Q.0

1. Explain Why we need Activation functions. Kishor Kumar Andekar

Activation Functions

→ They define the output of that node given an input or set of inputs

→ They decide whether a neuron should be activated or not.

→ Activation function can be divided into 2 types

1. Linear Activation function
2. Non Linear Activation function
   - → Sigmoid
   - → Tanh
   - → ReLu

★ Sigmoid: It exists b/w 0 to 1

→ Used for models where we have to predict the probability as an output, sigmoid is right choice

→ function is differentiable

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

★ Tanh: range is between -1 to 1, Tanh is also sigmoidal.

→ It is differentiable

→ tanh function is mainly used Classification between two classes

★ ReLU (Rectified Linear Unit)

→ ReLU is half rectified (from bottom). Range is 0 to ∞

$$R(z) = \max(0, z)$$

Activation function play a critical role in enabling neural networks to model and solve a wide range of real world problems.

2. When implementing a neural N/W Layer from scratch. we usually implement a forward and a backward function for each Layer Explain what these functions do potential variables that they need to save, which argument they take and what they return

Forward function:

Purpose: Computes the output of the Layer from given Input

Argument: Takes Input, weight and biases

Variables to save: May store weighted sum, pre activation and activated output

Return: Provides the activated output.

Backward function:

purpose: Calculates gradients for optimization during backpropagation

Arguments: Receives gradient of loss with respect to output, Input and weights

Variables to save: stores gradient of Loss with respect to pre activation

Return: offers gradients for Inputs, weight, biases for parameter updates.

3. Given a Convolution Layer with 8 filters, a filter size of 6, a stride of 2, and a padding of 1. for an input feature map of 32x32x32 what is the output dimensionality after applying the Convolution Layer to the input ?

Output Dimension = [(Input Dimension - filter size + 2x padding)/stride] +1

Given

Input Dimension : 32x32x32

Filter size : 6

Stride : 2

Padding : 1

Output Dimension = [(32 - 6 + 2x1)/2] + 1

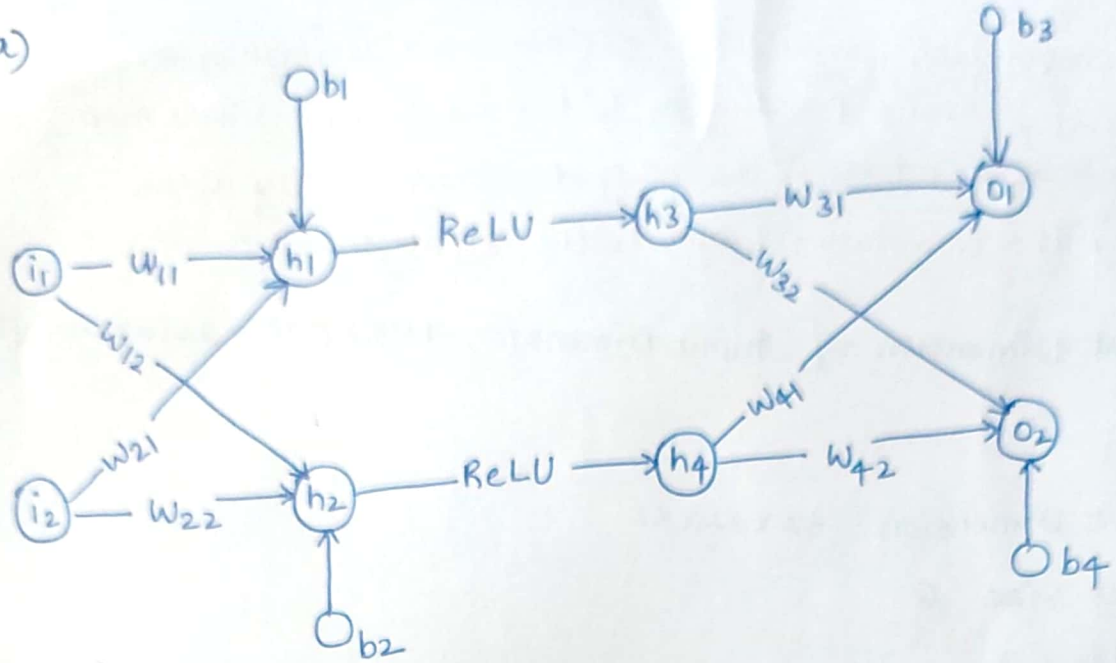
= (28/2) + 1

= 14 + 1

= 15

So, Output feature Map after applying the convolutional Layer would be 15 x 15 x 8.

