DEEP LEARNING

Interview Guide

simplilearn

Ace Your Deep Learning Interview

Deep learning is a subset of machine learning where artificial neural networks understand and learn from large amounts of data. Similarly to how we learn from our day to day experiences, the deep learning algorithm would perform a task repeatedly, every time tweaking it bit by bit to improve the responses and the final outcome. Deep Learning is one of the fastest-growing IT fields and has a set of techniques that permit machines to predict outputs from a layered set of inputs. Deep Learning is being embraced by companies all over the world, and anyone with software and data skills can find numerous job opportunities in this field.

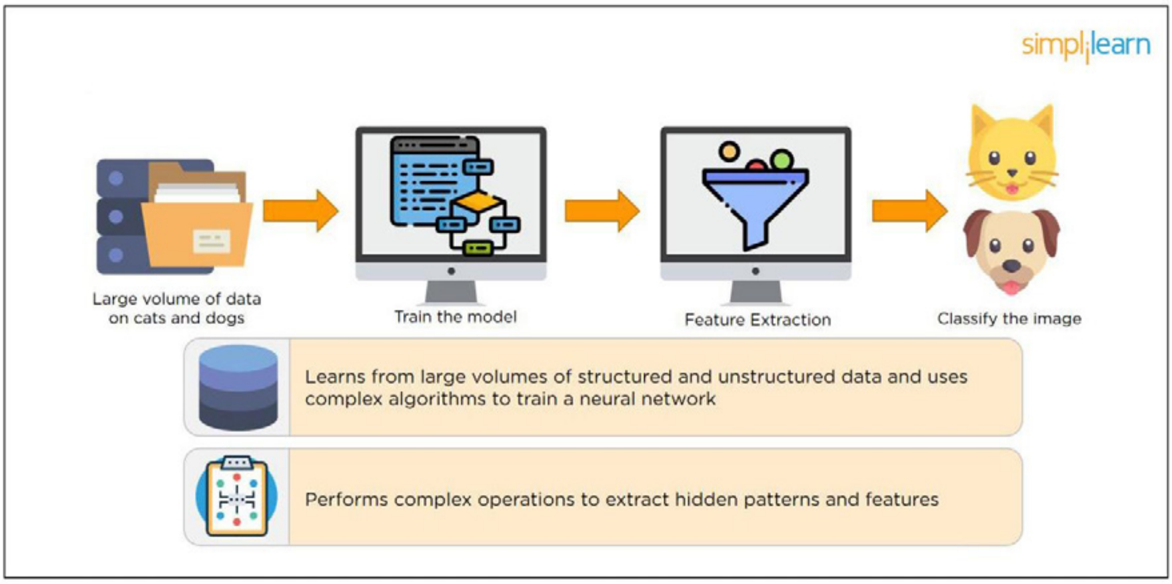
If you want to start a career in deep learning, you will come across various in-depth learning interviews. Worried? Don't, because we have clubbed a list of the 30+ most popular questions you can expect in an interview. So prepare ahead of time, and crack your Deep Learning Interview in the first go.

Topics Covered

- Basics of Deep Learning
- Artificial Neural Networks
- Types of Activation Functions
- Hyperparameters
- Gradient Descent and Backpropagation
- CNN, RNN and GAN
- TensorFlow

Q: What Is Deep Learning?

A: Deep Learning involves taking large volumes of structured or unstructured data and using complex algorithms to train neural networks. It performs complex operations to extract hidden patterns and features (for instance, distinguishing the image of a cat from that of a dog).

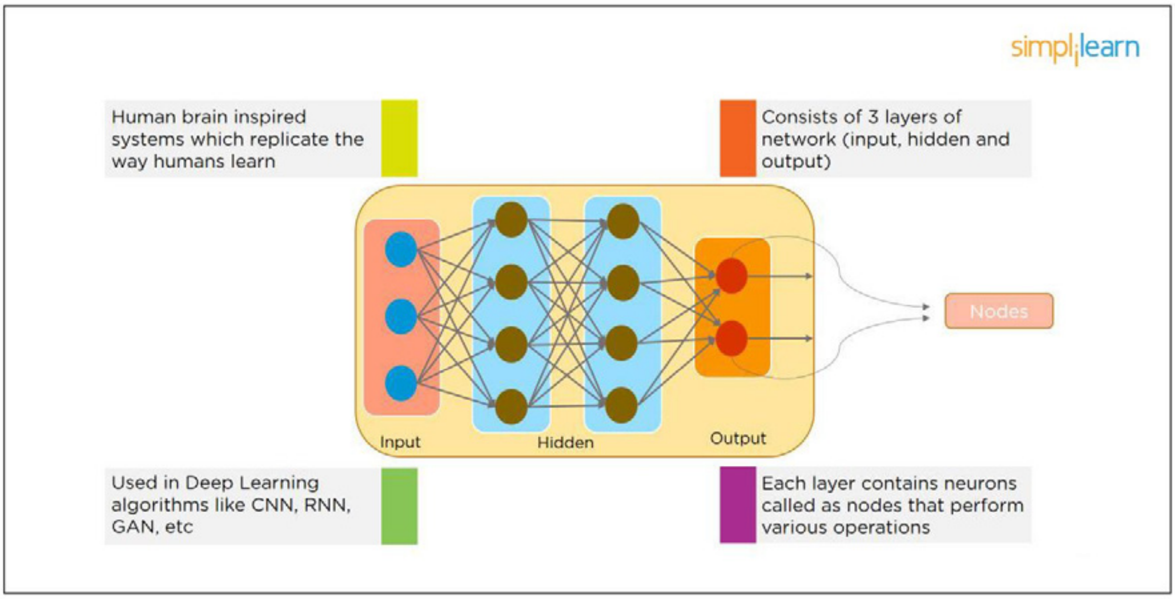


Learns from large volumes of structured and unstructured data and uses complex algorithms to train a neural network

Performs complex operations to extract hidden patterns and features

Q: What Is a Neural Network?

A: Neural Networks replicate the way humans learn, inspired by how the neurons in our brains fire, only much simpler.



Human brain inspired systems which replicate the way humans learn

Consists of 3 layers of network (input, hidden and output)

Used in Deep Learning algorithms like CNN, RNN, GAN, etc

Each layer contains neurons called as nodes that perform various operations

The most common Neural Networks consist of three network layers:

1. An input layer
2. A hidden layer (this is the most important layer where feature extraction takes place, and adjustments are made to train faster and function better)
3. An output layer

Each sheet contains neurons called "nodes," performing various operations. Neural Networks are used in deep learning algorithms like CNN, RNN, GAN, etc.

