Interview Questions

Q1. What is the difference between AI, Data Science, ML, and DL?

Artificial Intelligence: AI is purely math and scientific exercise, but when it became computational, it started to solve human problems formalized into a subset of computer science. Artificial intelligence has changed the original computational statistics paradigm to the modern idea that machines could mimic actual human capabilities, such as decision making and performing more “human” tasks. Modern AI into two categories -
1. General AI - Planning, decision making, identifying objects, recognizing sounds, social &
business transactions
2. Applied AI - driverless/ Autonomous car or machine smartly trade stocks

Machine Learning: Instead of engineers “teaching” or programming computers to have what they need to carry out tasks, that perhaps computers could teach themselves – learn something without being explicitly programmed to do so. ML is a form of AI based on more data, and they can change actions and response, which will make it more efficient, adaptable and scalable. e.g., navigation apps and recommendation engines.
Classified into:-
1. Supervised
2. Unsupervised
3. Reinforcement learning

Data Science: Data science has many tools, techniques, and algorithms called from these fields, plus others –to handle big data. The goal of data science, somewhat similar to machine learning, is to make accurate predictions and to automate and perform transactions in real-time, such as purchasing internet traffic or automatically generating content.
Data science relies less on math and coding and more on data and building new systems to process the data. Relying on the fields of data integration, distributed architecture, automated machine learning, data visualization, data engineering, and automated data-driven decisions, data science can cover an entire
spectrum of data processing, not only the algorithms or statistics related to data.

Deep Learning: It is a technique for implementing ML.
ML provides the desired output from a given input, but DL reads the input and applies it to other data. In ML, we can easily classify the flower based upon the features. Suppose

you want a machine to look at an image and determine what it represents to the human eye, whether a face, flower, landscape, truck, building, etc.

Machine learning is not sufficient for this task because machine learning can only produce an output from a data set – whether according to a known algorithm or based on the inherent structure of the data. You might be able to use machine learning to determine whether an image was of an “X” – a flower, say – and it would learn and get more accurate. But that output is binary (yes/no) and is dependent on the algorithm, not the data. In the image recognition case, the outcome is not binary and not dependent on the algorithm.
The neural network performs MICRO calculations with computation on many layers. Neural networks also support weighting data for ‘confidence. These results in a probabilistic system, vs. deterministic, and can handle tasks that we think of as requiring more ‘human-like’ judgment.

Q2. What are Epochs?

One Epoch is an ENTIRE dataset that is passed forwards and backwards through the neural network. Since one epoch is too large to feed to the computer at once, we divide it into several smaller batches. We always use more than one Epoch because one epoch leads to underfitting. As the number of epochs increases, several times the weight is changed in the neural network and the curve goes from underfitting up to optimal to overfitting curve.

Q3. What is the batch size?

The total number of training and examples present in a single batch.
Unlike the learning rate hyperparameter where its value doesn’t affect computational time, the batch sizes must be examined in conjunctions with the execution time of training. The batch size is limited by hardware’s memory, while the learning rate is not. Leslie recommends using a batch size that fits in hardware’s memory and enables using a larger learning rate.
If our server has multiple GPUs, the total batch size is the batch size on a GPU multiplied by the numbers of GPU. If the architectures are small or your hardware permits very large batch sizes, then you might compare the performance of different batch sizes. Also, recall that small batch sizes add regularization while large batch sizes add less, so utilize this while balancing the proper amount of regularization. It's often better to use large batch sizes so a larger learning rate can be used.

Q4. What Is Data Normalization, and Why Do We Need It?

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.

Q5. What Is Dropout and Batch Normalization?

