**Assignment - 03**

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

Ans: Doubling the number of filters after each stride-2 conv:

Doubling the number of filters after each stride-2 convolution helps increase the network's capacity to learn complex features and patterns.

Stride-2 convolutions reduce the spatial dimensions of the feature maps while increasing their depth. Increasing the number of filters compensates for this reduction in spatial resolution, allowing the network to capture more diverse and detailed features.

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

Ans: Using a larger kernel with MNIST in the first conv:

MNIST images are relatively small (28x28 pixels), and using a larger kernel in the first convolution helps capture larger spatial features and patterns.

Larger kernels have a wider receptive field, allowing the network to extract more global features from the input images.

This can improve the network's ability to learn discriminative features from the limited input resolution of MNIST images.

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

Ans: Data saved by ActivationStats for each layer:

ActivationStats saves statistical information about the activations (output values) of each layer during training.

This includes mean, standard deviation, maximum, minimum, and histogram of activations.

Monitoring these statistics can help diagnose issues such as vanishing or exploding gradients, saturation of activation functions, and model convergence problems.

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

Ans: Getting a learner's callback after completing training:

To get a learner's callback after completing training, you can use callbacks provided by deep learning frameworks such as PyTorch or TensorFlow.

For example, in PyTorch, you can define a custom callback function and register it with the on\_train\_end event of the training loop.

5. What are the drawbacks of activations above zero?

Ans: Drawbacks of activations above zero:

Activations above zero may lead to exploding gradients during training, causing instability and difficulty in optimizing the model.

They can also lead to saturation of activation functions, where the gradients become close to zero, hindering learning.

In some cases, overly large activations may indicate model overfitting to the training data, resulting in poor generalization performance on unseen data.

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

Ans: Benefits and drawbacks of practicing in larger batches:

Benefits:

Faster training times, as larger batches utilize parallel processing more efficiently.

Smoother convergence of training due to more stable gradients.

Can lead to better generalization performance in some cases.

Drawbacks:

Increased memory requirements, limiting the batch size that can fit into GPU memory.

May lead to poorer generalization performance, especially if batch size is too large and results in overfitting to the training data.

Larger batches may suffer from decreased model diversity, potentially hindering exploration of the parameter space.

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

Ans: Avoiding starting training with a high learning rate:

Starting training with a high learning rate can lead to unstable training dynamics, causing the model to overshoot optimal parameter values and converge poorly.

High learning rates may result in large updates to model parameters, leading to oscillations or divergence in the loss landscape.

Gradually increasing the learning rate during training allows the model to explore the parameter space more effectively while avoiding these issues.

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

Ans: Pros of studying with a high learning rate:

High learning rates can lead to faster convergence and shorter training times, especially in the initial stages of training.

They allow the model to escape from local minima and saddle points more easily, facilitating exploration of the parameter space.

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

Ans: Ending training with a low learning rate:

Ending training with a low learning rate helps stabilize the training process and fine-tune the model parameters near convergence.

Lower learning rates enable more precise adjustments to the model parameters, allowing the model to converge to a more optimal solution.

This approach helps improve the model's generalization performance and ensures that the training process does not overshoot or oscillate around the optimal solution.