Q: What Is a Multi-layer Perceptron (MLP)?

A: As in Neural Networks, MLPs have an input layer, a hidden layer, and an output layer. It has the same structure as a single layer perceptron with one or more hidden layers. A single layer perceptron can classify only linear separable classes with binary output (0,1), but MLP can classify nonlinear classes.

Except for the input layer, each node in the other layers uses a nonlinear activation function. This means the input layers, the data coming in, and the activation function is based upon all nodes and weights being added together, producing the output. MLP uses a supervised learning method called "backpropagation." In backpropagation, the neural network calculates the error with the help of cost function. It propagates this error backward from where it came (adjusts the weights to train the model more accurately).

Q: What Is Data Normalization, and Why Do We Need It?

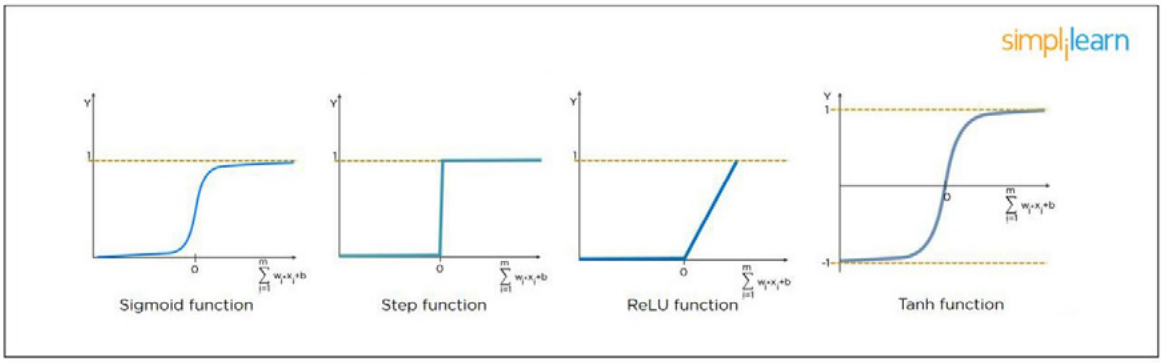
A: The process of standardizing and reforming data is called "Data Normalization." It's a pre-processing step to eliminate data redundancy. Often, data comes in, and you get the same information in different formats. In these cases, you should rescale values to fit into a particular range, achieving better convergence.

Q: What Is the Boltzmann Machine?

A: One of the most basic Deep Learning models is a Boltzmann Machine, resembling a simplified version of the Multi-Layer Perceptron. This model features a visible input layer and a hidden layer - just a two-layer neural net that makes stochastic decisions as to whether a neuron should be on or off. Nodes are connected across layers, but no two nodes of the same layer are connected.

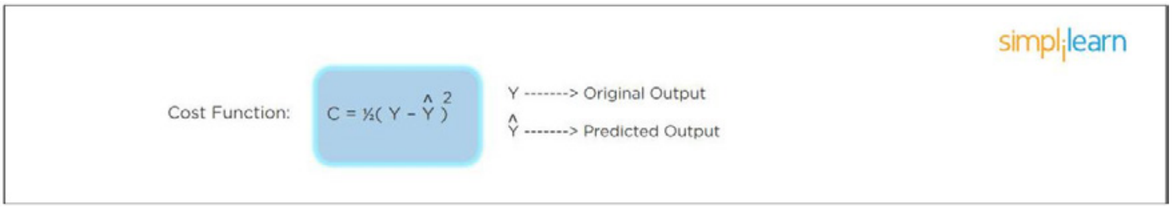
Q: What Is the Role of Activation Functions in a Neural Network?

A: At the most basic level, an activation function decides whether a neuron should be fired or not. It accepts the weighted sum of the inputs and bias as input to any activation function. Step functioWn, Sigmoid, ReLU, Tanh, and Softmax are examples of activation functions.



Q: What Is the Cost Function?

A: Also referred to as "loss" or "error," cost function is a measure to evaluate how good your model's performance is. It's used to compute the error of the output layer during backpropagation. We push that error backward through the neural network and use that during the different training functions.



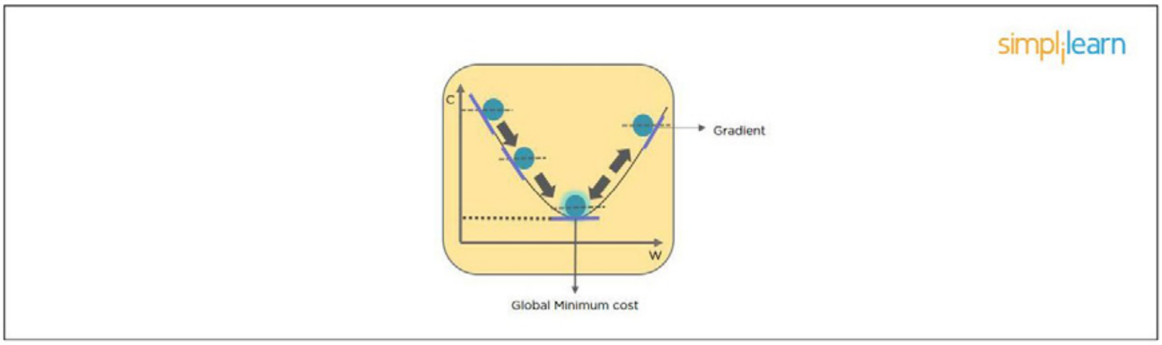
Cost Function: $C = \frac{1}{2}(Y - \hat{Y})^2$

