1. Explain convolutional neural network, and how does it work?

A convolutional neural network (CNN) is a type of deep learning model specifically designed for analysing visual data, such as images and videos. CNNs are widely used in various computer vision tasks, including image classification, object detection, and image segmentation.

1. Input layer
2. Convolutional layers
3. Activation function
4. Pooling layers
5. Fully connected layers
6. Output layer
7. Training and optimization

CNNs excel at automatically extracting relevant features from visual data, reducing the need for manual feature engineering. Their ability to learn hierarchical representations, exploit local relationships, and handle translation equivariance make them highly effective for a wide range of computer vision tasks.

1. How does refactoring parts of your neural network definition favour you?

Refactoring parts of a neural network definition can provide several benefits that ultimately favour the model development process. Here are some advantages of refactoring a neural network:

1. Modularity and code organization:
2. Reusability and extensibility
3. Readability and comprehension
4. Debugging and error isolation
5. Performance optimization
6. Maintainability and future updates

Overall, refactoring parts of a neural network definition promotes modularity, reusability, readability, debugging, performance optimization, and maintainability. It enhances the development process, making it more efficient, scalable, and adaptable to future needs and updates.

3. What does it mean to flatten? Is it necessary to include it in the MNIST CNN? What is the reason for this?

Flattening refers to the process of reshaping a multidimensional tensor into a one-dimensional vector. It transforms the spatial dimensions of the input data into a single dimension while preserving the information contained within.

In the case of the MNIST dataset, which consists of grayscale images of handwritten digits, each image is initially represented as a 2D matrix, typically of shape (height, width). However, before passing the image through fully connected layers in a CNN, it is necessary to flatten the image into a 1D vector.

In nutshell, flattening is necessary in the MNIST CNN architecture to convert the 2D image data into a 1D vector format compatible with the fully connected layers. It allows the network to treat each pixel as a separate input feature and learn the correlations and patterns in the pixel values, eventually leading to accurate digit classification.

1. What exactly does NCHW stand for?  
   NCHW stands for "Number of samples, Channels, Height, Width." It is a commonly used format for representing the dimensions or shape of tensors in deep learning frameworks, particularly in the context of convolutional neural networks (CNNs).
2. Why are there 7\*7\*(1168-16) multiplications in the MNIST CNN's third layer?

The number of multiplications in a CNN layer depends on the dimensions of the input, the dimensions of the convolutional kernel, and the number of output channels or feature maps. In a typical CNN layer, the number of multiplications can be calculated as follows:

Number of multiplications = (kernel\_height \* kernel\_width \* input\_channels) \* output\_channels

6. Explain definition of receptive field?

The receptive field in the context of neural networks, particularly convolutional neural networks (CNNs), refers to the region of the input data that a particular neuron or feature in a layer can "see" or influence. It represents the spatial extent of the input that contributes to the computation of a neuron's output.

7. What is the scale of an activation's receptive field after two stride-2 convolutions? What is the reason for this?

After two stride-2 convolutions, the scale or size of an activation's receptive field increases. The receptive field size is influenced by the stride value, which determines the step size of the convolutional kernel as it moves across the input data.

When a stride of 2 is applied during convolution, the kernel shifts by two steps horizontally and vertically after each operation. This leads to a down sampling effect on the input data, reducing its spatial dimensions. Consequently, the receptive field size increases relative to the original input.

1. What is the tensor representation of a colour image?

The tensor representation of a colour image typically follows the NCHW (Number of samples, Channels, Height, Width) format commonly used in deep learning frameworks like PyTorch and Tensor Flow. In this format, a colour image is represented as a 4-dimensional tensor with the following dimensions:

1. How does a colour input interact with a convolution?

When a colour input (e.g., RGB image) interacts with a convolutional layer in a convolutional neural network (CNN), the convolution operation is applied to each colour channel separately. The interaction occurs through the convolutional kernel, which is a set of learnable filters.