ADVANCED DEEP LEARNING INTERVIEW QUESTIONS

1. What are some of the limitations of Deep Learning?

There are a few disadvantages of Deep Learning as mentioned below:

- Networks in Deep Learning require a huge amount of data to train well.
- Deep Learning concepts can be complex to implement sometimes.
- Achieving a high amount of model efficiency is difficult in many cases.

These are some of the vital advanced deep learning interview questions that you have to know about!

2. What are the variants of gradient descent?

There are three variants of gradient descent as shown below:

- Stochastic gradient descent: A single training example is used for the calculation of gradient and for updating parameters.
- Batch gradient descent: Gradient is calculated for the entire dataset, and parameters are updated at every iteration.
- Mini-batch gradient descent: Samples are broken down into smaller-sized batches and then worked on as in the case of stochastic gradient descent.

3. Why is mini-batch gradient descent so popular?

Mini-batch gradient descent is popular as:

- It is more efficient when compared to stochastic gradient descent.
- Generalization is done by finding the flat minima.
- It helps avoid the local minima by allowing the approximation of the gradient for the entire dataset.

4. What are deep autoencoders?

Deep autoencoders are an extension of the regular autoencoders. Here, the first layer is responsible for the first-order function execution of the input. The second layer will take care of the second-order functions, and it goes on.

Usually, a deep autoencoder is a combination of two or more symmetrical deep-belief networks where:

- The first five shallow layers consist of the encoding part
- The other layers take care of the decoding part

On the next set of Deep Learning questions, let us look further into the topic.

5. Why is the Leaky ReLU function used in Deep Learning?

Leaky ReLU, also called LReL, is used to manage a function to allow the passing of small-sized negative values if the input value to the network is less than zero.

6. What are some of the examples of supervised learning algorithms in Deep Learning?

There are three main supervised learning algorithms in Deep Learning:

- Artificial neural networks
- Convolutional neural networks
- Recurrent neural networks

7. What are some of the examples of unsupervised learning algorithms in Deep Learning?

There are three main unsupervised learning algorithms in Deep Learning:

- Autoencoders
- Boltzmann machines
- Self-organizing maps

Next up, let us look at more neural network interview questions that will help you ace the interviews.

8. Can we initialize the weights of a network to start from zero?

Yes, it is possible to begin with zero initialization. However, it is not recommended to use because setting up the weights to zero initially will cause all of the neurons to produce the same output and the same gradients when performing backpropagation. This means that the network will not have the ability to learn at all due to the absence of asymmetry between each of the neurons.

9. What is the meaning of valid padding and same padding in CNN?

- Valid padding: It is used when there is no requirement for padding. The output matrix will have the dimensions (n – f + 1) X (n – f + 1) after convolution.
- Same padding: Here, padding elements are added all around the output matrix. It will have the same dimensions as the input matrix.

10. What are some of the applications of transfer learning in Deep Learning?

Transfer learning is a scenario where a large model is trained on a dataset with a large amount of data and this model is used on simpler datasets, thereby resulting in extremely efficient and accurate neural networks.

The popular examples of transfer learning are in the case of:

- BERT
- ResNet
- GPT-2
- VGG-16

11. How is the transformer architecture better than RNNs in Deep Learning?

With the use of sequential processing, programmers were up against:

- The usage of high processing power
- The difficulty of parallel execution

12. What are the steps involved in the working of an LSTM network?

There are three main steps involved in the working of an LSTM network:

- The network picks up the information that it has to remember and identifies what to forget.
- Cell state values are updated based on Step 1.
- The network calculates and analyzes which part of the current state should make it to the output.

13. What are the elements in TensorFlow that are programmable?

In TensorFlow, users can program three elements:

- Constants
- Variables
- Placeholders

14. What is the meaning of bagging and boosting in Deep Learning?

Bagging is the concept of splitting a dataset and randomly placing it into bags for training the model.

Boosting is the scenario where incorrect data points are used to force the model to produce the wrong output. This is used to retrain the model and increase accuracy.

15. What are generative adversarial networks (GANs)?

Generative adversarial networks are used to achieve generative modeling in Deep Learning. It is an unsupervised task that involves the discovery of patterns in the input data to generate the output.

The generator is used to generate new examples, while the discriminator is used to classify the examples generated by the generator.

16. Why are generative adversarial networks (GANs) so popular?

Generative adversarial networks are used for a variety of purposes. In the case of working with images, they have a high amount of traction and efficient working.

- Creation of art: GANs are used to create artistic images, sketches, and paintings.
- Image enhancement: They are used to greatly enhance the resolution of the input images.
- Image translation: They are also used to change certain aspects, such as day to night and summer to winter, in images easily.

17. How does the choice of cost function impact the convergence properties of a deep neural network?

The convergence properties of a deep neural network are heavily influenced by the choice of cost function as it defines the gradient landscape. For example, cross-entropy loss creates a more aggressive gradient, providing more “signal” per update when the predictions are wrong, thus often leading to faster convergence, particularly in classification problems, where the output probabilities are being modeled.

18. Can you explain the advantages of using a cross-entropy loss over mean squared error in classification tasks?

Cross-entropy loss tends to work better for classification since it penalizes incorrect classifications more heavily than mean squared error (MSE), which can lead to quicker and more stable training. Cross-entropy aligns with the gradient updates of probabilistic outcomes, directly correlating to the likelihood of predicting true labels.

19. Describe a scenario where you would use a custom loss function and how you would go about implementing it.

If a problem has a unique cost structure (e.g., a different cost for different types of misclassifications), I would design a custom loss function. This requires ensuring the custom loss is differentiable for gradient-based optimization, and I would use automatic differentiation capabilities of deep learning frameworks like TensorFlow or PyTorch for implementation.