Y -------> Original Output

$\hat{Y}$ -------> Predicted Output

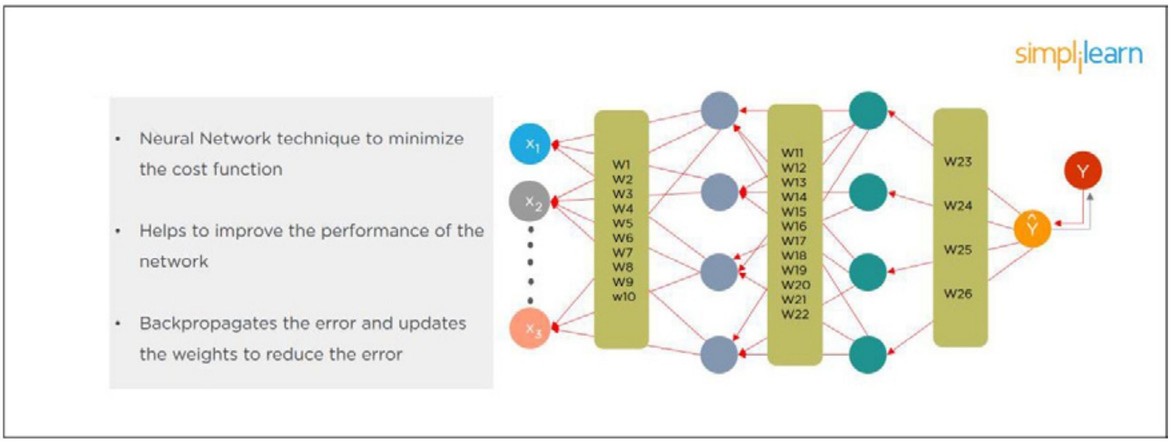
Q: What Is Gradient Descent?

A: Gradient Descent is an optimal algorithm to minimize the cost function or to minimize an error. The aim is to find the local-global minima of a function. This determines the direction the model should take to reduce the error.



Q: What Do You Understand by Backpropagation?

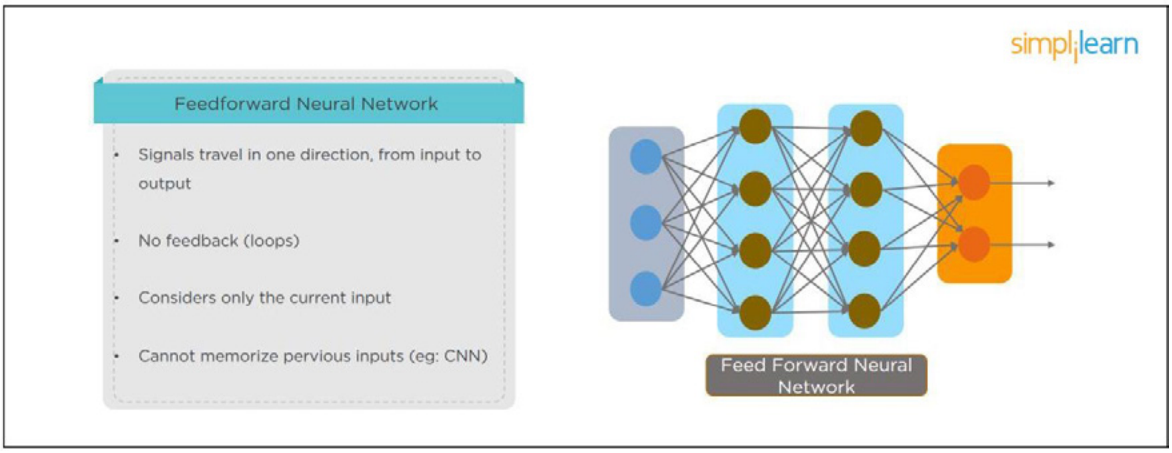
A: Backpropagation is a technique to improve the performance of the network. It backpropagates the error and updates the weights to reduce the error.



- Neural Network technique to minimize the cost function
- Helps to improve the performance of the network
- Backpropagates the error and updates the weights to reduce the error

Q: What Is the Difference Between a Feedforward Neural Network and Recurrent Neural Network?

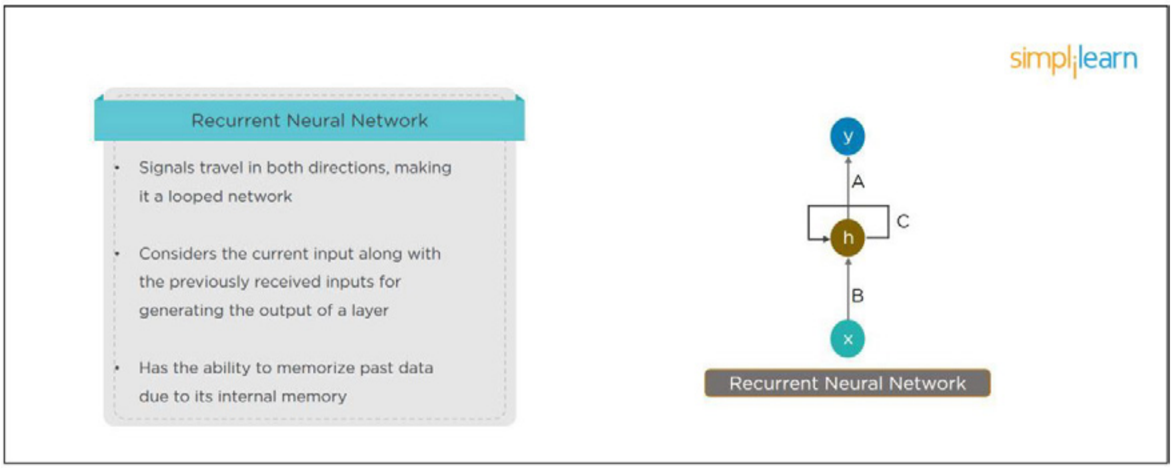
A: A Feedforward Neural Network signals travel in one direction from input to output. There are no feedback loops; the network considers only the current input. It cannot memorize previous inputs (e.g., CNN).



Feedforward Neural Network

- Signals travel in one direction, from input to output
- No feedback (loops)
- Considers only the current input
- Cannot memorize pervious inputs (eg: CNN)

A Recurrent Neural Network's signals travel in both directions, creating a looped network. It considers the current input with the previously received inputs for generating the output of a layer and can memorize past data due to its internal memory.