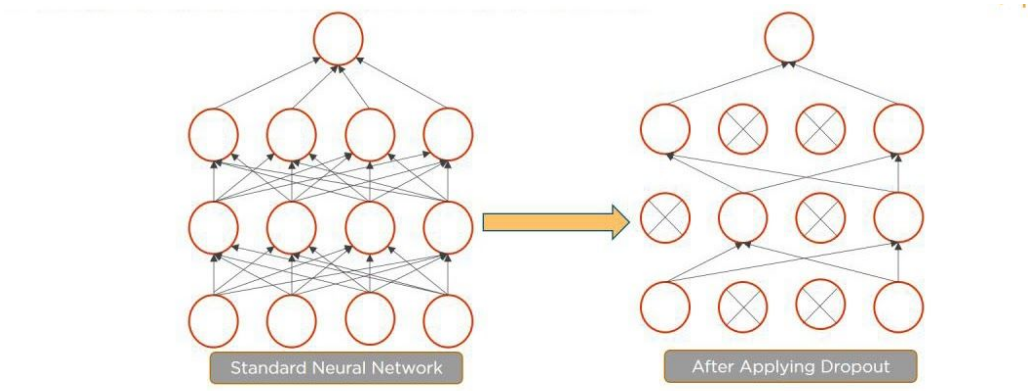
Dropout is a technique of dropping out hidden and visible units of a network randomly to prevent overfitting of data (typically dropping 20 percent of the nodes). It doubles the number of iterations needed to converge the network.

Where to use -

- Dropout is implemented per-layer in a neural network.
- It can be used with most types of layers, such as dense fully connected layers, convolutional layers, and recurrent layers such as the long short-term memory network layer.
- Dropout may be implemented on any or all hidden layers in the network as well as the visible or input layer. It is not used on the output layer.

Benefits -

1. Dropout forces a neural network to learn more robust features that are very useful in conjunction with different random subsets of the other neurons.
2. Dropout generally doubles the number of iterations required to converge. However, the training time for each epoch is less.



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.

Q6. Explain hyperparameter tuning in deep learning.

The process of setting the hyper-parameters requires expertise and extensive trial and error. There are no simple and easy ways to set hyper-parameters — specifically, learning rate, batch size, momentum, and weight decay.

Approaches to searching for the best configuration:
• Grid Search
• Random Search

Approach
1. Observe and understand the clues available during training by monitoring validation/test loss early in training, tune your architecture and hyper-parameters with short runs of a few epochs.

2. Signs of underfitting or overfitting of the test or validation loss early in the training process are useful for tuning the hyper-parameters.

Tools for Optimizing Hyperparameters
• Sage Maker
• Comet.ml
• Weights &Biases
• Deep Cognition
• Azure ML

Q7. What Is the Role of Activation Functions in a Neural Network?

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 function, Sigmoid, ReLU, Tanh, and Softmax are examples of activation functions.

A neural network without activation functions is essentially a linear regression model. The activation functions do the non-linear transformation to the input, making it capable of learning and performing more complex tasks.

1. Identity
2. Binary Step
3. Sigmoid
4. Tanh
5. ReLU
6. Leaky ReLU
7. Softmax

The activation function do the non-linear transformation to the input, making it capable of learning and performing more complex tasks.

Q8. What do you understand by Error functions?

In most learning networks, an error is calculated as the difference between the predicted output and the actual output.

The function that is used to compute this error is known as Loss Function J(.). Different loss functions will give different errors for the same prediction, and thus have a considerable effect on the performance of the model. One of the most widely used loss function is mean square error, which calculates the square of the difference between the actual values and predicted-value. Different loss functions are used to deal with a different type of tasks, i.e. regression and classification.