4. a)



Given values

variable	i_1	i_2	w_{11}	w_{12}	w_{21}	w_{22}	w_{31}	w_{32}	w_{41}	w_{42}
Value	2.0	-1.0	1.0	-0.5	0.5	-1.0	0.5	-1.0	-0.5	1.0

variable	b_1	b_2	b_3	b_4	t_1	t_2
Value	0.5	-0.5	-1.0	0.5	1.0	0.5

Calculating weighted sum and ReLU activation for the hidden units

For h1: $h_{in1} = (i_1 \times w_{11}) + (i_2 * w_{21}) + b_1$

$= (2.0 \times 1.0) + (-1.0 * 0.5) + 0.5$

$= 2.0 - 0.5 + 0.5$

$= 2.0$

~~$h_1 = \text{ReLU}(z_1)$~~

~~$= \max(0, z_1)$~~

~~$= \max(0, 2.0)$~~

$h_1 = 2.0$

For h_2:

$$h_2 = (i_1 * w_{12}) + (i_2 * w_{22}) + b_2$$
$$= (2.0 * -0.5) + (-1.0 * -1.0) + (-0.5)$$
$$= -1.0 + 1.0 - 0.5$$
$$= -0.5$$

$h_2 = \text{ReLU}(z_2) = \max(0, z_2) = \max(0, -0.5)$

$h_2 = 0$

for h_3:

$$h_3 = \text{ReLU}(h_1)$$
$$= \max(0, 2.0)$$
$$h_3 = 2.0$$

For h_4:

$$h_4 = \text{ReLU}(h_2) = \max(0, -0.5) = 0$$
$$h_4 = 0$$

for $o_1 = (h_3 * w_{31}) + (h_4 * w_{41}) + b_3$
$$= (2.0 * 0.5) + (0 * -0.5) + (-1.0)$$
$$= (1.0) - 1.0 = 0$$

for $o_2 = (h_4 * w_{42}) + (h_3 * w_{32}) + b_4$
$$= (0 * 1.0) + (2.0 * (-1.0)) + 0.5$$
$$= -2.0 + 0.5 = -1.5$$

final output $(o_1, o_2) = (0, -1.5)$

b) Compute mean squared error of o/p (O_1, O_2) calculate above and target (t_1, t_2)

$O_1 = 0$, $O_2 = -1.5$, $t_1 = 1.0$, $t_2 = 0.5$

$$MSE\_O_1 = \left(\frac{1}{2}\right) \times [(O_1 - t_1)^2]$$

$$= \frac{1}{2}[(0 - 1.0)^2]$$

$$= 1/2 = 0.5$$

$$MSE\_O_2 = \left(\frac{1}{2}\right) \times [(O_2 - t_2)^2]$$

$$= \frac{1}{2} \times [(-1.5 - 0.5)^2] = \frac{1}{2}(2)^2 = \frac{4}{2} = 2$$

Total Mean square error $TMSE = (MSE\_O_1 + MSE\_O_2)/2$

$$= \frac{(0.5 + 2)}{2}$$

$$= \frac{2.5}{2}$$

$$TMSE = 1.25$$

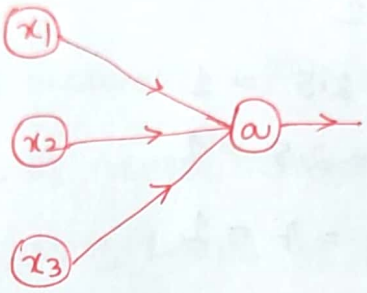
Mean squared error of o/p (O_1, O_2) with respect to target (t_1, t_2) is approx 1.25

5. Let us assume we implement an AND function to a single neuron.

Below is a tabular representation of an AND function

X_1	X_2	X_1 AND X_2
0	0	0
0	1	0
1	0	0
1	1	1

The activation function of our neuron is denoted as



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

for which values of w1, w2 and b does our neuron implement an AND function?

