1. After each stride-2 conv, why do we double the number of filters?

Ans:

The number of filters is doubled after each stride-2 convolutional layer to increase the capacity of the model to learn more complex features. As the spatial resolution of the feature maps is reduced by half after each stride-2 convolution, increasing the number of filters compensates for the loss of spatial information and allows the network to capture more fine-grained details

1. Why do we use a larger kernel with MNIST (with simple cnn) in the first conv?

Ans:

The MNIST dataset consists of 28x28 grayscale images of handwritten digits, which are relatively small compared to many other image datasets. Using a larger kernel in the first convolutional layer of a convolutional neural network (CNN) can help capture more information from the input images and learn more complex features.

3. What data is saved by ActivationStats for each layer?

1. Activation mean: the average value of the activations for each channel.
2. Activation standard deviation: the standard deviation of the activations for each channel.
3. Sparsity: the proportion of activations that are zero.
4. L0 norm: the number of non-zero activations.
5. Min activation: the minimum value of the activations for each channel.
6. Max activation: the maximum value of the activations for each channel.

4. How do we get a learner's callback after they've completed training?

To get a learner's callback after they've completed training, you can use the **add\_cbs** method of the learner object. The **add\_cbs** method allows you to add a list of callbacks to the learner object, which will be called at specific points during training.

To get a callback that will be called after training is complete, you can use the **AfterFit** callback. This callback will be called after the final epoch of training, and will have access to the final model and the final validation loss.

5. What are the drawbacks of activations above zero?

when activations get too large, they can cause the gradients to become very large during backpropagation. This can lead to a phenomenon known as the "exploding gradient problem," which can make training unstable and prevent the model from converging.

6.Draw up the benefits and drawbacks of practicing in larger batches?

Ans: large batches converge to global minima faster but they need more memory or computational power.

1. Why should we avoid starting training with a high learning rate?

Ans: High Learning rate can make the convergence harder to reach as it will take bigger steps to move from one point to the other.

1. What are the pros of studying with a high rate of learning?

Ans: A high learning rate can help the model escape from local minima because it allows the model to jump over small valleys in the loss landscape that may trap the model with a lower learning rate.

1. Why do we want to end the training with a low learning rate?

Ans:

Initially, during the early stages of training, a large learning rate is usually used to quickly converge to a good solution. However, as the optimization process continues, the model parameters get closer and closer to the optimal values, and smaller updates are needed to further minimize the loss.