Regressive loss functions:

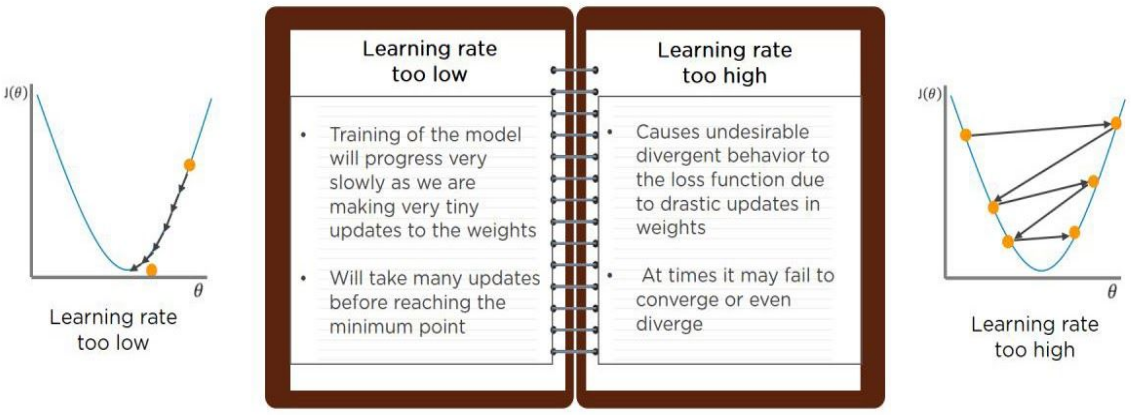
- Mean Square Error
- Absolute error

- Smooth Absolute Error

Q9. What Will Happen If the Learning Rate Is Set Too Low or Too High?

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).



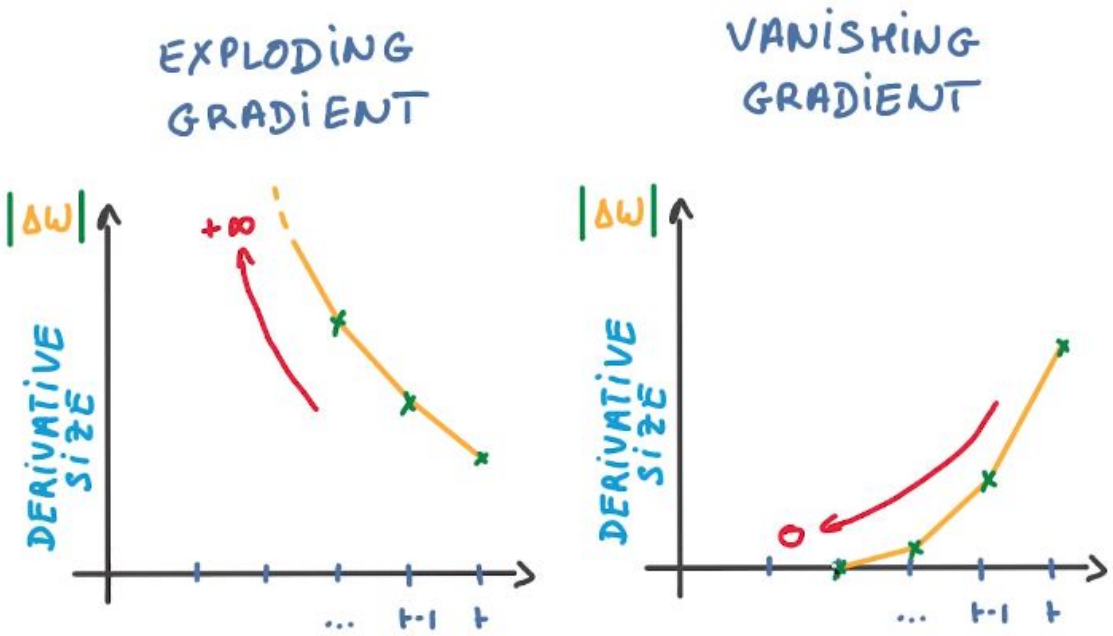
Q10. How Are Weights Initialized in a Network?

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.

Q11. What Are Vanishing and Exploding Gradients?

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.



Q12. Why is Tensorflow the Most Preferred Library in Deep Learning?

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.

Q13. Explain a Computational Graph.

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.”

