ABHISHEK SHARMA.
CS THIRD YEAR.
SECTION - 'I'
ROLL NO.: 01
ENROLLMENT NO.: 12019009001127.
Subject : Deep Learning
Assignment No. : PEC501/01.
Date :- 16.08.2021.

Q1. What is exploding gradient problem ? Mention some solutions to solve this problem ?

(A) In machine Learning, the exploding gradient problem is an issue found in training artificial neural network with gradient based learning methods and backpropagation. An ANN is a learning algorithm, also called neural network or neural net. that uses a network functions to understand and translate data input to a specific output. This type of learning algorithms are decided to mimic the way neurons functions in the human brain. Exploding gradient problems are a problem which occur when large error gradients accumulate and result in very large updates to neural networks. model weights during training.

Solutions to solve the exploding gradient problem —
(i) Re - Design the network model.
(ii) use Long short term memory (LSTM) networks.
(iii) use gradient clipping.
(iv) use weight regularization.

Q2. How does the initialization of weight affect the performance of the ANN?

(A) There are different types of weight initialization schemes. Among them these are the following —

(i) weight initialized to all zeros.
(ii) weight initialized to all ones.
(iii) weights initialized with values sampled from a uniform distribution with a fixed bound.
(iv) weight initialized with values sampled from a uniform distribution with a careful tweak.
(v) weight initialized with values sampled from a normal distribution with a careful tweak.

Now let's deploy these weights on a fashion MNIST dataset -

(a) If we initialize to all the weights to zeros then we can see the difference b/w the validation loss and training loss. Our model shows the efficiency and is really struggling to train, also it should not be case ideally for a given architecture because of all the values starting from zero. Hence, the weight updates that happening because of propagation is not effective enough for the model to cut-through.

(b) Studies have shown having the initializing the weights with values sampled from a random distribution instead of constant values like zeros and ones actually helps a neural net to train their model better and faster.

(c) Ideally we would want the values of the weight vector to be finished in such a way that they do not end up causing a data loss in input vector. Ultimately we are multiplying the weight vector with the input vector so we need to be very careful.

(d) This type of weight initialization leaves us to our final experiment where we would sample values from a normal distribution with its standard deviation set to y.

(e) This weight distribution is as same as the earlier one provides the standard deviation of set to y.

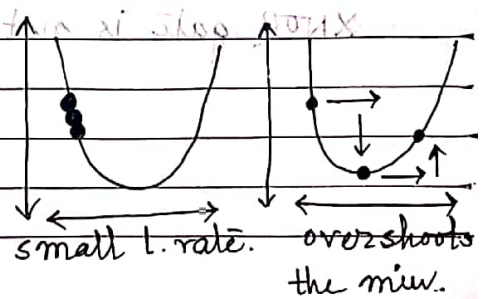
We can clearly notice that, when our network is initialized with the contained uniform distribution the dispersion in the weight distribution is less and most of the values are closer to zero, which we wanted.

Q3. In optimization techniques, what is the significance of the learning rate?

(A) Learning rate parameter denoted by α (alpha) is used to tune how accurately a model converges on a result. This can be thought of as a ball thrown down a staircase And a higher learning rate value is equivalent to the higher speed of the descending ball. This ball will leap skipping adjacent steps and reaching the bottom quickly but not settling because of the momentum it carries.

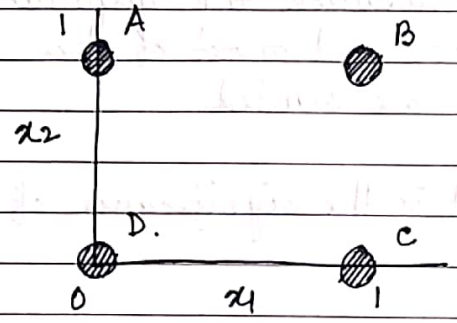
Learning rate is a scalar, a value which tells the machine how fast or slow to arrive any solution. The speed at which a model learns a ~~app~~ important and it varries with different applications. A super fast learning algorithm can miss a few data points or correlations which can give better insights on the data, missing this will eventually lead to wrong classifications.

