## **DEEP LEARNING**

## **WORKSHEET – 3**

## **Solutions**

1. C) As learning rate increases, 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)
6. B) Simulate the network on a test dataset after every epoch of training. Stop training when the generalization error starts to increase
7. B) Stochastic Gradient Descent
8. A) Freeze all the layers except the last, re-train the last layer
9. B) Training is too slow C) Restrict activations to become too high or low
10. B) sigmoid
11. The activation functions add non-linearity to the output of the neural network - if there were no activation functions the entire network would only be able to model linear functions (so be no better than linear regression).
12. **Forward propagation** is where we would give a certain input to your neural network, say an image or text. The network will calculate the output by propagating the input signal through its layers. In other words, the output form one layer becomes the input to the next one, where the output from the last one is the “answer”. In order to do that accurately, the network needs to be trained. This is done through **Backpropagation**. Basically, during training all the parameters of the network’s layers need to be updated (“optimized”). For this reason, the network needs to know in what “direction” the update should go thus it needs to calculate the so-called gradient with respect to a function known as the loss function, which is a way of saying how “wrong” or “incorrect” the network still is, so hopefully it becomes better next time. Since finding the gradients of all the parameters with respect to this loss function is a complex equation involving the so-called “chain rule”, in computation it is being propagated, where every layer adds its own contribution to that gradient, starting from the last one (hence the name).
13. **Gradient Descent** is an optimization algorithm used for minimizing the cost function in various machine learning algorithms. It is basically used for updating the parameters of the learning model. Types of gradient Descent:

**Batch Gradient Descent:** This is a type of gradient descent which processes all the training examples for each iteration of gradient descent. But if the number of training examples is large, then batch gradient descent is computationally very expensive. Hence if the number of training examples is large, then batch gradient descent is not preferred. Instead, we prefer to use stochastic gradient descent or mini-batch gradient descent.

**Stochastic Gradient Descent:** This is a type of gradient descent which processes 1 training example per iteration. Hence, the parameters are being updated even after one iteration in which only a single example has been processed. Hence this is quite faster than batch gradient descent. But again, when the number of training examples is large, even then it processes only one example which can be additional overhead for the system as the number of iterations will be quite large.

**Mini Batch gradient descent:** This is a type of gradient descent which works faster than both batch gradient descent and stochastic gradient descent. Here b examples whereb<m are processed per iteration. So even if the number of training examples is large, it is processed in batches of b training examples in one go. Thus, it works for larger training examples and that too with lesser number of iterations.

1. The main benefits of Mini-batch Gradient Descent are :-

* Easily fits in the memory
* It is computationally efficient Benefit from vectorization
* If stuck in local minimums, some noisy steps can lead the way out of them
* Average of the training samples produces stable error gradients and convergence

1. **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. It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems**.**

It 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.