**Image Processing**

Ex. input image = 150 x 150 pixels dimensions and color channels = 3 refers to (R,G,B).

so, parameters = 150x150x3 = 67500

If we use 128 neurons to build model, total parameters = (67500 + 1 bias) x 128 = 8.6 million

model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(32, 32, 3)))

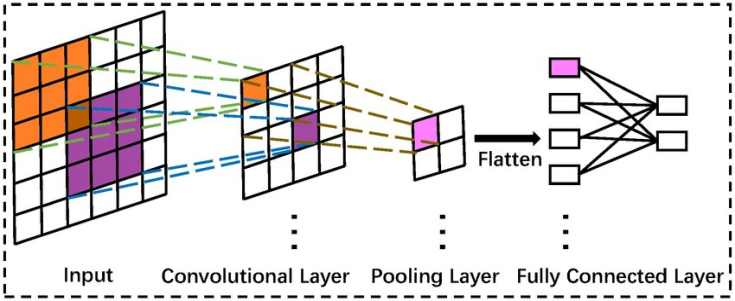
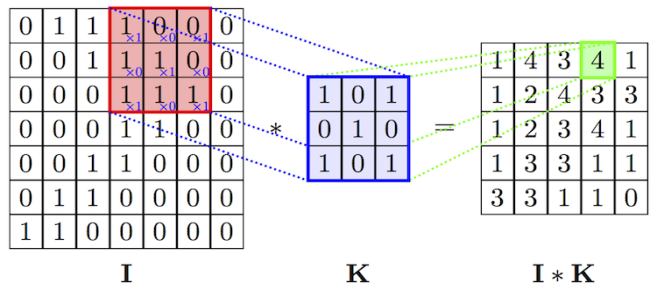
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

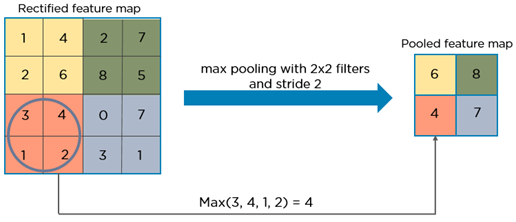
model.add(layers.Conv2D(64, (3, 3), activation='relu'))