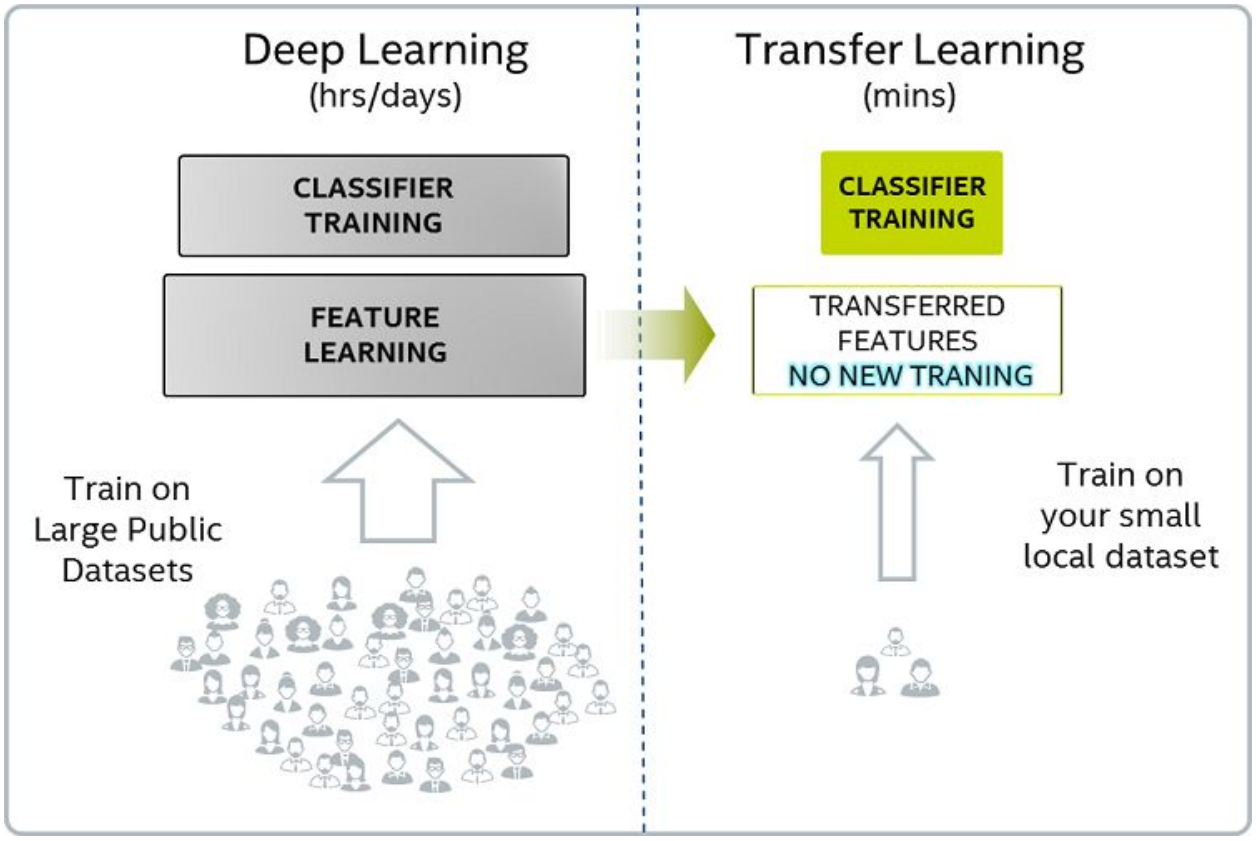
Q14. What is Transfer learning in deep learning ?

Transfer learning: It is a machine learning method where a model is developed for the task and is again used 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.

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task.

Transfer learning is an optimization that allows rapid progress or improved performance when modellingthe second task. Transfer learning only works in deep learning if the model features learned from the first task are general.


Deep Learning
(hrs/days)
CLASSIFIER TRAINING
FEATURE LEARNING
Train on Large Public Datasets
Transfer Learning
(mins)
CLASSIFIER TRAINING
TRANSFERRED FEATURES
NO NEW TRANING
Train on your small local dataset