**DEEP LEARNING – WORKSHEET 3**

**Q1 to Q8 are MCQs with only one correct answer. Choose the correct option -**

Q1. Which of the following is true about model capacity (where model capacity means the ability of neural network to approximate complex functions)?

A) As dropout ratio increases, model capacity increases B) As number of hidden layers increase, model capacity increases C) As learning rate increases, model capacity increases D) None of the above

Ans. A

Q2. Batch Normalization is helpful because?

A) It is a very efficient backpropagation technique B) It returns back the normalized mean and standard deviation of weights C) It normalizes (changes) all the input before sending it to the next layer D) None of the above

Ans. C

Q3. What if we use a learning rate that’s too large?

A) Network will not converge B) Network will converge C) either A or B D) None of the above

Ans. A

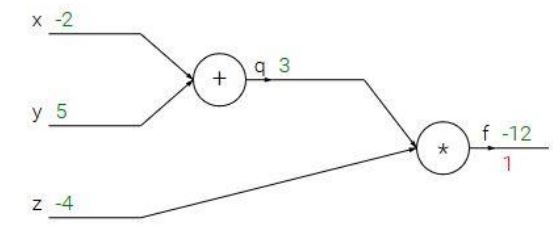
Q4. What are the factors to select the depth of neural network?

i) Type of neural network (e.g. MLP, CNN etc.) ii) Input data iii) Computation power, i.e. Hardware capabilities and software capabilities iv) Learning Rate v) The output function to map

A) 1, 2, 4, 5 B) 2, 3, 4, 5 C) 1, 3, 4, 5 D) All of these

Ans. D

Q5. Suppose you have inputs as x, y, and z with values -2, 5, and -4 respectively.



You have a neuron ‘q’ and neuron ‘f’ with functions: q = x + y f = q \* z Graphical representation of the functions is as follows: What is the gradient of F with respect to x, y, and z? (use chain rule of derivatives to find the solution)

A) (3, -4, -4) B) (-3, 4, 4) C) (-4, -4, 3) D) (4, 4, 3)

Ans. C

Q6. Which of the following statement is the best description of early stopping?

A) Train the network until a local minimum in the error function is reached B) Simulate the network on a test dataset after every epoch of training. Stop training when the generalization error starts to increase C) Add a momentum term to the weight update in the Generalized Delta Rule, so that training converges more quickly D) None of the above

Ans. B

Q7. Which gradient descent technique is more advantageous when the data is too big to handle in RAM simultaneously?

A) Mini Batch Gradient Descent B) Stochastic Gradient Descent C) Full Batch Gradient Descent D) either A or B

Ans. B

Q8. Consider the scenario. The problem you are trying to solve has a small amount of data. Fortunately, you have a pre-trained neural network that was trained on a similar problem. Which of the following methodologies would you choose to make use of this pre-trained network?

A) Freeze all the layers except the last, re-train the last layer B) Assess on every layer how the model performs and only select a few of them C) Fine tune the last couple of layers only D) Re-train the model for the new dataset

Ans. A

**Q9 and Q10 are MCQs with one or more correct answers. Choose all the correct options -**

Q9. Which of the following neural network training challenge can be solved using batch normalization?

A) Overfitting B) Training is too slow C) Restrict activations to become too high or low D) None of these

Ans. B, C

Q10. For a binary classification problem, which of the following activations may be used in output layer?

A) ReLU B) sigmoid C) softmax D) Leaky ReLU

Ans. A, B and D

**Q11 to Q15 are subjective answer type question. Answer them briefly -**

Q11. What will happen if we do not use activation function in artificial neural networks?

Ans. **Activation function** decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to **introduce non-linearity** into the output of a neuron.

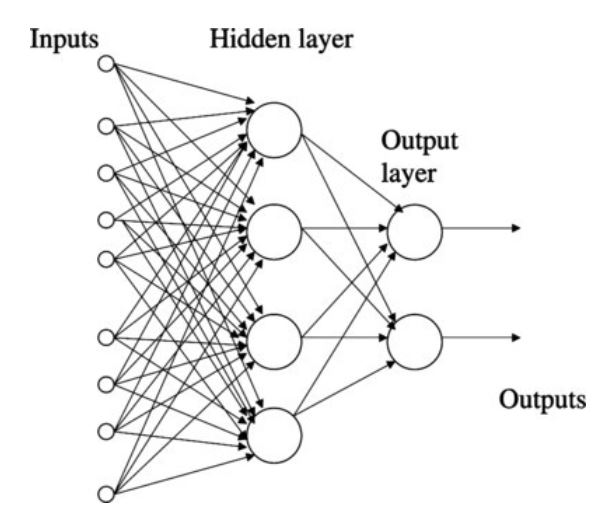
**Explanation :-** We know, neural network has neurons that work in correspondence of *weight, bias* and their respective activation function. In a neural network, we would update the weights and biases of the neurons on the basis of the error at the output. This process is known as *back-propagation*. Activation functions make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases.

