Case Study X

Krushang Shah

Prof. Richard Khan

Senior Design I

ID 23821856

Convolutional Neural Network

The term Convolutional Neural Network says that it is a Neural Network, that uses Convolution. A Neural Network is a multiple layer of neurons that are connected through different functions. The convolution operation is a linear operation that can be used instead of matrix multiplication in mathematics. A Convolutional Neural Network (CNN/ ConvNet) consists of an input layer, an output layer and several hidden layers that consist of different layers that do different types of operations.

The main advantage of a Convolutional Neural Network is that the pre-processing required for a Convolutional Neural Network is way lesser than that for general classification algorithms. In older and traditional algorithms, filters were designed by humans, however a Convolutional Neural Network has ability of improving the accuracy overtime by learning and improving the filters and characteristics by itself. The different types of hidden layers that are a part of a Convolutional Neural Network are

Convolutional Layer: This layer uses convolution to convert the matrix representation of an input image to a feature map, by multiplying each entry of the matrix with a feature detector, which are also known as filter and filter biases.

Local Connectivity: In a neural network with large amounts of neurons, it is almost impossible to connect each neuron to all the other neurons. Hence, this layer uses local spatial connectivity to establish a pattern between neurons and adjacent layers.

Spatial Arrangement: This layer controls the size of the output volume using three parameters depth, stride and zero-padding. The depth controls the number of neurons, the stride controls the dimensions like height and width, and the zero-padding adds the border of zeros on the input layer.

Parameter Sharing: This layer determines the number of free and shared parameters in a convolutional layer. This layer determines the depth slices in the input, which are slices of neurons in each depth, that have to use common biases and weights.

Pooling Layer: This is another important layer in a Convolutional Neural Network. It applies non-linear function like max pooling to take only the maximum cell from each block of elements. While this does not really make much difference to the result accuracy, it drastically decreases the time and memory footprint.

ReLU Layer: This layer is called Rectified Linear Unit, which applies non-saturating activation function max on the input, which takes maximum between the input value and zero, essentially changing all the negative values to zero.

Fully Connected Layer: This layer is used after multiple layers of convolution and pooling. This layer is responsible for the high-level reasoning of a neural network. It is a fully connected network, where each neuron has connections to all the activation functions in the previous layer.

Loss Layer: This layer determines the loss that occurred throughout the training of the network, and penalizes the network based on the predicted and true labels by comparing them.