20. Describe the role of convolutional layers in CNNs and how they differ from fully connected layers regarding feature extraction.

Convolutional layers act as feature extractors that slide across input space and produce feature maps, highlighting features like edges or textures, whereas fully connected layers take those features to learn non-linear combinations that aid in classification.

21. Discuss the concept of bias-variance trade-off in the context of neural network weights and model complexity.

The bias-variance trade-off in the context of neural networks is about finding the right balance between a model that is too simple (high bias) and one that is too complex (high variance). If the weights are poorly chosen, the network can either fail to capture the underlying patterns (underfitting) or capture too much noise (overfitting).

22. Explain the concept of receptive field in a CNN and how it relates to the architecture's ability to recognize patterns of different scales.

The receptive field in a CNN is the area of the input image that a neuron „sees.‟ Initially, it captures basic features like edges, and in deeper layers, it represents more complex patterns due to larger receptive fields. The architecture is designed so that these fields grow progressively to recognize objects of various sizes, balancing the need to detect both fine details and broader patterns.

23. How does the concept of feature map concatenation in networks like DenseNet affect the performance and parameter efficiency of a model?

Feature map concatenation in DenseNet architectures allows each layer to access feature maps from all preceding layers, promoting feature reuse, which significantly improves the parameter efficiency. By concatenating, instead of summing, we provide subsequent layers with a rich, diverse set of features. This enhances the network‟s representational power and tends to improve model performance, particularly on complex tasks. Moreover, it leads to a reduction in the number of parameters compared to traditional CNNs, since each layer is thinner and only contributes a small number of feature maps.

24. How can we detect and prevent the vanishing or exploding gradients problem in deep neural networks?

To detect vanishing or exploding gradients, monitor the magnitude of gradients during backpropagation. If they are too small or too large, that‟s a sign of trouble. To prevent these issues, we typically use better weight initialization methods like Xavier or He initialization, employ batch normalization, use appropriate activation functions like ReLU, and potentially apply gradient clipping to cap the gradients during training.

25. Discuss how L1 and L2 regularization terms affect the distribution of weights in a neural network model.

L1 regularization, also known as Lasso regularization, tends to push the weights towards zero, creating a sparse solution where some weights can become exactly zero. This is useful for feature selection in high-dimensional datasets. On the other hand, L2 regularization, also known as Ridge regularization, encourages the weights to be small but not necessarily zero, leading to a more diffuse, small weight distribution. It helps in preventing overfitting by penalizing the magnitude of the weights without promoting sparsity.

26. How would you design a CNN to handle input images of varying sizes?

To design a CNN for handling input images of varying sizes, one approach is to incorporate global average pooling layers towards the end of the network. This allows the network to aggregate feature information efficiently, resulting in a fixed-length output regardless of the input image‟s dimensions, which is particularly useful when you‟re dealing with images of different resolutions.

27. What do you know about Dropout?

Dropout is a regularization approach that helps to avoid overfitting and improve the generalizability of the dataset. During the training, randomly selected neurons are ignored for each pass or update of the model; this means that during each iteration, a random subset of neurons is excluded and the model is trained on the remaining neurons.

28. What is the Vanishing Gradient Problem in Artificial Neural Networks?

The vanishing gradient problem is part of an artificial neural network with a gradient-based learning method. In this method, each of the neural networks receives the weights and updates them proportional to the partial derivative of the error function concerning the current weight in each iteration.

29. What exactly do you mean by Exploding and Vanishing Gradients?

Exploding Gradient: Exploding gradient is a problem that occurs during the training of deep neural networks, which leads to the gradients of the network losing weight.
Vanishing Gradient: Vanishing gradient is a problem that occurs when gradients used to update the network become very small as they are back propagated from the output layer to the earlier layers.

30. What is the difference between Batch Normalization, Instance Normalization, and Layer Normalization?

Batch Normalization: In batch normalization, the mean and variance are calculated for each channel across all samples and their relative dimensions, i.e., the height of each activation map (H) and the width of each activation map(W).

Instance Normalization: In Instance normalization, the mean and variance are calculated for each channel for each sample across both the height of each activation map (H) and the width of each activation map (W).

Layer Normalization: In layer normalization, the mean and the variance are calculated for each sample across all channels and their relative dimensions i.e., the height of each activation map (H) and the width of each activation map (W).

31. What's the difference between GAN and Autoencoders?

GAN: Generative Adversarial Networks (GAN) is used as an adversarial feedback loop to learn how to generate some information that seems real.
Autoencoder: An autoencoder is used to learn some input information with high efficiency and, subsequently, how to reconstruct the input from its compressed form.

32. What's the difference between Recurrent Neural Networks and Recursive Neural Networks?

Recurrent Neural Network: It is used for sequential inputs where the time factor is the main differentiating factor between the elements of the sequence. Due to this, it‟s commonly used in time series, and the weights are shared with the length of the sequence.
Recursive Neural Network: It is more like a hierarchical network where there is no time aspect to the input sequence, but the input has to be processed hierarchically in a tree fashion, and the weights are shared at every node.

33. What is the importance of using the Non-linear Activation Function?

Neural networks with only linear activation do not gain from increasing the number of layers in them since all linear functions add up to a single linear function.

Non-linear activation functions allow us to stack different layers, and they will not be treated like a single layer as in the linear activation layer.

The derivation of a linear function has no relation to the input, so it is not possible to use backpropagation when it comes to linear functions. Non-linear functions allow backpropagation because they can be differentiated, and their derivative is related to the input.