If a learning rate is too small then, learning will take too long. And if learning will take too long, then next point will appear across the bottom of the valley.



Q4. Consider the XNOR operation. Assuming that it is an classification problem with the output being in both the classes as 0 and 1. Check if it linearly separable?

(A) According the Lemma stated that, if 3 points are collinear and the middle point has a different label than the other two, then these 3 points can be not a linearly separable points.



Consider these 4 points that present in a XNOR table. Let us label them clock wise, so the top left of the graph as A, top right as B. bottom left as D and the rest one is C.

Here, A and C has the same label and B and D are having the same label. We want to show that A, B, C and D are not linearly separable. Let's imagine the fifth point to be labelled as in the center and name as E. Since, E lies in the center, three points A, B, C are collinear and also B, E, D are collinear. Because, we assume, a line A, B, C, D and the line must have the label of E of being linearly separable. If E shares the same lable with A and C then it will be sharing different lable with B and D, which provides the contradiction towards the lemma of the given labels.

Therefore, it is impossible to give a label to E while satisfying the linearly separability. As a result, the four points A, B, C and D can not be linearly separable.

XNOR gate is not linearly separable [proved].

Q5. The Harley Davidson Iron 883 has the following normalized features : (Displacement, Mileage, Kerb wt., IsRed available) = (8.83, 0.20, 2.56, 1) Now consider a person who wants to decide whether to buy Harley Davidson bike. He assigns the following inputs; W = [0.9, 0.4, 0.7, 1] Further suppose that, θ = 8. Based on the above information do you think he will buy the bike based on a simple McCulloch Pitts Neuron.

Here the normalized features are, [8.83, 0.20, 2.56, 1] where only the last feature, IsRedavailable is inhibitory and is binary in nature {0, 1}.

In the weights, there is also the same parameter with 1 value

But, the θ value is 8. Hence, the buyer can not be able to buy the bike based on the McCulloch Pitts neuron as $g(x) < \theta$ here, $1 < 8$.

Q6. Starting from inception to back propagation logically establish the evaluation of artificial neuron. Always specify the reason and solution at every step.

(A) The steps that are going to be followed from inception to back propagation in a logical way —

(i) initialize the weights of the neural network.
(ii) propagate inputs forward through the network to generate the input values.
(iii) Calculate the error.
(iv) Propagation of the output back through the network, in order to generate the error of all output and hidden neurons.

When we are propagation the error back to the neural network and when we are calculating the errors, there is a difference depending on where the neuron is located in the network. Meaning we use different equations to calculate the error of the neuron in the output layer of the equations, that we are trying to calculate the error of the neurons in the hidden layers.

$$\frac{\partial C}{\partial w^{ij}} = \frac{\partial C}{\partial o^j} \frac{\partial o^j}{\partial net^j} \frac{\partial net^j}{\partial w^{ij}}$$

$$\delta = \frac{\partial C}{\partial o^j} \frac{o^j}{net^j}$$

Now by solving each part of this equation we will get the final value of the derivative of the cost function with respect to the output to the neuron, is pretty straightforward to calculate. Since, the output value of the output neuron is the output of the a, and can be-

$$\frac{\partial C}{\partial o^j} = \frac{\partial C}{\partial a} = \frac{\partial}{\partial a} \frac{1}{2} (y(x) - a) = a - y(x).$$

$$\gamma(x) = \frac{1}{1 + e^{-x}}.$$

Then, we can write that.