Recurrent Neural Network

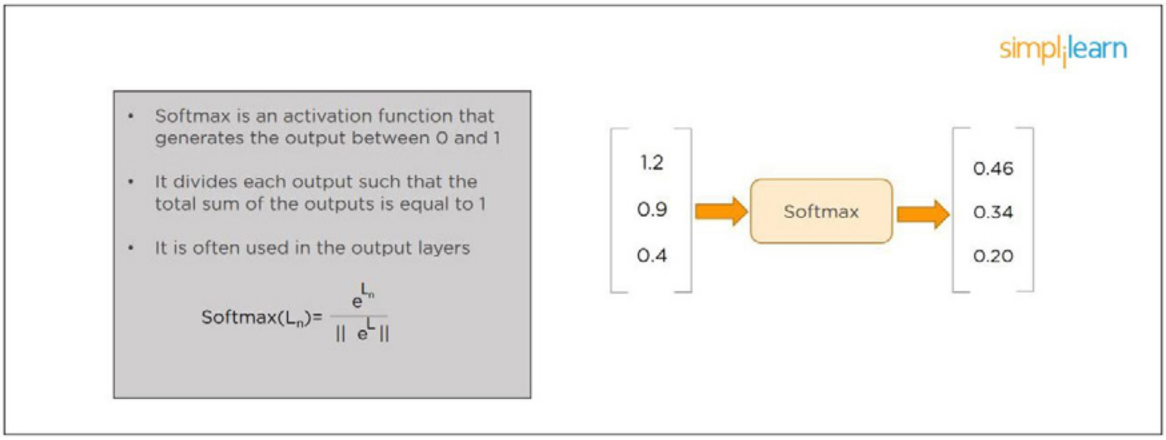
- Signals travel in both directions, making it a looped network
- Considers the current input along with the previously received inputs for generating the output of a layer
- Has the ability to memorize past data due to its internal memory

Q: What Are the Applications of a Recurrent Neural Network (RNN)?

A: The RNN can be used for sentiment analysis, text mining, and image captioning. Recurrent Neural Networks can also address time series problems such as predicting the prices of stocks in a month or quarter.

Q: What Are the Softmax and ReLU Functions?

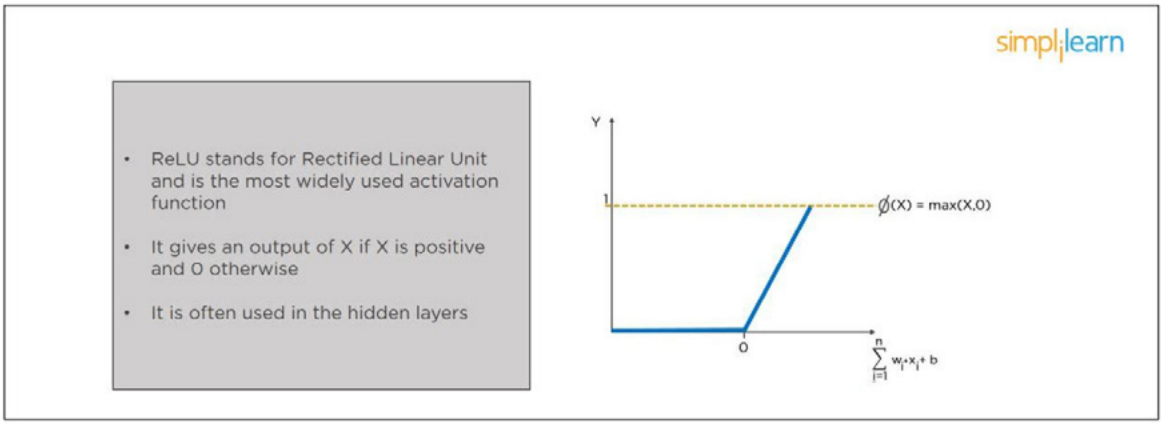
A: Softmax is an activation function that generates the output between zero and one. It divides each output, such that the total sum of the outputs is equal to one. Softmax is often used for output layers.



- Softmax is an activation function that generates the output between 0 and 1
- It divides each output such that the total sum of the outputs is equal to 1
- It is often used in the output layers

$$\text{Softmax}(L_n) = \frac{e^{L_n}}{\| e^{L} \|}$$

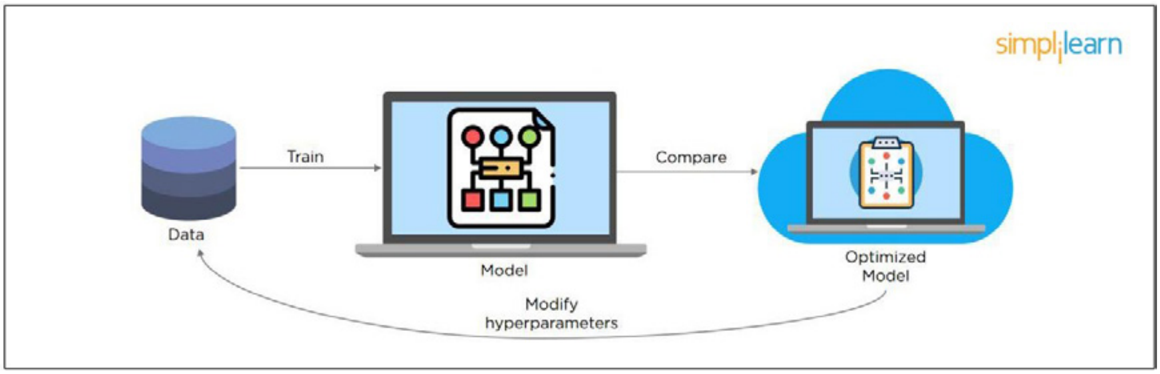
ReLU (or Rectified Linear Unit) is the most widely used activation function. It gives an output of X if X is positive and zeroes otherwise. ReLU is often used for hidden layers.



- ReLU stands for Rectified Linear Unit and is the most widely used activation function
- It gives an output of X if X is positive and 0 otherwise
- It is often used in the hidden layers

Q: What Are Hyperparameters?

