**CV Assignment 2**

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

Ans. It *is a type of deep learning model specifically designed to process and analyze visual data, such as images or videos. CNNs are widely used in computer vision tasks, including image classification, object detection, segmentation, and more.*

*Here's a high-level explanation of how a CNN works:*

* *Convolutional Layers: CNNs start with one or more convolutional layers. Each layer consists of multiple filters or kernels that slide over the input image in a convolution operation. The filters perform element-wise multiplication and summation with the input image pixels, producing feature maps that capture local patterns and features.*
* *Non-Linear Activation: After the convolution operation, a non-linear activation function (e.g., ReLU - Rectified Linear Unit) is applied element-wise to introduce non-linearity and enable the network to learn complex representations.*
* *Pooling Layers: Pooling layers are used to downsample the feature maps and reduce their spatial dimensions. Common pooling methods include max pooling or average pooling, which aggregate the information from local regions, reducing computational complexity and providing translation invariance.*
* *Additional Layers: CNNs often have additional layers like fully connected layers, also known as dense layers or fully connected layers. These layers connect every neuron from the previous layer to the current layer, allowing the network to learn high-level representations and make final predictions.*
* *Output Layer: The final layer of a CNN is usually a softmax layer for classification tasks, which provides the probability distribution over different classes. In other tasks like object detection or segmentation, the output layer may have a different configuration depending on the specific problem.*
* *Training: CNNs are trained using a large dataset with labeled examples. During training, the network adjusts the weights and biases of its layers through a process called backpropagation. This process minimizes the difference between predicted outputs and the ground truth labels by optimizing a loss function, typically cross-entropy loss.*

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

Ans. *Here are some ways in which refactoring can be advantageous:*

* *Code Organization and Readability*
* *Modularity and Reusability*
* *Scalability and Flexibility*
* *Performance Optimization*
* *Debugging and Troubleshooting*
* *Maintainability and Extensibility*

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

Ans. *Flattening refers to the process of reshaping a multidimensional tensor into a one-dimensional vector. It involves collapsing all dimensions except the first one into a single dimension. The resulting flattened vector can then be connected to fully connected layers in the network.*

*For example, consider an input tensor with dimensions [batch\_size, channels, height, width]. Flattening this tensor would convert it into a vector with dimensions [batch\_size, channels \* height \* width], where the spatial dimensions are flattened into a single dimension.*

*In the case of the MNIST dataset, which consists of 28x28 grayscale images, flattening is typically performed before passing the image data into fully connected layers for classification. The reason for flattening in the MNIST CNN is as follows:*

* *Fully Connected Layers: The MNIST CNN often includes fully connected layers at the end of the network. Fully connected layers require inputs in the form of one-dimensional vectors, where each element corresponds to a specific feature or pixel value. Flattening the image data allows us to reshape the 2D image grid into a vector format that can be connected to these fully connected layers.*
* *Preserving Spatial Information: Flattening discards the spatial structure of the input image, converting it into a linear representation. In the case of MNIST, this is acceptable since the spatial relationships between pixels are less relevant for the task of digit classification. The network can learn meaningful features from the flattened vector and make accurate predictions based on the overall distribution of pixel values.*

4. What exactly does NCHW stand for?

Ans. *NCHW stands for "Number of samples, Channels, Height, Width". It is a notation commonly used to describe the dimensions or shape of multidimensional tensors, particularly in the context of deep learning frameworks such as PyTorch and TensorFlow.*

5. Why are there 7\*7\*(1168-16) multiplications in the MNIST CNN's third layer?

Ans. *The output shape is 64x4x14x14, and this will therefore become the input shape to the next layer. The next layer, according to summary, has 296 parameters. Let’s ignore the batch axis to keep things simple. So, for each of 14\*14=196 locations we are multiplying 296-8=288 weights (ignoring the bias for simplicity), so that’s 196\*288=56,448 multiplications at this layer. The next layer will have 7\*7\*(1168-16) = 56,448 multiplications.*

6.Explain definition of receptive field?

Ans. *The receptive field refers to the region in the input space that a particular feature in a network layer can "see" or be influenced by. It represents the spatial extent of the input data that contributes to the activation of a specific neuron in a layer. To understand the receptive field, let's consider a CNN layer that applies convolutional operations. Each neuron in that layer has a corresponding receptive field, which is defined by the size of the convolutional kernel and the spatial arrangement of the layers in the network.*

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

Ans. *When two stride-2 convolutions are applied consecutively, the scale of an activation's receptive field increases. Each stride-2 convolution effectively reduces the spatial dimensions of the activation map by a factor of 2 in both width and height. Consequently, the receptive field expands. To understand why the scale of the receptive field increases, let's consider the effects of stride-2 convolutions:*

* *First Stride-2 Convolution: The first stride-2 convolution reduces the spatial dimensions of the activation map by a factor of 2. This occurs because, with a stride of 2, the convolutional kernel moves two pixels at a time during the convolution operation. As a result, the width and height of the activation map are halved compared to the input.*
* *Second Stride-2 Convolution: The second stride-2 convolution further reduces the spatial dimensions of the activation map by another factor of 2. Similar to the first convolution, the stride of 2 causes the kernel to move two pixels at a time, resulting in another halving of the width and height.*

8. What is the tensor representation of a color image?

Ans. *A color image can be represented as a tensor with three channels: Red, Green, and Blue (RGB). The tensor representation of a color image has a shape of [height, width, 3] or [3, height, width], depending on the convention used by the deep learning framework.*

9. How does a color input interact with a convolution?

Ans. *Here's how the interaction between a color input and a convolution unfolds:*

*Input Channels: A color image consists of three channels: Red, Green, and Blue (RGB). Each channel represents the intensity or color values for that specific color component. When the color image is passed through a convolutional layer, the convolution operation is applied independently to each channel.*

* *Convolution Operation: For each color channel, the convolutional layer applies a convolution operation using a set of learnable filters (kernels). These filters have smaller spatial dimensions compared to the input and are responsible for extracting different features or patterns from the input.*
* *Element-wise Multiplication and Summation: The convolutional layer slides each filter over the input, performing element-wise multiplications between the filter weights and the corresponding input values at each position. These multiplications are then summed to produce a single value, which becomes the output at that position in the feature map.*
* *Multiple Feature Maps: Typically, a convolutional layer has multiple filters, each producing a separate feature map. Each feature map captures different patterns or information from the input. The number of filters in a convolutional layer determines the number of output channels or feature maps generated by that layer.*
* *Bias and Activation Function: After the convolution operation, a bias term may be added to each feature map, and an activation function (such as ReLU) is applied element-wise to introduce non-linearity into the network.*