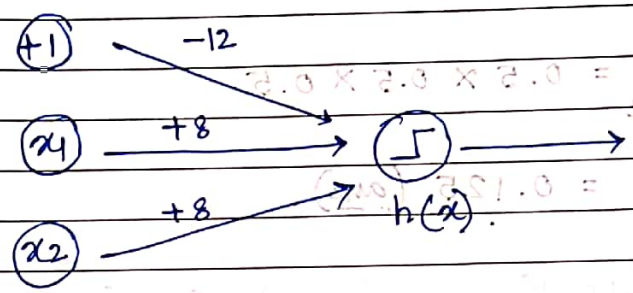
$$\frac{\partial o^j}{\partial net^j} = \frac{\partial}{\partial net^j} \gamma(net^j) = \gamma(net^j)(1 - \gamma(net^j)).$$

Now, when the error is available, we can finally adjust our weight of connection. However, before, we do that, we need to pick the learning rate - η. This learning rate is quite important since it is dictating next step in the gradient descent process and it is not properly adjusted, this may cause the minimum to be missed or learning process to be very slow.

$$\Delta w^{ij} = -\eta \frac{\partial C}{\partial w^{ij}} = -\eta y^i \delta^j$$

given

Q7. (i) You are the following neural networks which are taking 2 binary valued inputs $x_1\ x_2 \in \{0, 1\}$ and the activation function is the threshold function ($h(x) = 1$ if $x > 0$; 0 otherwise). Which of the following logical function does it compute?



By constructing the truth table, we can get the logical function that is being followed here —

x_1	x_2	0
0	0	0
0	1	0
1	0	0
1	1	1

From the truth table we can conclude that 'AND' function is operating here.

(ii) We have a function which takes two dimensional input $x = (x_1, x_2)$ and has two parameters, $\omega = (\omega_1, \omega_2)$ given by $f(x, \omega) = \sigma(\sigma(x_1 \omega_1)\omega_2 + x_2)$ where $\sigma(x) = \frac{1}{1+e^{-x}}$. We use backpropagation to estimate the right parameters values. We start by setting both the parameters to 0. Assume that we are given a training point $x_1 = 1$, $x_2 = 0$ and $y = 5$. Given the information,

What is the value of $\frac{\partial f}{\partial \omega_2}$?

(A) $\sigma(x_1 w_1) w_2 + x_2$ as o_2 and $x_1 w_1$ as o_1

$$\frac{\partial f}{\partial w_2} = \frac{\partial f}{\partial o_2} \frac{\partial o_2}{\partial w_2}.$$

$$\frac{\partial f}{\partial w_2} = \sigma(o_2)(1 - \sigma(o_2)) \times \sigma(o_1)$$

$$\frac{\partial f}{\partial w_2} = 0.5 \times 0.5 \times 0.5$$

$$= 0.125 \text{ (ans)}.$$

Q8. Discuss about the effectiveness of vectorization?

(A) Just like in the real world we are interested in solving any kind of problem efficiently in such a way that the amount of error is reduced as much as possible.

In ML, there is an concept of optimization algorithm that tries to reduce the error and computes to get the best parameters of the ML model.

So, by using vectorized implementation in an optimization algorithm we can make the process of computation much faster compared to the unvectorized implementation

- Advantages of vectorization :-

(a) Our code runs efficiently.
(b) Our code becomes simpler and easy to debug.

Q9. Discuss about how about perception learning algorithm confirms the convergence?

(A) According to perception, when x belongs to p, then $w.x = 0$. or, when x belongs to p, the angle b/w w and x should be less than 90°. Because the cosine angle is proportional to the dot product.

$$\cos\alpha = \frac{W^T x}{\|W\|\|x\|}$$

and, $\cos\alpha = W^T x$.

so, if $W^T x > 0 \Rightarrow \cos\alpha > 0 \Rightarrow \alpha < 90$.

and, similarly, $W^T x < 0 \Rightarrow \cos\alpha < 0 \Rightarrow \alpha < 90°$.

Also, according to the rule of convergence, W vector must be an angle less than 90° with positive examples data vectors ($x \in P$) and an angle more than 90° with the negative examples data ($x \in N$).

The perception learning algorithm was among the earliest demonstrations of the learnability of the concepts of data. The algorithm makes rather strong assumption linear separability of the data, which seldom actively encountered. However nothing steps us from the application of applying Perception algorithm in practice for the hope of achieving good, if not perfect result. Indeed there first refinements of Perception algorithm such that even when the input points are linearly separable, the algorithm converges to a configuration that minimises the number of unclassified points.