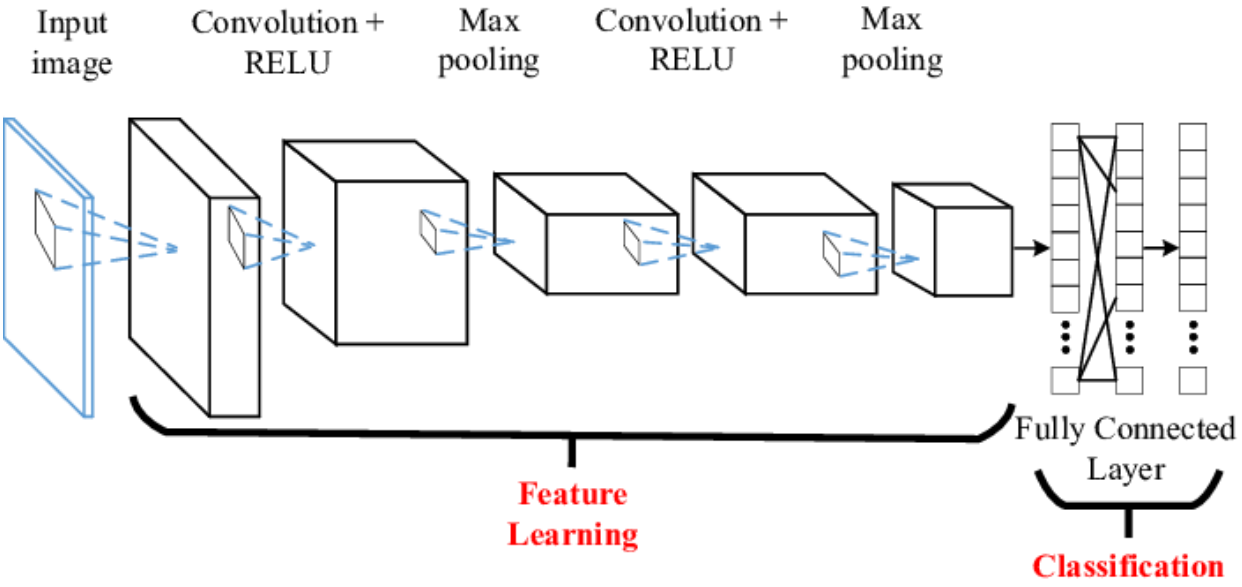

Figure 1: General example of a Convolutional Neural Network

The above shown image is a general example of a convolutional neural network, which is generally made up of two parts feature learning and classification. The feature learning part is used to increase the learning and do the operations on the input, which includes layers like convolutional, pooling, ReLU, Parameter sharing and local connectivity. The classification part is used to determine the class of the input after all the processes, which include layers like fully connected layer and loss layer. The above image has the first layer of convolution, which separates the image into pixels, applies convolution on it, which increases the dimensions of the input. The ReLU layer is applied to soften the input data. Then it sends the processed data to the next layer, which is pooling, that is used to reduce the size of the input, based on the type of pooling and the size of the slices. Then we have another set of convolutions + ReLU layer and Pooling layer, which further flattens the

input data. Then the classification part starts, which first has the fully connected layer. This layer determines the class of the processed data, based on the previous learning from the training of the model. Then finally, we get an output as a type of classes based on the described classes. In the example, you can see that the input is an image of boat, which is then processed through the layers, to classify it based on the classes depending on the input types. This explains the working of a general convolutional network, next we will focus on the fully connected layer, to know more about it.

Spatial Arrangement

The spatial arrangement of the Convolutional Neural Network controls the size of the output volume of the Convolutional Layer, which is controlled by three main hyperparameters depth, stride and zero-padding.

Depth

The number of neurons in each layer which ae connected to the same region of the input depends on the depth of the output volume. The neurons get an artificial learning to activate depending on the different features of the input. For example, the input to the first convolutional layer is a raw image. The different neurons get activated based on the depth dimension to combine blobs of color and present various oriented edges.

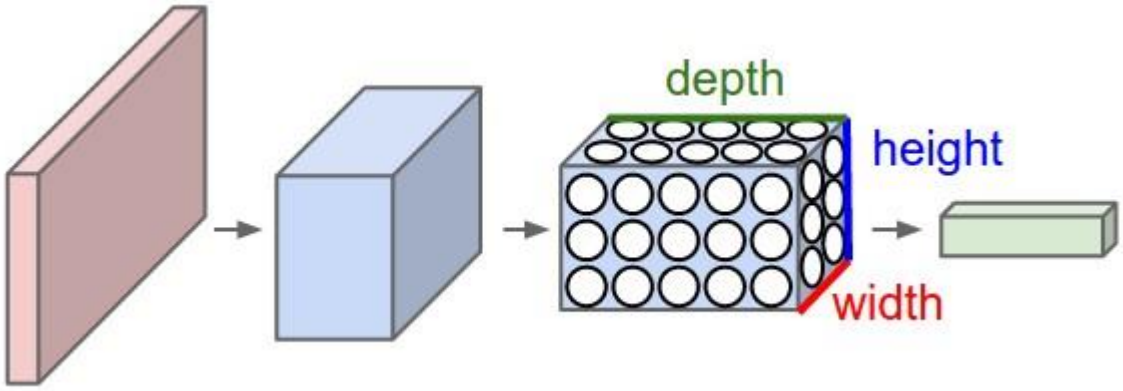

Figure 2: Depth in a neural network

The above image shows the depth of the neural network, which determines the number of neurons that will be interconnected.

Stride

The stride parameter is used to control the depth allocated around the spatial dimensions like width and height. When the stride of a filter is 1, it the filter moves one pixel at a time, which generates a lot of overlapping receptive fields between columns. When the stride is 2, it jumps 2 pixels at a time, when the filter is moving. Hence, for all integers $S > 0$, the filter jumps S units at a time. Although $S > 3$ is fairly rare to find, because there is very less overlapping of receptive fields, which creates a smaller resulting output volume and smaller spatial dimensions.