**Convolutional Layer (filter 3x3)**

****

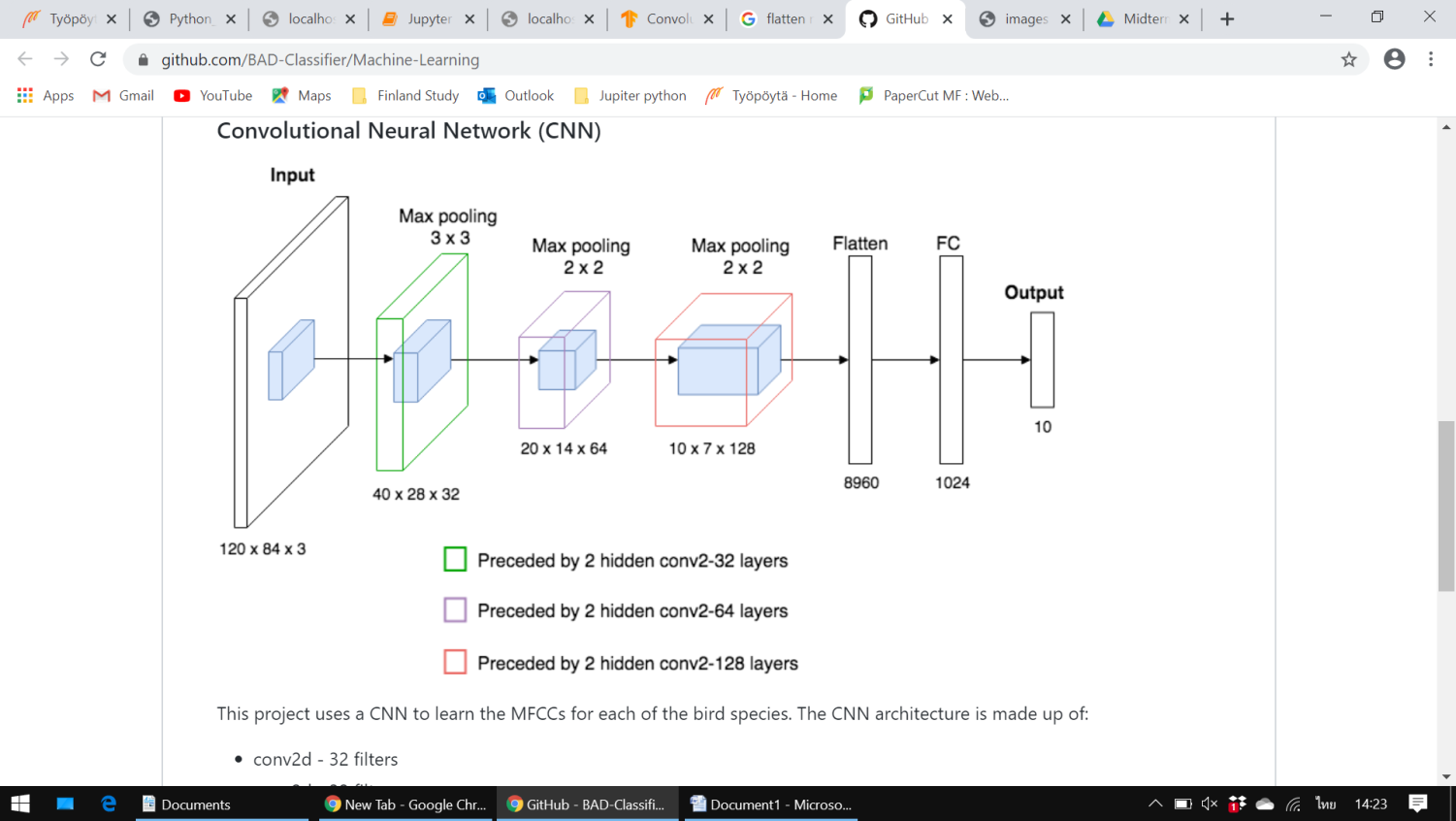
- Image size will reduce -2 in each dimension after convoluted with 3x3 filters in the 1st hidden layer.

**Pooling Layer (2x2)**

****

- Image size will reduce /2 in each dimension after max pooling with 2x2 filter.

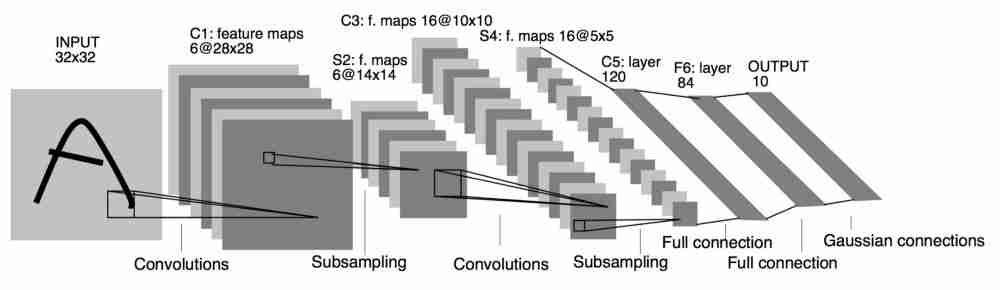
Ex. 150x150 > convolution (3x3) 5 filters >148x148x5 > max pool (2x2) > 74x74x5



Need to specify before doing convolution

* Numbers of filters
* Filter/window/receptive field size
* Stride/padding

**LeNet-5 (1994)**



Filter size in C1: 5x5 S2: 2x2 C3: 5x5

Trainable Parameters in C1: 6\*(5x5x1+1) = 156 C3: 16\*(5x5x6+1) = 2416

C5: 120\*(5x5x16+1) = 48,120

Neuron in that layer \* (w.filter x h.filter x no. of input + 1 bias)