Q10 Consider the four data points with 3D that are divided into two classes class 1 and 2. Let the initial weight vector be [-0.5, 1, 0.2]. Apply the simple trial and error based perception learning algorithm to the sample data and order. Find the values of the weight vector after it converges. If $x \in 1$, do $w + x$ otherwise $w - x$ and do untill the system converges.

(A) Let's take the initial values of weight vector be

$$W = \begin{pmatrix} -0.5 \\ 1 \\ 0.2 \end{pmatrix} \quad \text{and, } \eta = 0.2$$

$0 = \omega_0 + \omega_1 x_1 + \omega_2 x_2$.

$\Rightarrow 0 = -0.5 + x_1 + 0.2 x_2$.

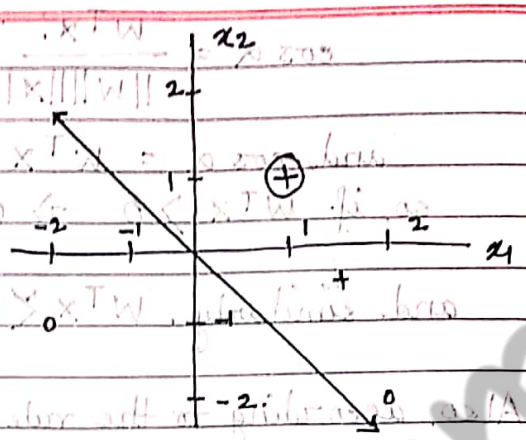
$\Rightarrow 0 = x_1 + 0.2 x_2$.

$\Rightarrow -0.2 x_2 = x_1$

$\Rightarrow -\frac{2}{5/10} x_2 = x_1$

$\Rightarrow -x_2 = 5 x_1$

$\Rightarrow x_2 = -5 x_1$

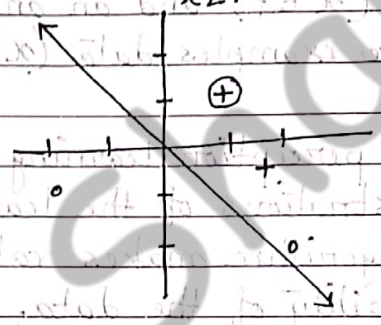


Now, $x_1 = 1$, $x_2 = 1$.

$W^T x > 0$.

correct classification.

No Action.



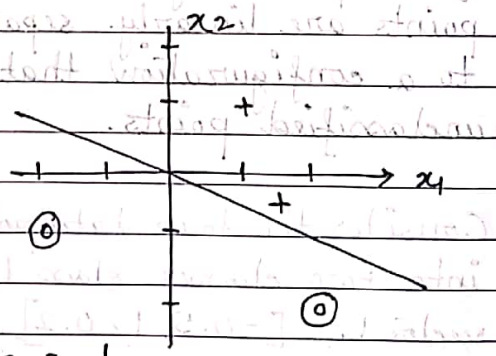
Now, $x_1 = 2$ and $x_2 = -2$.

$\omega_0 = \omega_0 - 0.2 \times 1 = -0.5 - 0.2 = -0.7$

$\omega_1 = \omega_1 - 0.2 \times 2 = 1 - 0.4 = 0.6$.

$\omega_2 = \omega_2 - 0.2 \times (-2) = 0.2 + 0.4 = 0.6$.

$\eta = 0.2$ and. $\omega = \begin{pmatrix} -0.7 \\ 0.6 \\ 0.6 \end{pmatrix}$



$\eta = 0.2$ and.

$\omega = \begin{pmatrix} -0.7 \\ 0.6 \\ 0.6 \end{pmatrix}$ $x_1 = -2$ and $x_2 = -1$

$W^T x < 0$.

correct classifier, No action.

Now, $x_1 = 1.5$ and $x_2 = 0.5$.

$W^T x < 0$.

correct classifier, No action.

Hence, the final weight vector will be.

$\omega = \begin{pmatrix} -0.7 \\ 0.6 \\ 0.6 \end{pmatrix}$ (ans).

A

—— X ——