 \neq d_1$

$$\Delta w = c(d_1 - y_1)x_1$$

$$= 1(-1 - (+1))\, x_1$$

$$= -2 \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -4 \\ -2 \\ 2 \end{bmatrix}$$

$w_{new} = w_{old} + \Delta w$

$w_2 = w_1 + \Delta w$

$$= \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -4 \\ -2 \\ 2 \end{bmatrix} = \begin{bmatrix} -4 \\ -1 \\ 2 \end{bmatrix}$$

Consider $x = x_2$

$$net = \sum w^T x$$

$$= \begin{bmatrix} -4 & -1 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 + 1 - 2 \end{bmatrix}$$

$$net = -1$$

$$y_2 = sign(net) = -1$$

$$y_2 \neq d_2$$

$$\Delta w = c(d_2 - y_2)\, x_2$$

$$= 1(2) \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$$

$$\Delta w = \begin{bmatrix} 0 \\ -2 \\ -2 \end{bmatrix}$$

$$w_{new} = w_{old} + \Delta w$$

$$w_3 = w_2 + \Delta w$$

$$= \begin{bmatrix} -4 \\ -1 \\ 2 \end{bmatrix} + \begin{bmatrix} 0 \\ -2 \\ -2 \end{bmatrix}$$

$$w_3 = \begin{bmatrix} -4 \\ -3 \\ 0 \end{bmatrix}$$

Epoch 1 Complete

Epoch 3 :

Consider $x = x_1$

$$net = \sum w^T x$$

$$= \begin{bmatrix} -4 & -3 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}$$

$$= [-8 \; -3 + 0]$$

$$net = -11$$

$$y_1 = f(net) = -1$$

$$y_1 = d_1$$

$$\Delta w = c(d_1 - y_1)\, x_1$$

$$= 1\,(-1-(-1))\, x_1$$

$$\Delta w = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\cancel{w_{new}} \; w_4 = w_3 + \Delta w$$

$$w_4 = w_3$$

$$= \begin{bmatrix} -4 \\ -3 \\ 0 \end{bmatrix}$$

Consider $x = x_2$

$$net = \sum w^T x$$

$$= \begin{bmatrix} -4 & -3 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$$

$$= [0 + 3 + 0]$$

$$net = 3$$

$$y_2 = f(net) = 1$$

$$y_2 = d_2$$

$$\Delta w = c(d_1 - y_1)x_1$$
$$= \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$w_5 = w_4$

$\underline{y_1 = d_1 \ \& \ y_2 = d_2}$

Epoch 2 Complete

Epoch 3

Consider $X = X_1$

$$net = \sum w^T x$$
$$= \begin{bmatrix} -4 & -3 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}$$
$$= [-8 - 3 + 0]$$

$net = -11$

$y_1 = f(net) = -1$

$y_1 = d_1$

$$\Delta w = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$w_6 = w_5$

Consider $X = X_2$

$$net = \sum w^T x$$
$$= \begin{bmatrix} -4 & -3 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$$

$net = [3]$

$y_2 = f(net) = 1 \quad \underline{y_2 = d_2}$

$$\Delta w = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$w_6 = w_5$

$\underline{y_1 = d_1 \ \& \ y_2 = d_2}$

Epoch 3 Complete

Q.59. Comment on the following:
For a binary bipolar neuron, the weight change formula for a perceptron (Adaline) training rule reduces to $\Delta w = \pm 2ex$

~~Explain~~ For a binary bipolar neuron,

y	$\hat{y}$	$(y-\hat{y})$
-1	1	-2
-1	-1	0
1	1	0
1	-1	2

For correct prediction, $(y-\hat{y}) = 0$
When $y = -1$, $\hat{y} = 1$,
change in weight in Adaline will be,

$$\Delta w = c \cdot (y - \hat{y})\, x$$
$$= c(-2)x$$
$$= -2cx$$

When $y = 1$, $\hat{y} = -1$, change in weight will be,

$$\Delta w = c \cdot (y-\hat{y}) \cdot x$$
$$= c \cdot (2)\, x$$
$$= 2cx$$

Weight updation formula reduces to
$$\Delta w = \pm 2cx$$

DATE: / /

Q60. Suppose there are 20 nodes in a Deep Neural Network and we implement Dropout by removing few nodes to obtain a thinned network. What is total number of such thinned networks that can be found?

Total number of thinned networks = 2^n

Here, $n = 20 \rightarrow 2^{20}$

Q.61. You are training a neural network model using Early Stopping Technique. Given that the patience parameter is 2, when will you stop training.

Epochs	Training Loss	Validation Loss
1	3.4	2.0
2	2.0	1.9
3	1.9	1.8
4	1.8	1.8
5	1.7	1.9
6	1.6	2.1

$\because p = 2$

We'll stop after epoch 4 since there is no updation in ~~previous~~ the validation loss in the ~~per~~ previous 2 steps.