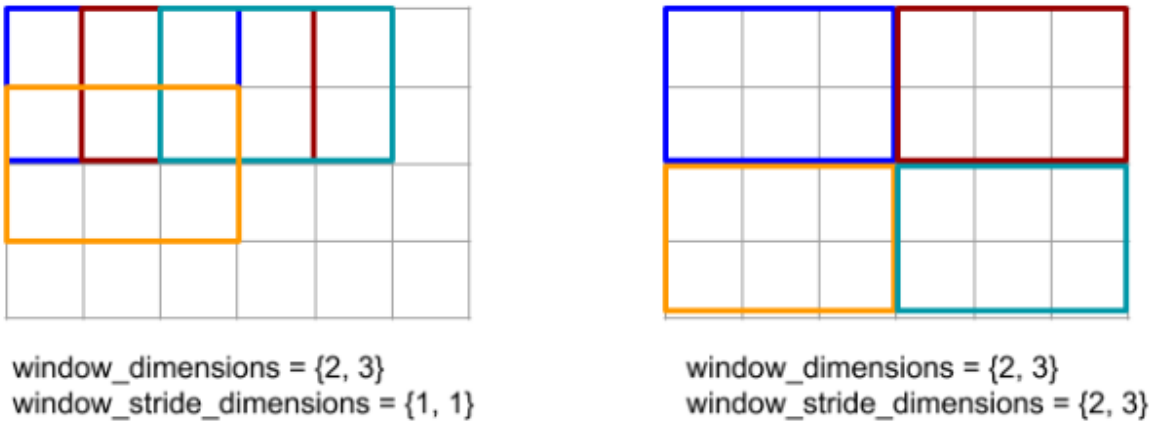

Figure 3: Difference between strides of 1,1 and 2,3

The above image shows the difference between strides of {1,1} and {2,3}. The colored boxes would represent the number of neurons in the next layer.

Zero-Padding

In some of the layers, it is important to pad the edges with 0, in order to avoid miscalculations. The zero-padding parameter controls the length of the padding that needs to be provided. The padding also provides a control over the spatial dimensions of the output volume. There are some cases where the most desired spatial dimensions of output volume are the original dimensions of the input volume. Here's where padding comes into play, in order to maintain the dimensions of the input volume.

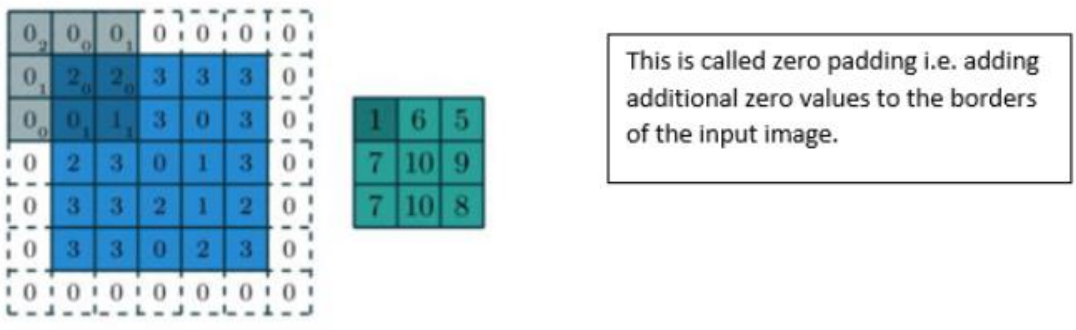

Figure 4: Zero-padding operation on Feature map

The above image shows the Zero-padded input, which adds additional zeros to the border in order to maintain the values on the edge, otherwise they will be only processed once, after which they are removed in the pooling layer.

The spatial dimensions of the output can be determined by a function of the input volume size represented by W, the kernel field size of the Convolutional layer neurons represented by K, the Stride which is applied on them represented by S, and amount of the zero-padding applied on the border represented by P. The formula to determine how many neurons can fit in the given volume can be given as

$$\frac{W - K + 2P}{S} + 1$$

When this number is a non-integer, it means that the strides are incorrect, the neurons cannot be tiled in a symmetric way across the input volume. When S =1, using P = (K-1)/2, ensures that the input volume spatial size is equal to the output volume dimensions.