Q12. How does forward propagation and backpropagation work in deep learning?

Ans.

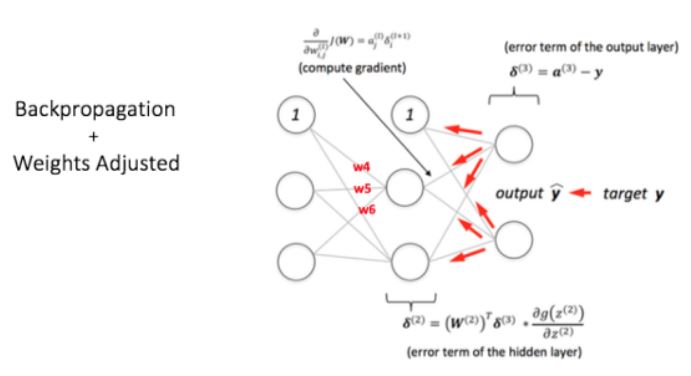
**Feed-forward neural networks:**

* The signals in a feedforward network flow in one direction, from input, through successive hidden layers, to the output.
* The connections between the nodes do not form a cycle as such, it is different from recurrent neural networks.



**Backpropagation** is a **training algorithm** consisting of 2 steps:

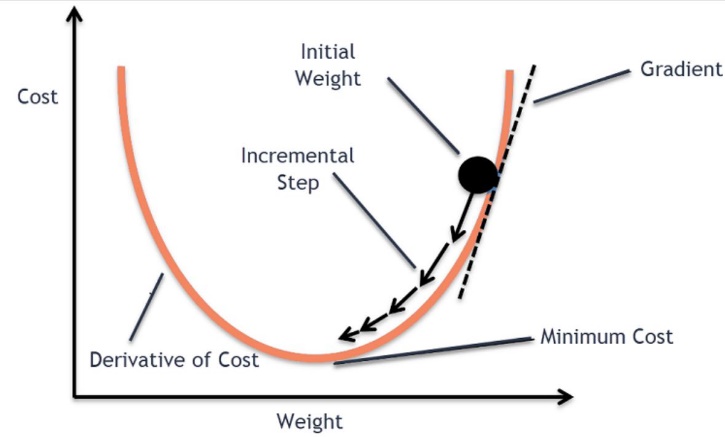
* **Feedforward** the values.
* **Calculate the error** and **propagate it back** to the earlier layers



Forward-propagation is a part of the backpropagation algorithm but comes before back-propagating the signals from the nodes. The basic type of [neural network](https://intellipaat.com/blog/tutorial/machine-learning-tutorial/neural-network-tutorial/) is a multi-layer perceptron, which is a Feed-forward backpropagation neural network.

Q13. Explain briefly the following variant of Gradient Descent: Stochastic, Batch, and Mini-batch?

Ans. **Gradient Descent:** Gradient descent is a [first-order](https://en.wikipedia.org/wiki/Category:First_order_methods) [iterative](https://en.wikipedia.org/wiki/Iterative_algorithm) [optimization](https://en.wikipedia.org/wiki/Mathematical_optimization) [algorithm](https://en.wikipedia.org/wiki/Algorithm) for finding the minimum of a function. The goal of the gradient descent is to minimise a given function which, in our case, is the loss function of the neural network. To achieve this goal, it performs two steps iteratively.



* Compute the slope (gradient) that is the first-order derivative of the function at the current point
* Move-in the opposite direction of the slope increase from the current point by the computed amount

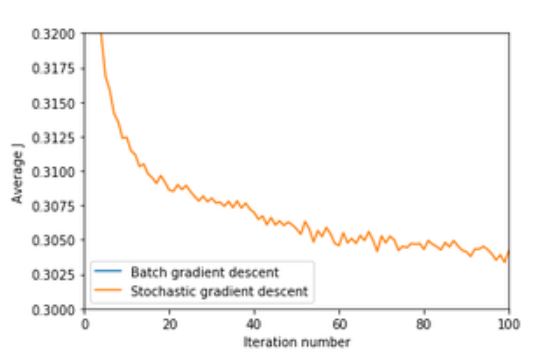
# Batch Gradient Descent: In Batch Gradient Descent, all the training data is taken into consideration to take a single step. We take the average of the gradients of all the training examples and then use that mean gradient to update our parameters. So that’s just one step of gradient descent in one epoch. Batch Gradient Descent is great for convex or relatively smooth error manifolds. In this case, we move somewhat directly towards an optimum solution.