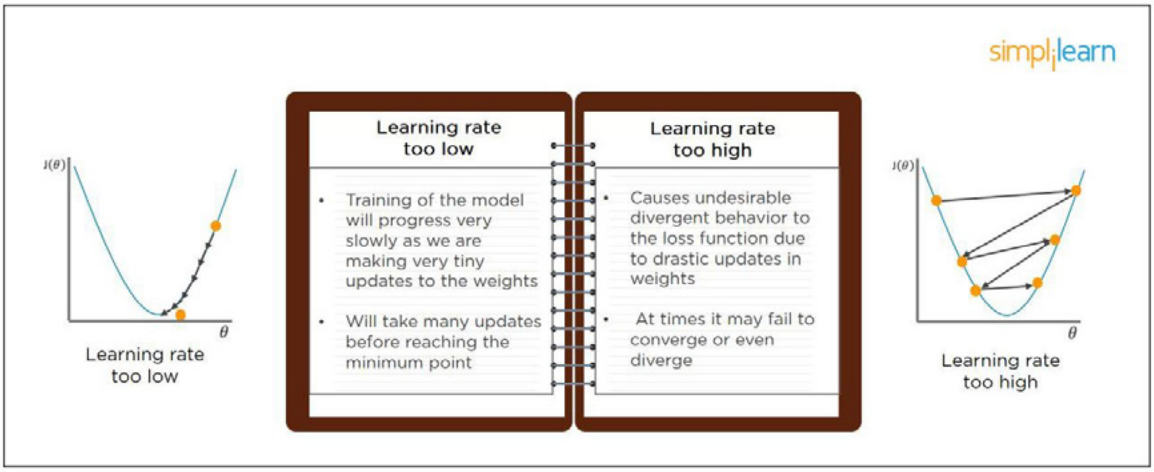
A: With neural networks, you're usually working with hyperparameters once the data is formatted correctly. A hyperparameter is a parameter whose value is set before the learning process begins. It determines how a network is trained and the structure of the network (such as the number of hidden units, the learning rate, epochs, etc.).



Q: What Will Happen If the Learning Rate Is Set Too Low or Too High?

A: When your learning rate is too low, training of the model will progress very slowly as we are making minimal updates to the weights. It will take many updates before reaching the minimum point.

If the learning rate is set too high, this causes undesirable divergent behavior to the loss function due to drastic updates in weights. It may fail to converge (model can give a good output) or even diverge (data is too chaotic for the network to train).



Learning rate too low

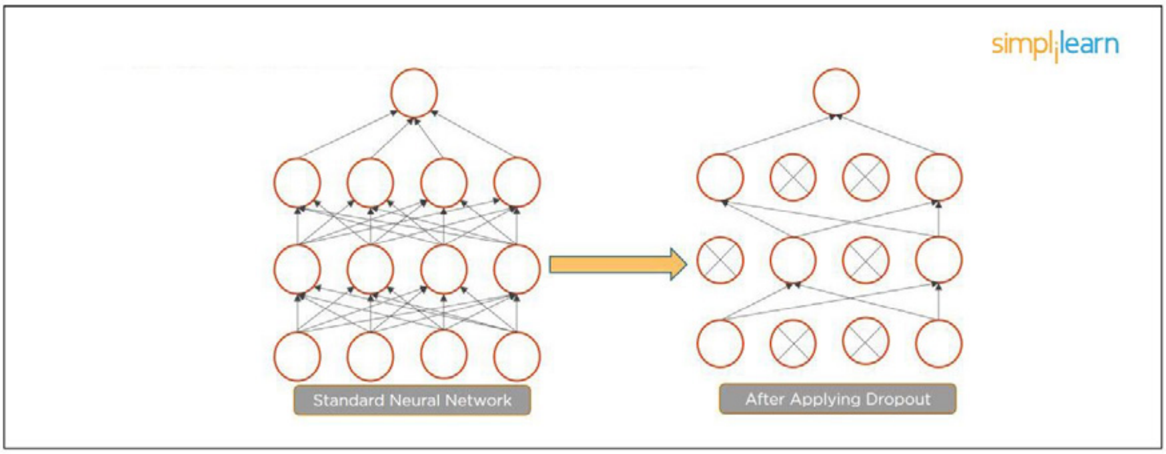
- Training of the model will progress very slowly as we are making very tiny updates to the weights
- Will take many updates before reaching the minimum point

Learning rate too high

- Causes undesirable divergent behavior to the loss function due to drastic updates in weights
- At times it may fail to converge or even diverge

Q: What Is Dropout and Batch Normalization?

A: Dropout is a technique of dropping out hidden and visible units of a network randomly to prevent overfitting of data (typically dropping 20 percent of the nodes). It doubles the number of iterations needed to converge the network.



Batch normalization is the technique to improve the performance and stability of neural networks by normalizing the inputs in every layer so that they have mean output activation of zero and standard deviation of one.

Q: What Is the Difference Between Batch Gradient Descent and Stochastic Gradient Descent?

A: Batch Gradient Descent

- The batch gradient computes the gradient using the entire dataset.
- It takes time to converge because the volume of data is huge, and weights update slowly.

Stochastic Gradient Descent

- The stochastic gradient computes the gradient using a single sample.
- It converges much faster than the batch gradient because it updates weight more frequently.

Q: What Is Overfitting and Underfitting, and How to Combat Them?

A: Overfitting occurs when the model learns the details and noise in the training data to the degree that it adversely impacts the execution of the model on new information. It is more likely to occur with nonlinear models that have more flexibility when learning a target function. An example would be if a model is looking at cars and trucks, but only recognizes trucks that have a specific box shape. It might not be able to notice a flatbed truck because there's only a particular kind of truck it saw in training. The model performs well on training data, but not in the real world.

Underfitting alludes to a model that is neither well-trained on data nor can generalize to new information. This usually happens when there is less and incorrect data to train a model. Underfitting has both poor performance and accuracy.

To combat overfitting and underfitting, you can resample the data to estimate the model accuracy (k-fold cross-validation) and by having a validation dataset to evaluate the model.

Q: How Are Weights Initialized in a Network?

A: There are two methods here: we can either initialize the weights to zero or assign them randomly.

Initializing all weights to 0: This makes your model similar to a linear model. All the neurons and every layer perform the same operation, giving the same output and making the deep net useless.

Initializing all weights randomly: Here, the weights are assigned randomly by initializing them very close to 0. It gives better accuracy to the model since every neuron performs different computations. This is the most commonly used method.

Q: What Are the Different Layers on CNN?

