Case Study I

Krushang Shah

Prof. Richard Khan

Senior Design I

ID 23821856

Convolutional Neural Network

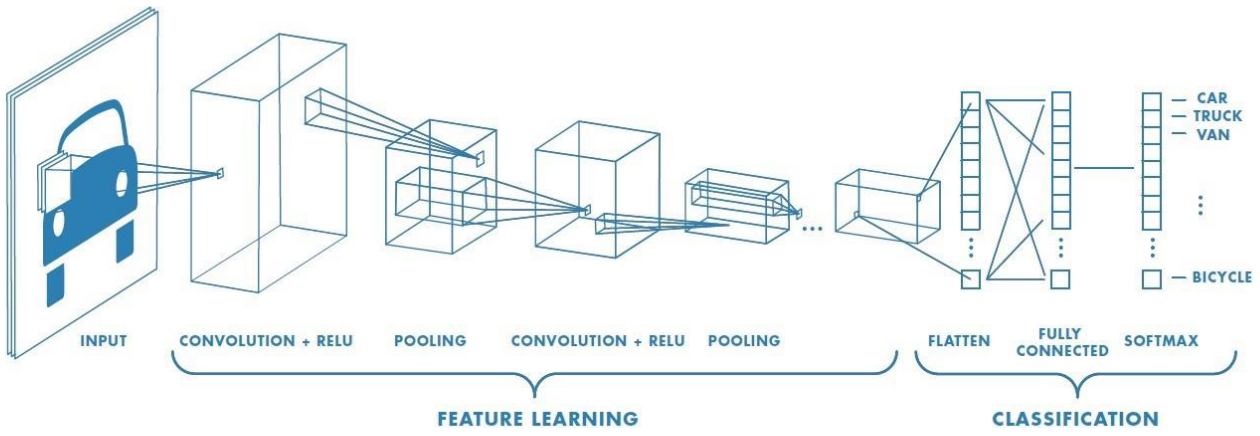
Convolutional Neural Networks are mainly used in the domain of Computer Vision. They are used to enable machines to view the worlds similar to humans, and even use the Intelligence and Knowledge for a multitude of tasks such as Video and Image recognition, Image classification, Image analysis, Media Recreation, Natural Language Processing, Recommendation systems and many more. The advancements in Computer Vision and Deep Learning have been perfected over time, mainly using one main algorithm, which is Convolutional Neural Networks.

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm, which can take an input image, process it through multiple filters, biases and learnable weights. The pre-processing required in a CNN is way lower than that required in a general classification algorithms. While in older methods, filters were designed by humans with enough training, CNN's have an ability to learn these filters and characteristics overtime by themselves.

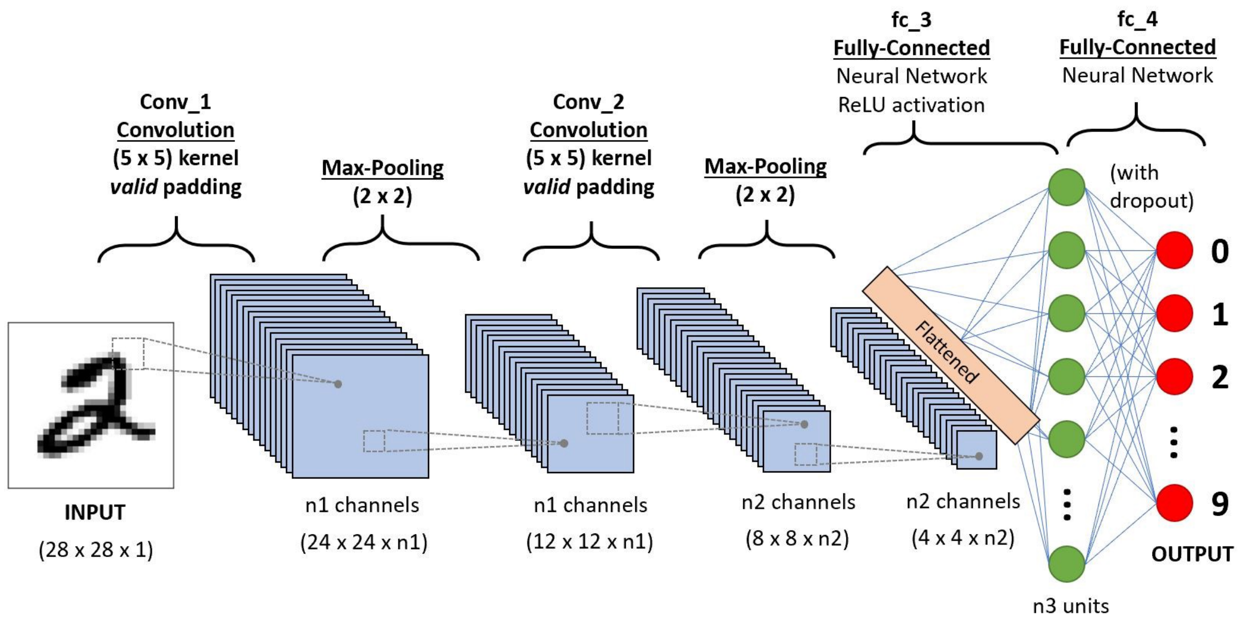
The architecture of a CNN is similar to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field, which is also known as receptive field. A collection of such distinct fields overlaps to cover the entire visual area.