# 

The graph of cost vs epochs is also quite smooth because we are averaging over all the gradients of training data for a single step. The cost keeps on decreasing over the epochs.

**Stochastic Gradient Descent :** In Batch Gradient Descent we were considering all the examples for every step of Gradient Descent. But what if our dataset is very huge. Deep learning models crave for data. The more the data the more chances of a model to be good. Suppose our dataset has 5 million examples, then just to take one step the model will have to calculate the gradients of all the 5 million examples. This does not seem an efficient way. To tackle this problem we have Stochastic Gradient Descent. In Stochastic Gradient Descent (SGD), we consider just one example at a time to take a single step. We do the following steps in one epoch for SGD:

1. Take an example
2. Feed it to Neural Network
3. Calculate it’s gradient
4. Use the gradient we calculated in step 3 to update the weights
5. Repeat steps 1–4 for all the examples in training dataset



Since we are considering just one example at a time the cost will fluctuate over the training examples and it will not necessarily decrease. But in the long run, you will see the cost decreasing with fluctuations.

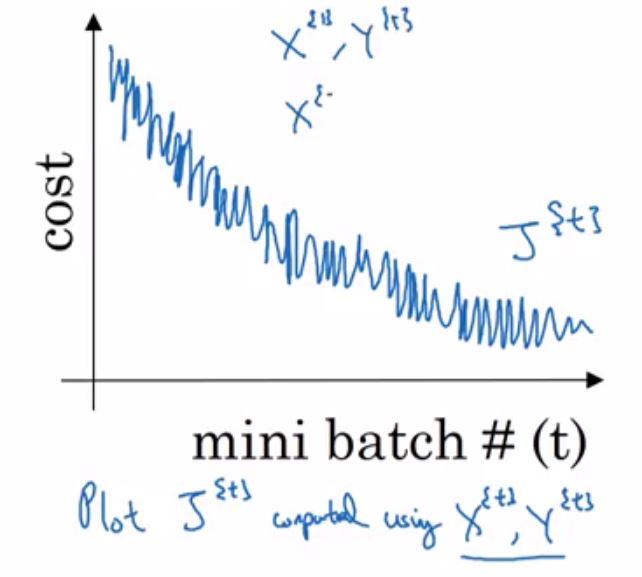
Q14. What are the main benefits of Mini-batch Gradient Descent?

Ans. **Mini Batch Gradient Descent:**  We have seen the Batch Gradient Descent. We have also seen the Stochastic Gradient Descent. Batch Gradient Descent can be used for smoother curves. SGD can be used when the dataset is large. Batch Gradient Descent converges directly to minima. SGD converges faster for larger datasets. But, since in SGD we use only one example at a time, we cannot implement the vectorized implementation on it. This can slow down the computations. To tackle this problem, a mixture of Batch Gradient Descent and SGD is used.

Neither we use all the dataset all at once nor we use the single example at a time. We use a batch of a fixed number of training examples which is less than the actual dataset and call it a mini-batch. Doing this helps us achieve the advantages of both the former variants we saw. So, after creating the mini-batches of fixed size, we do the following steps in one epoch:

1. Pick a mini-batch
2. Feed it to Neural Network
3. Calculate the mean gradient of the mini-batch
4. Use the mean gradient we calculated in step 3 to update the weights
5. Repeat steps 1–4 for the mini-batches we created

Just like SGD, the average cost over the epochs in mini-batch gradient descent fluctuates because we are averaging a small number of examples at a time.



So, when we are using the mini-batch gradient descent we are updating our parameters frequently as well as we can use vectorized implementation for faster computations.

Q15. What is transfer learning?

Ans. Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task. Transfer learning and domain adaptation refer to the situation where what has been learned in one setting … is exploited to improve generalization in another setting. Transfer learning is an optimization that allows rapid progress or improved performance when modeling the second task. Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned.

In transfer learning, we first train a base network on a base dataset and task, and then we repurpose the learned features, or transfer them, to a second target network to be trained on a target dataset and task. This process will tend to work if the features are general, meaning suitable to both base and target tasks, instead of specific to the base task.