**WORKSHEET-2**

**DEEP LEARNING (SOLUTIONS)**

**ANS 1)** C) serially or parallel

**ANS 2)** A) Rosenblatt, 1958

**ANS 3)** C)

**ANS 4)** B) [8×3], [5×8]

**ANS 5)** A) A unit which does not respond completely to any of the training patterns

**ANS 6)** B) softmax

**ANS 7)** D) weights

**ANS 8)** B) output units are updated sequentially

**ANS 9)** A) Early Stopping, B) Dropout, D) ReduceLROnPlateau

**ANS 10)**

B) It is strictly more powerful than a Convolutional Neural Network

C) It is applicable when the input/output is a sequence (e.g., a sequence of words)

* **ANS 11)** A neural network with a linear activation function reduces to a linear equation. The neural network with **linear activation functions** fails the purpose of creating a very complex function that can fit on any sort of data and as it can be clearly seen.
* **Hence,** if we want to build a perceptron to give outputs of a linear regression problem **it should be strictly noted that a linear function cannot be used as an activation function for the neural network,** although **it can be used only in the last layer for regression problem*.***

**ANS 12)**

* The learning rate is a hyper-parameter that controls how much to change the model in response to the estimated error each time the model weights are updated. The learning rate controls how quickly the model is adapted to the problem.
* Specifically, the learning rate is a configurable hyper-parameter used in the training of neural networks that has a small positive value, **often in the range between 0.0 and 1.0.**
* **Smaller learning rates** require more training epochs given the smaller changes made to the weights each update, learning rate that is too small can cause the process to get stuck.
* **Larger learning rates** large can cause the model to converge too quickly to a suboptimal solution, larger learning rates result in rapid changes and require fewer training epochs.

**ANS 13)**

For three inputs the number of combinations of 0 and 1 is 8:

x1: 0 1 0 1 0 1 0 1

x2: 0 0 1 1 0 0 1 1

x3: 0 0 0 0 1 1 1 1

and for four inputs the number of combinations is 16:

x1: 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

x2: 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1

x3: 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1

x4: 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1

We may check that for five inputs the number of combinations will be 32.

**Note**: for **3 nodes**: 8 = 2^ 3,

for **4 nodes**: 16 = 2^4 and

for **5 nodes**: 32 = 2^ 5.

Thus, the **formula** for the number of binary input patterns is: **2^n**, where n in the number of nodes.

**ANS 14)**

**Exploding gradient**:

In a network of n hidden layers, n derivatives will be multiplied together. If the derivatives are large then the gradient will increase exponentially as we propagate down the model until they eventually explode, and this is what we call the problem of exploding gradient.

**Vanishing gradient:**

If the derivatives are small then the gradient will decrease exponentially as we propagate through the model until it eventually vanishes, and this is the vanishing gradient problem.

**ANS 15)** In the neural network terminology:

* **Epochs:**One epoch is when an entire dataset is passed forward and backward through the neural network only once.
* **Batch size**: the number of training examples in one forward/backward pass. The higher the batch size, the more memory space you'll need.
* **Iterations**: Iterations is the number of passes needed to complete one epoch. Each pass using [batch size] number of examples. one pass = one forward pass + one backward pass

**Example**: if you have 6000 training examples, and your batch size is 500, then it will take 12 iterations to complete 1 epoch.