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

Doubling the number of filters after each stride-2 convolutional operation is a common practice in certain network architectures, such as the VGG (Visual Geometry Group) network. This design choice aims to increase the capacity and expressive power of the network while controlling the spatial dimensions of the feature maps

The doubling of filters serves two primary purposes:

Hierarchical feature extraction:

Control of spatial dimension

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

Using a larger kernel in the first convolutional layer of a simple CNN architecture for the MNIST dataset can help capture more global features and provide a broader receptive field. This choice allows the network to learn higher-level representations of the input data.

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

The specific data saved by ActivationStats may vary depending on its implementation or the purpose for which it was designed. However, in general, ActivationStats is often used to collect and analyze statistical information about the activations or outputs of each layer in a neural network during training or inference. It includes.

Activation statistics

Activation histograms

Activation gradients

Layer-wise statistics

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

Define a custom callback class: Create a custom class that inherits from the framework's callback base class. For PyTorch, you can inherit from the torch.nn.Module class and override specific methods to define the desired behaviour after training completion.

Instantiate the callback class: Create an instance of the custom callback class.

Register the callback: Register the callback instance with the learner or trainer object to ensure it is called after training completion. The exact method for registering callbacks may vary depending on the framework and specific training setup. In PyTorch, you can use the register\_backward\_hook() method during training.

1. What are the drawbacks of activations above zero?

Activations above zero (positive activations) are generally desirable in neural networks as they represent the presence and strength of certain features or patterns. However, it's important to note that there are situations where activations above zero can have certain drawbacks or limitations.

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

Drawbacks of larger batches:

1. Higher memory requirements: Larger batches require more memory to store the activations and gradients during the forward and backward passes.
2. Slower updates: With larger batches, parameter updates occur less frequently, as the gradients are averaged over more samples.
3. Potential for suboptimal local minima: It has been observed that larger batch sizes can sometimes result in convergence to suboptimal local minima.

Benefits of larger batches:

1. Improved parallelism: Larger batches allow for better utilization of parallel processing capabilities of modern hardware, such as GPUs.
2. Smoother convergence: Larger batches can provide a smoother gradient estimation compared to smaller batches.
3. Increased memory efficiency: Using larger batches can be more memory-efficient, as it reduces the frequency of memory transfers between the CPU and GPU.
4. Why should we avoid starting training with a high learning rate?

Starting training with a high learning rate can lead to several issues and hinder the effectiveness of the training process in deep learning

For example:

Unstable convergence

Overshooting the minimum

Difficulty in finding a good solution

Instability in gradients

Difficulty in fine-tuning

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

Studying with a high learning rate, or adopting an intense and fast-paced learning approach, can offer certain advantages and benefits in certain scenarios like Rapid knowledge acquisition, Increased motivation and engagement, Efficient use of time, Improved retention and recall and Preparation for time-sensitive scenarios etc.

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

Ending the training process with a low learning rate, often referred to as learning rate decay or annealing, can provide several benefits in the training of deep learning models.

Here are the reasons why it is advantageous to conclude training with a low learning rate:

Enhanced fine-tuning

Stable convergence

Enhanced generalization

Escaping local minima

Smoother parameter updates