A general Convolutional Neural Network has multiple hidden layers, and an input and output layer. The name convolutional neural network indicates that it uses a mathematical operation called convolution. Convolution is a special kind of linear operation. They use convolution instead of a general matrix multiplication in at least one of their layers.

A general example of a convolutional neural network is shown below, which is used to determine the class or type of the vehicle based on the input, after passing the input image through multiple hidden layers.



Another example shown below is for a convolutional neural network, which uses MNIST dataset for handwritten digit classification. This dataset 70,000 images in it, of which 60,000 are used to train the model and 10,000 are used to test the trained model. They are known as training and testing images respectively.



As it is visible in both images, both the networks have multiple layers, which include layers like Convolutional layer, Pooling layer, Local connectivity, Spatial arrangement, Parameter sharing, ReLU layer, Fully connected layer and loss layer. They are layered in the sequence as per the needs and requirements of the model.

Convolutional Layer: It is the core of the model, which consists of a set of learnable filters. During the pass, each layer is convolved along the width and height of the input volume.

Local Connectivity: Convolutional networks exploit spatial local correlation by enforcing a sparse local connectivity pattern between neurons and each neuron is connected to a small region of the input volume.

Spatial arrangement: The spatial arrangement is controlled by depth, stride and zero-padding of the convolutional layer, which determines the size of the output volume.

Parameter sharing: This scheme is used in convolutional layers, to control the number of free parameters, assuming that if a patch feature is useful to compute at some spatial position, then it can also be used to compute other positions.

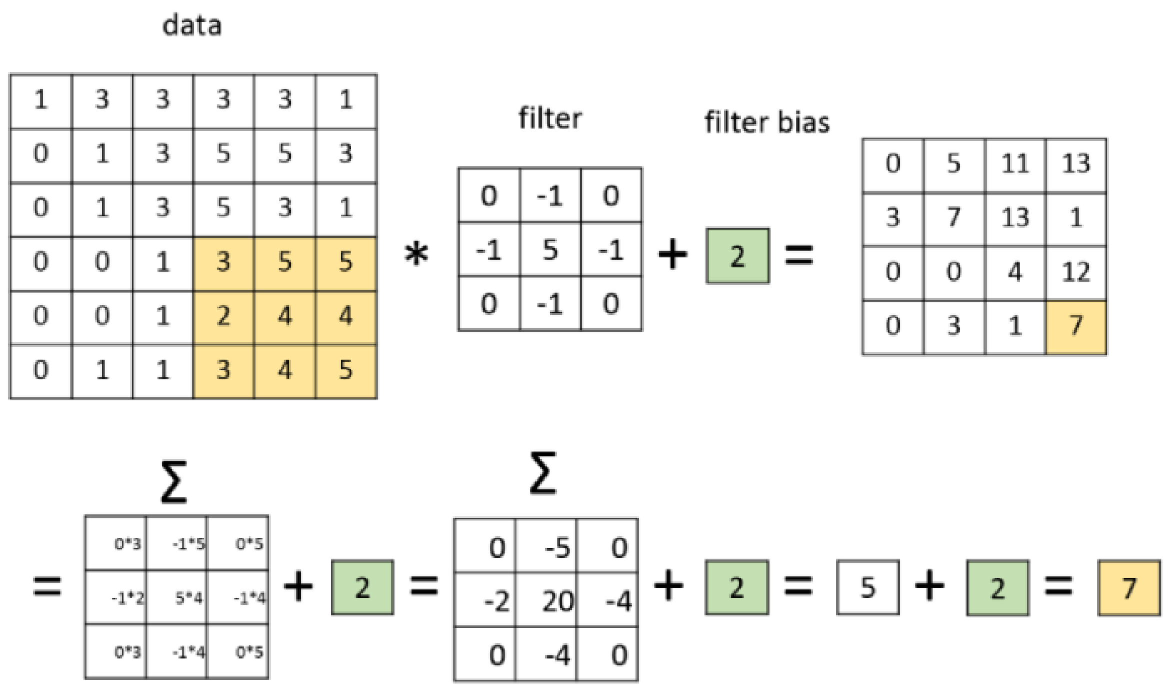
Pooling layer: There are multiple types of pooling layers like max pooling, which takes the maximum in each X*X layer and takes the maximum from each block. The average pooling layer is similar which takes average instead of maximum of each blocks.

ReLU layer: ReLU stands for rectified linear unit, which applies the non-saturating activation function. That is it removes negative values from an activation map by setting them to zero.

Fully connected layer: After several convolutional and pooling layers, the high-level reasoning is done via fully connected layers.

Loss layer: This layer specifies how training penalizes the deviation between predicted output and true labels, usually this is the final layer of a neural network.

2-D convolution example



The above given is an example of 2D Convolution. It has a data matrix, let's say of orientation n*n, and it also has a filter and a filter bias. The data is convoluted with filter and then all the cells of the obtained results are added together as shown in the figure. Finally, the filter bias is added to the sum of the (data*filter) and then we obtain the value of the one cell which is the output after applying the convolutional layer.

So, it can be summed up as output layer after convolutional layer

= (data * filter) + filter bias

This is for a grayscale image, if the input is a colored image, then this layers are separated based on either RGB or CMYK format into separate layers after which each of the separated layers are converted to grayscale (because they have different intensity, hence they would be different) and finally, each of these separate grayscale layers are convoluted separately, to obtain the ouput after the convolutional layer.