Bias = -1.5, $w_1 = 1$, $w_2 = 1$

Bias = 1.5, $w_1 = 2$, $w_2 = 2$

Bias = 1, $w_1 = 1.5$, $w_2 = 1.5$

→ X_1 and X_2 represents binary inputs to AND functions. Output represents output of the AND function for given input.

→ Both inputs are 0, O/p is 0

→ All other combinations of I/p the output is 1, indicating that neuron fires (or activates) only when both inputs are 1.

first Evaluate Each set of weight and bias

$$X_1 * w_1 + X_2 * w_2 + \text{Bias} = \text{Output}$$

→ $0 \times 1 + 0 * 1 - 1.5 = -1.5 = 0$

→ $0 \times 1 + 1 \times 1 - 1.5 = -0.5 = 0$

→ $1 \times 0 + 0 \times 1 - 1.5 = -0.5 = 0$

→ $1 \times 1 + 1 \times 1 - 1.5 = 0.5 = 1$ (successfully Implement AND function)

Second set

$0 \times 2 + 0 \times 2 + 1.5 = 1.5 = 1$

$0 \times 2 + 1 \times 2 + 1.5 = 3.5 = 1$

$1 \times 2 + 0 \times 2 + 1.5 = 3.5 = 1$

$1 \times 2 + 1 \times 2 + 1.5 = 5.5 = 1$

all values successfully implement AND functions

Third set

$0 \times 1.5 + 0 \times 1.5 + 1 = 1$

$0 \times 1.5 + 1 \times 1.5 + 1 = 2.5 = 1$

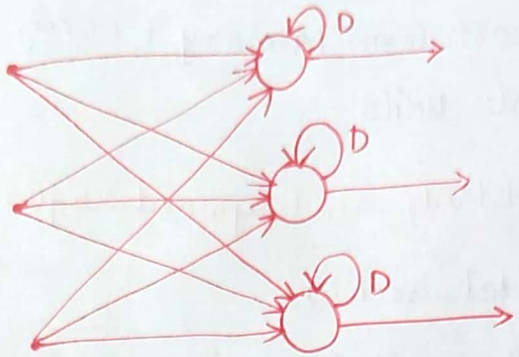
$1 \times 1.5 + 0 \times 1.5 + 1 = 2.5 = 1$

$1 \times 1.5 + 1 \times 1.5 + 1 = 4 = 1$

All values successfully implement AND functions.

Hence, All three sets of bias and weights successfully implement the AND function.

6. You are building a neural network where it gets input from the previous layer as well as from itself. which architecture has feedback connections ? Explain



Recurrent Neural Network (RNN)

An RNN is a type of neural network architecture that introduces the concept of Loops or feedback connections in the Network.

→ It allows information to be passed from one step of the network to the next, enabling it to maintain a form of memory or context.

→ In RNN, each Layer receives input not only from the previous layer output but also from its own o/p at previous timestamp.

→ This Looping Mechanism enables RNNs to process Sequence of data, making them suitable for tasks involving sequential data like time series, natural Language and Speech

→ In RNN, the current i/p and o/p from previous timestamp are combined to produce current o/p and this process Continues iteratively as the n/w processes each step of sequences.

However, Traditional RNN have limitations in capturing long range dependencies due to vanishing gradient problem.

To Address Limitations, variations of RNN have been developed such as Long short Term Memory (LSTM) networks and Gated Recurrent units

7. what comes in blank as Keyword argument ?

```
model = model_arch()
model.compile (optimizer = Adam (learning_rate = 1e-3),
               loss = 'sparse_categorical_crossentrophy)
               metrics = ['sparse_categorical_accuracy'])
model.summary().
```

8. AutoEncoders are incapable of Learning nonLinear manifolds ?

False

9. Sparse autoencoders introduces information bottleneck by reducing the number of nodes at hidden Layers

True

10. Recurrent Neural Networks Architecture is more effective for time series data ?

True