1.Explain the Activation Functions in your own language:

Ans.

a) Sigmoid: Sigmoid function maps any input value to a value between 0 and 1. It is useful in binary classification tasks where we need to output probabilities of an input belonging to a particular class.

b) Tanh: Tanh function is similar to Sigmoid function but maps input values to a range between -1 and 1. It is useful in some regression tasks.

c) ReLU: ReLU stands for Rectified Linear Unit. It outputs the input value if it's positive, otherwise outputs 0. It is the most commonly used activation function and is useful in deep neural networks.

d) ELU: ELU stands for Exponential Linear Unit. It is similar to ReLU, but it outputs a negative value for negative input values. It has been shown to perform better than ReLU in some scenarios.

e) LeakyReLU: Leaky ReLU is similar to ReLU, but instead of outputting 0 for negative input values, it outputs a small negative value. It can help prevent the dying ReLU problem.

f) Swish: Swish function is a recently proposed activation function that performs better than ReLU in some scenarios. It is a smooth function that applies a sigmoid-like function to the input value.

2.What happens when you increase or decrease the optimizer learning rate?

Ans.

When you increase the optimizer learning rate, the model can converge faster as the weights are updated more aggressively. However, if the learning rate is too high, the model may overshoot the optimal solution and fail to converge. Conversely, if you decrease the learning rate, the model will update the weights more slowly, which may result in slower convergence. But it can help to find the optimal solution.

3.What happens when you increase the number of internal hidden neurons?

Ans.

Increasing the number of internal hidden neurons can increase the model's capacity to learn complex features. However, it can also increase the risk of overfitting if the number of neurons is too high. Furthermore, increasing the number of neurons can also lead to increased computational cost and slower training times.

4.What happens when you increase the size of batch computation?

Ans.

Increasing the size of batch computation can speed up the training process, as more training examples are processed in parallel. However, it can also result in less accurate weight updates, as the gradient estimation is based on a smaller subset of the data. Additionally, increasing batch size can also increase memory requirements, which may become a bottleneck for large models or limited hardware resources.

5.Why do we adopt regularization to avoid overfitting?

Ans.

Regularization is used to prevent overfitting, which occurs when a model is too complex and fits the training data too closely. Regularization techniques such as L1, L2 regularization, Dropout, and Early Stopping help to prevent overfitting by imposing constraints on the model's complexity, reducing the variance of the model, or stopping training early.

6.What are loss and cost functions in deep learning?

Ans.

Loss and cost functions are used to measure the error between the predicted output of a model and the actual output. Loss function measures the error for a single training example, while the cost function is the average of the loss functions across the entire training dataset. The objective of deep learning is to minimize the cost function, i.e., to find the set of model parameters that minimize the error between predicted and actual outputs.

7.What do you mean by underfitting in neural networks?

Ans.

Underfitting occurs when a model is too simple to capture the underlying patterns in the data, resulting in poor performance on both training and testing data. It can occur when the model's capacity is too low or when the training dataset is too small. Signs of underfitting include high training error and high testing error.

8.Why do we use Dropout in Neural Networks?

Ans.

Dropout is a regularization technique used to prevent overfitting in neural networks. It randomly drops out some neurons during training, which helps to prevent the network from relying too heavily on any one neuron and thus reduces the risk of overfitting. Dropout has been shown to be an effective technique in improving the generalization performance of neural networks.