1. Explain the Activation Functions in your own language

a) sigmoid

b) tanh

c) ReLU

d) ELU

e) LeakyReLU

f) swish

Ans: a) Sigmoid: The sigmoid activation function is a type of mathematical function used in artificial neural networks that maps any real number to a value between 0 and 1. This allows the model to make predictions based on the input data.

b) Tanh: The tanh activation function is similar to the sigmoid activation function, but it maps any real number to a value between -1 and 1. This allows the model to make more accurate predictions compared to the sigmoid activation function.

c) ReLU: The ReLU (Rectified Linear Unit) activation function is a type of activation function that takes any real number and outputs 0 if the number is less than 0, and the same number if it is greater than or equal to zero. This allows the model to learn from the data and make more accurate predictions.

d) ELU: The ELU (Exponential Linear Unit) activation function is a type of activation function that takes any real number and outputs a negative number if the number is less than 0, and the same number if it is greater than or equal to zero. This allows the model to learn from the data and make more accurate predictions.

e) LeakyReLU: The LeakyReLU activation function is similar to the ReLU activation function, but it outputs a small negative number if the input is less than 0. This helps to reduce the "dying ReLU" problem, where the activation function outputs 0 for all inputs.

f) Swish: The swish activation function is a type of activation function that takes any real number and outputs a value in the range of -1 to 1. This allows the model to learn from the data and make more accurate predictions.

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

Ans: When you increase the optimizer learning rate, the optimizer is able to make larger jumps in parameter space and is more likely to find a better local minimum in a shorter amount of time. Conversely, when you decrease the optimizer learning rate, the optimizer is less likely to reach a local minimum and may take more time to find a suitable solution.

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

Ans: Increasing the number of internal hidden neurons can help the model to better capture complex patterns in the data, leading to better performance. However, too many hidden neurons can lead to overfitting, where the model is too tailored to the training data and fails to generalize to unseen data.

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

Ans: Increasing the size of batch computation can lead to faster and more efficient training, as larger batches can be processed in parallel and allow for more efficient use of hardware resources. However, too large of a batch size can lead to longer training times and can lead to reduced accuracy.

5. Why we adopt regularization to avoid overfitting?

Ans: Regularization is used to avoid overfitting by adding constraints to the model parameters, such as weight decay or dropout. This forces the model to learn more generalizable patterns in the data, rather than fitting to the noise or outliers in the training data. Regularization also helps to prevent the model from overfitting by encouraging simpler models.

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

Ans: Loss and cost functions are used in deep learning to measure the performance of the model. Loss functions measure the difference between the predictions of the model and the ground truth labels, while cost functions are measures of how well the model is performing. Cost functions are typically a combination of the loss function and the regularization component.

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

Ans: Underfitting occurs when a neural network model fails to capture the underlying relationships in the data, leading to low accuracy on both the training data and the unseen test data. Underfitting can be caused by not having enough neurons and layers in the model, or by not training the model for long enough.

8. Why we use Dropout in Neural Networks?

Ans: Dropout is a regularization technique used in neural networks to prevent overfitting. It randomly removes neurons from the network during training, forcing the network to learn more robust representations of the data. This helps the model to generalize better to unseen data.