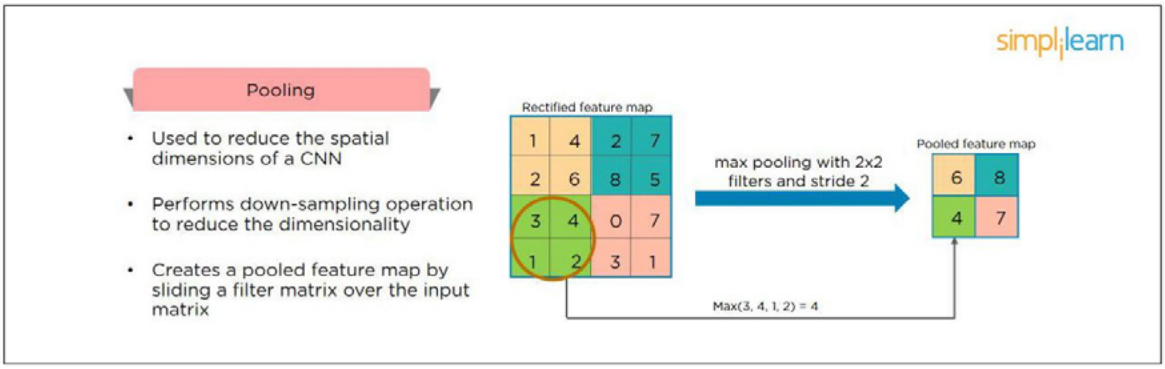
A: There are four layers in CNN:

1. Convolutional Layer - the layer that performs a convolutional operation, creating several smaller picture windows to go over the data.
2. ReLU Layer - it brings non-linearity to the network and converts all the negative pixels to zero. The output is a rectified feature map.
3. Pooling Layer - pooling is a down-sampling operation that reduces the dimensionality of the feature map.
4. Fully Connected Layer - this layer recognizes and classifies the objects in the image.

Master the deep learning concepts and TensorFlow open-source framework with the Deep Learning Course. Check out the course preview today.

Q: What Is Pooling on CNN, and How Does It Work?

A: Pooling is used to reduce the spatial dimensions of a CNN. It performs down-sampling operations to reduce the dimensionality and creates a pooled feature map by sliding a filter matrix over the input matrix.



Pooling

- Used to reduce the spatial dimensions of a CNN
- Performs down-sampling operation to reduce the dimensionality
- Creates a pooled feature map by sliding a filter matrix over the input matrix

Rectified feature map

1	4	2	7
2	6	8	5
3	4	0	7
1	2	3	1

max pooling with 2x2 filters and stride 2

Pooled feature map

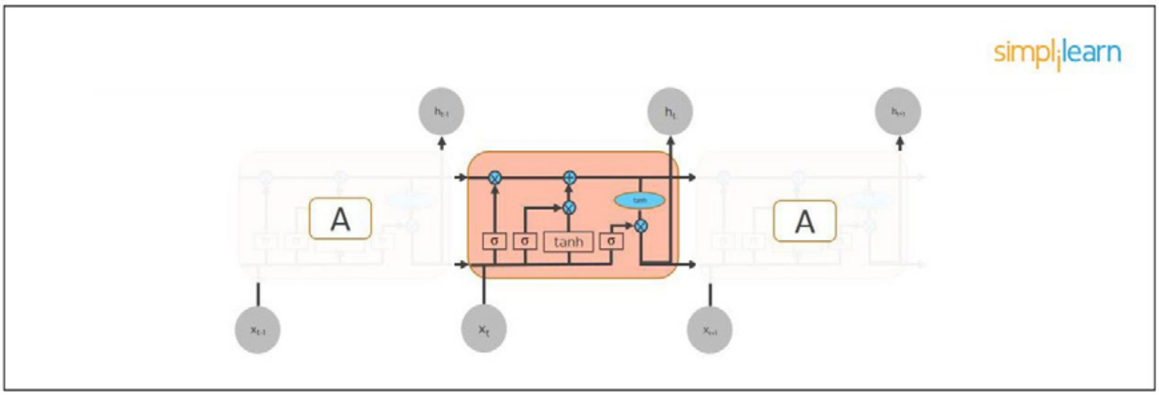
6	8
4	7

Max(3, 4, 1, 2) = 4

Q: How Does an LSTM Network Work?

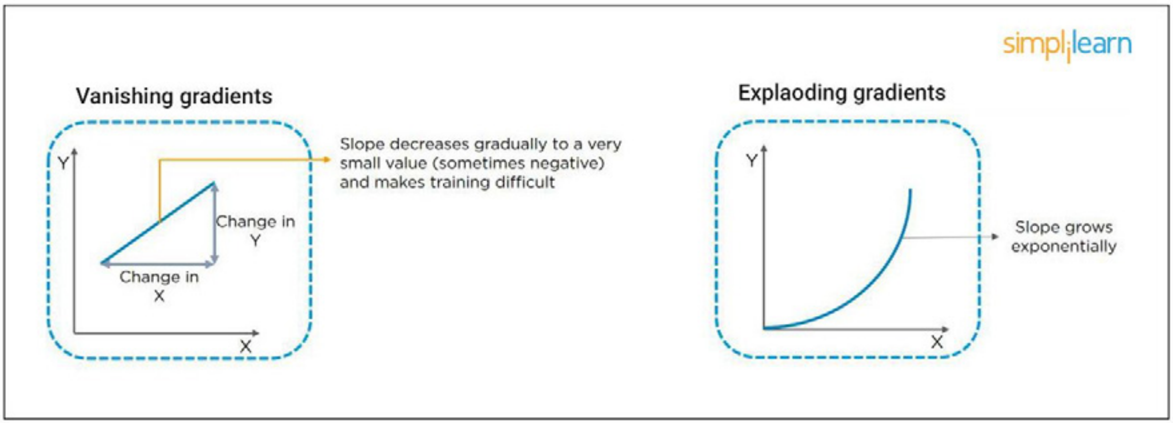
A: Long-Short-Term Memory (LSTM) is a special kind of recurrent neural network capable of learning long-term dependencies, remembering information for long periods as its default behavior. There are three steps in an LSTM network:

- Step 1: The network decides what to forget and what to remember.
- Step 2: It selectively updates cell state values.
- Step 3: The network decides what part of the current state makes it to the output.



Q: What Are Vanishing and Exploding Gradients?

A: While training an RNN, your slope can become either too small or too large; this makes the training difficult. When the slope is too small, the problem is known as a "Vanishing Gradient." When the slope tends to grow exponentially instead of decaying, it's referred to as an "Exploding Gradient." Gradient problems lead to long training times, poor performance, and low accuracy.


simplilearn
Vanishing gradients
Y
Change in Y
Change in X
X
Slope decreases gradually to a very small value (sometimes negative) and makes training difficult
Explaoding gradients
Y
X
Slope grows exponentially

Q: What Is the Difference Between Epoch, Batch, and Iteration in Deep Learning?

