DEEP LEARNING – WORKSHEET 3

1. (B) As number of hidden layers increase, model capacity increases
2. (C) It normalizes (changes) all the input before sending it to the next layer
3. (A) Network will not converge
4. (D) All of these
5. (C) (-4, -4, 3). Find df/dx, df/dy and df/dz using chain rule of differentiation.
6. (B) Simulate the network on a test dataset after every epoch of training. Stop training when the generalization error starts to increase
7. (D) either A or B
8. (A) Freeze all the layers except the last, re-train the last layer.

Reason: If the dataset is mostly similar, the best method would be to train only the last layer, as previous all layers work as feature extractors.

9. (A) Overfitting

(B) Training is too slow

10. (B) sigmoid

(C) softmax

11. In ANNs, the output of a layer is given when weighted sum of inputs and bias is passed through an activation function. The weighted sum of inputs is given as follows:

$$z = b + \sum_{i=1}^{N} w_i x_i$$

Where, b = bias, x(i)= i-th input and w(i)= weight given to i-th input.
Whereas, final output is given as follows:
y = f(z)
Where, y is the final output and f(z) is activation function.
Now, if we do not use activation function, our model will give weighted sum of inputs and biases as final output which will be of linear form. This output is similar to the one we get in linear regression. In this case our ANN won't be able to understand more complex functions which are non-linear and hence out ANN will have no special power. Therefore, use of activation function is necessary to build an ANN.

12. **Forward propagation:** The inputs are provided with weights to the hidden layer. At each hidden layer, we calculate the output of the activation function at each node and this further propagates to the next layer till the final output layer is reached. Since we start from the inputs to the final output layer, we move forward and it is called forward propagation.

Backpropagation: We minimize the cost function by its understanding of how it changes with changing the weights and biases in a neural network. This change is obtained by calculating the gradient at each hidden layer (using the chain rule of derivatives). Since we start from the final cost function and go back each hidden layer, we move backward and thus it is called backward propagation

13. **A) Stochastic Gradient Descent:** Stochastic gradient descent is used to calculate the gradient and update the parameters by using only a single training example.

B) Batch Gradient Descent: Batch gradient descent is used to calculate the gradients for the whole dataset and perform just one update at each iteration.

C) Mini-batch Gradient Descent: Mini-batch gradient descent is a variation of stochastic gradient descent. Instead of a single training example, mini-batch of samples is used. Mini-batch gradient descent is one of the most popular optimization algorithms.

14. Some of the main benefits of using Mini Batch Gradient Descent are as follows:

- It is computationally efficient compared to stochastic gradient descent.
- It improves generalization by finding flat minima.
- It improves convergence by using mini-batches. We can approximate the gradient of the entire training set, which might help to avoid local minima.

15. Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. Transfer learning make use of the knowledge gained while solving one problem and applying it to a different but related problem. For example, knowledge gained while learning to recognize cars can be used to some extent to recognize trucks. Transfer learning is popular now days because many complex open source Deep learning architectures have been built for different type of tasks such as face recognition, image classification, NLP, etc. So rather than developing a new complex architecture from scratch, we use pre trained architectures and depending on the size of our dataset we may train a few last layers or the complete model again for our purpose.