A

- ✓ Epoch - Represents one iteration over the entire dataset (everything put into the training model).
- ✓ Batch - Refers to when we cannot pass the entire dataset into the neural network at once, so we divide the dataset into several batches.
- ✓ Iteration - if we have 10,000 images as data and a batch size of 200. then an epoch should run 50 iterations (10,000 divided by 50).

Q: Why Is Tensorflow the Most Preferred Library in Deep Learning?

A: Tensorflow provides both C++ and Python APIs, making it easier to work on and has a faster compilation time compared to other Deep Learning libraries like Keras and Torch. Tensorflow supports both CPU and GPU computing devices.

Q: What Do You Mean by Tensor in Tensorflow?

A: A tensor is a mathematical object represented as arrays of higher dimensions. These arrays of data with different dimensions and ranks fed as input to the neural network are called "Tensors."

Q: What Are the Programming Elements in Tensorflow?

A: Constants - Constants are parameters whose value does not change. To define a constant we use tf.constant() command. For example:

```
a = tf.constant(2.0,tf.float32)
b = tf.constant(3.0)
Print(a, b)
```

Variables - Variables allow us to add new trainable parameters to graph. To define a variable, we use the tf.Variable() command and initialize them before running the graph in a session. An example:

```
W = tf.Variable([.3].dtype=tf.float32)
b = tf.Variable([-.3].dtype=tf.float32)
```

Placeholders - these allow us to feed data to a tensorflow model from outside a model. It permits a value to be assigned later. To define a placeholder, we use the tf.placeholder() command. An example:

```
a = tf.placeholder (tf.float32)
b = a*2
with tf.Session() as sess:
result = sess.run(b,feed_dict={a:3.0})
print result
```

Sessions - a session is run to evaluate the nodes. This is called the “Tensorflow runtime.” For example:

```
a = tf.constant(2.0)
b = tf.constant(4.0)
c = a+b
# Launch Session
Sess = tf.Session()
# Evaluate the tensor c
print(sess.run(c))
```

Q: Explain a Computational Graph

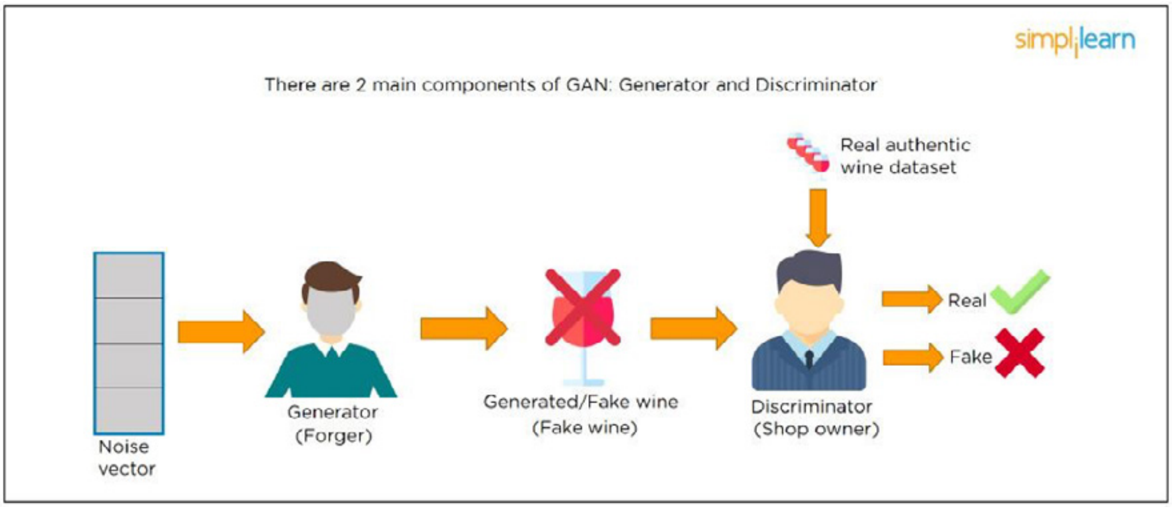
A: Everything in a tensorflow is based on creating a computational graph. It has a network of nodes where each node operates, Nodes represent mathematical operations, and edges represent tensors. Since data flows in the form of a graph, it is also called a “DataFlow Graph.”

Q: Explain Generative Adversarial Network.

A: Suppose there is a wine shop purchasing wine from dealers, which they resell later. But some dealers sell fake wine. In this case, the shop owner should be able to distinguish between fake and authentic wine.

The forger will try different techniques to sell fake wine and make sure specific techniques go past the shop owner's check. The shop owner would probably get some feedback from wine experts that some of the wine is not original. The owner would have to improve how he determines whether a wine is fake or authentic.

The forger's goal is to create wines that are indistinguishable from the authentic ones while the shop owner intends to tell if the wine is real or not accurately.


simplilearn
There are 2 main components of GAN: Generator and Discriminator
Real authentic wine dataset
Noise vector
Generator (Forger)
Generated/Fake wine (Fake wine)
Discriminator (Shop owner)
Real
Fake

Let us understand this example with the help of an image shown above.

There is a noise vector coming into the forger who is generating fake wine.

Here the forger acts as a Generator.

The shop owner acts as a Discriminator.

The Discriminator gets two inputs; one is the fake wine, while the other is the real authentic wine. The shop owner has to figure out whether it is real or fake.

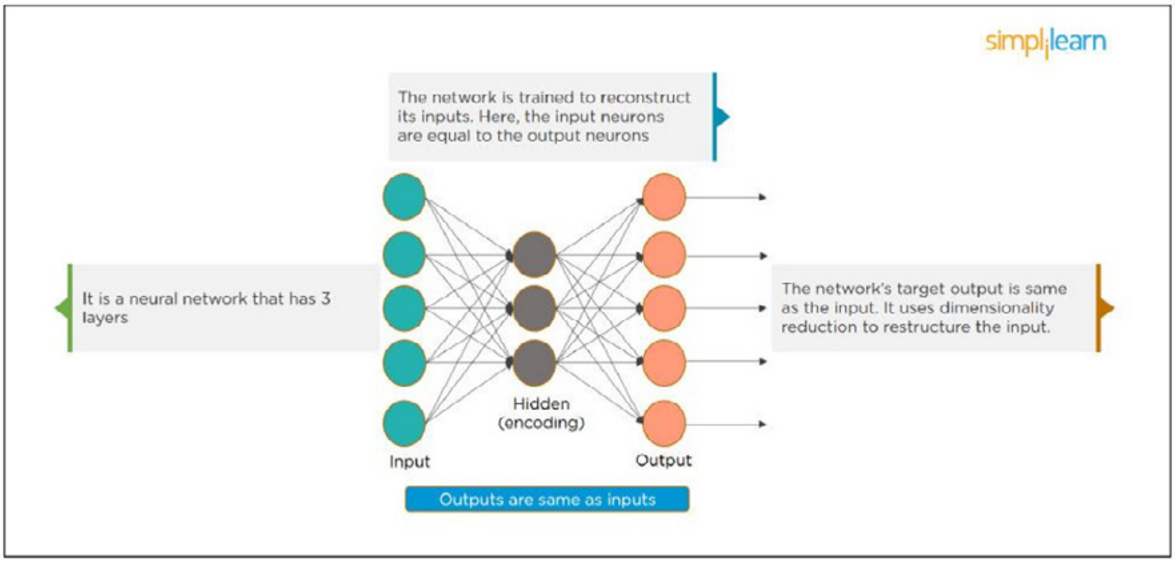
So, there are two primary components of Generative Adversarial Network (GAN) named:

1. Generator
2. Discriminator

The generator is a CNN that keeps keys producing images and is closer in appearance to the real images while the discriminator tries to determine the difference between real and fake images The ultimate aim is to make the discriminator learn to identify real and fake images.

Q: What Is an Auto-encoder?

A:

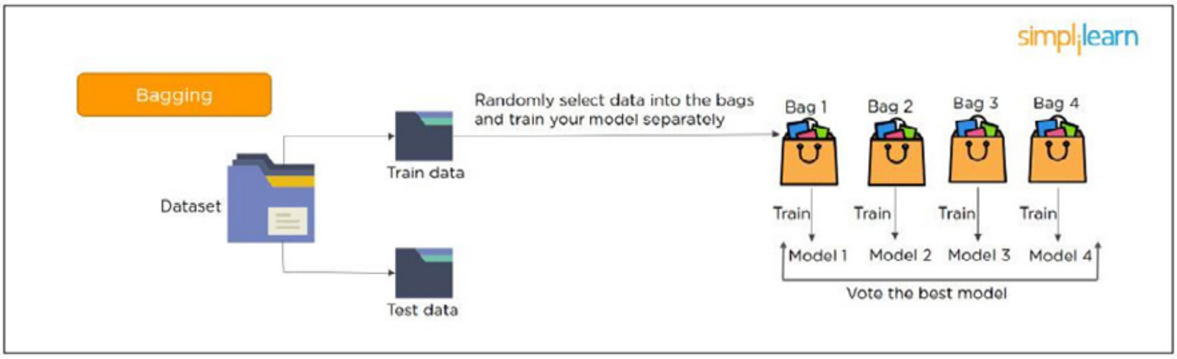


simplilearn
The network is trained to reconstruct its inputs. Here, the input neurons are equal to the output neurons
It is a neural network that has 3 layers
The network's target output is same as the input. It uses dimensionality reduction to restructure the input.
Hidden (encoding)
Input
Output
Outputs are same as inputs

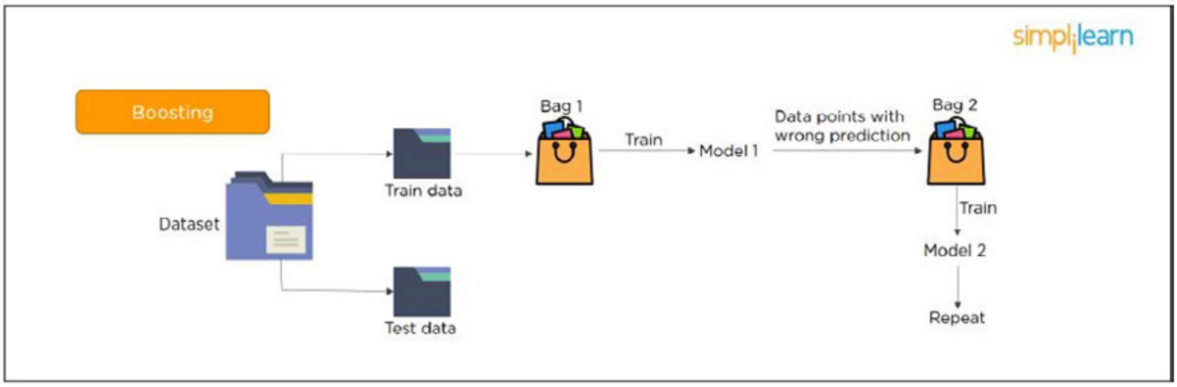
This Neural Network has three layers in which the input neurons are equal to the output neurons. The network's target outside is the same as the input. It uses dimensionality reduction to restructure the input. It works by compressing the image input to a latent space representation then reconstructing the output from this representation

Q: What Is Bagging and Boosting?

A: Bagging and Boosting are ensemble techniques to train multiple models using the same learning algorithm and then taking a call.



With Bagging, we take a dataset and split it into training data and test data. Then we randomly select data to place into the bags and train the model separately.



With Boosting, the emphasis is on selecting data points which give wrong output to improve the accuracy.

Your Next Step to Success

So we have covered several deep learning interview questions that will help you land the perfect job that you always desired. Do you feel unprepared regarding the concepts covered in these interview questions? We are here to help you out. We offer **Deep Learning with the TensorFlow Certification course** that will assist you in gaining expertise in all the concepts of Deep Learning. In this Deep Learning with Keras and TensorFlow course, you will become familiar with the language and fundamental concepts of artificial neural networks, PyTorch, autoencoders, and more. Upon completion, you will be able to build deep learning models, interpret results and build your own deep learning project.

simplilearn

INDIA

Simplilearn Solutions Pvt Ltd.
# 53/1 C, Manoj Arcade, 24th Main,
Harlkunte, 2nd Sector, HSR Layout
Bangalore: 560102
Call us at: 1800-212-7688

USA

Simplilearn Americas, Inc.
201 Spear Street, Suite 1100, San Francisco, CA 94105
United States
Phone No: +1-844-532-7688

www.simplilearn.com