Dimension of the Hidden layer

Let's take the size of the input image to be $n_W^{[0]} \times n_H^{[0]}$.

Typically the filter size fs, set to be 3, 5 or sometimes 7.

This determines how local we want the model to be.

The size of the hidden layer would be:

$$\left[n_W^{[0]} - fs + 1 \right] \times \left[n_H^{[0]} - fs + 1 \right]$$

e.g. $fs = 3$

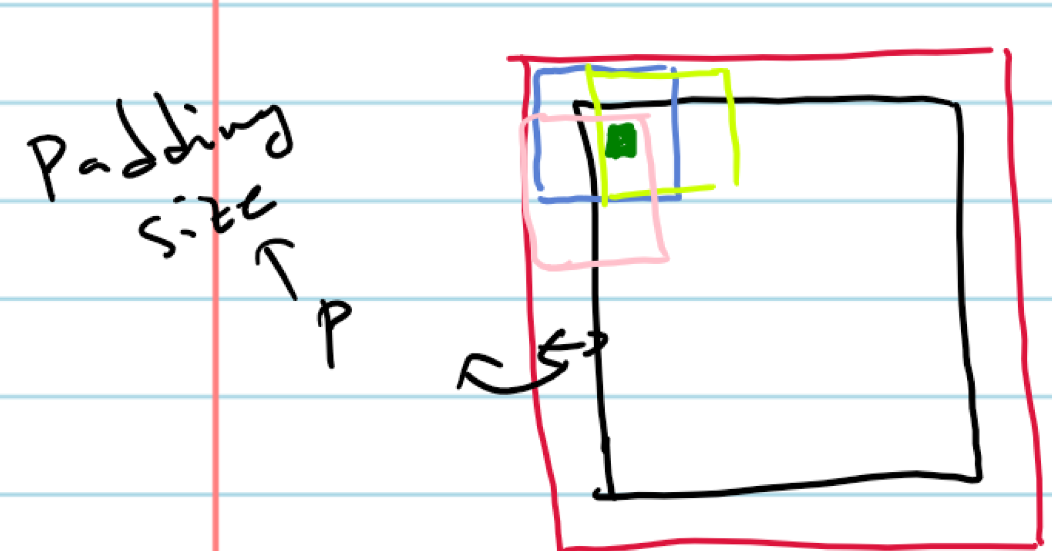
$\hookrightarrow n_W^{[1]} = n_W^{[0]} - 2 \quad , \quad n_H^{[0]} = n_H^{[0]} - 2$.

This is basically the number of positions that we can put the filter inside the image.

Padding:

A point in the middle of the image would be involved in 9 nodes of the Hidden layer. But points in the border would have less contributions.

To balance this out, we could add padding to the image.



$$n_W^{[1]} = n_W^{[0]} - fs + 1 + 2ps$$

$$n_H^{[1]} = n_H^{[0]} - fs + 1 + 2ps$$

Padding is also used to keep the hidden layer size the same as the input image.

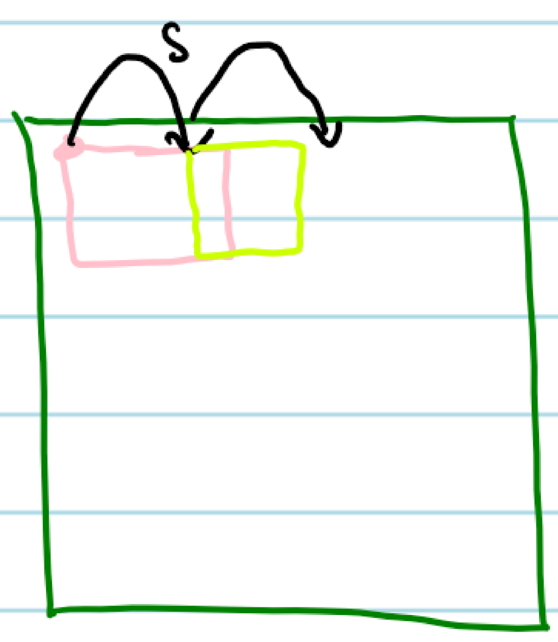
↳ $\cancel{n_W} = \cancel{n_W} - fs + 1 + 2ps \Rightarrow ps = \frac{fs-1}{2}$

e.g. for $fs = 3 \rightarrow ps = 1$

$fs = 5 \rightarrow ps = 2$

Stride

We could also set the step size for moving the filter over the image.



Typically, we take $s = 1$ but it can be treated as another hyperparameter.

With stride of s,

$$n_W^{[1]} = \left\lfloor \frac{n_W^{[0]} - fs + 2PS}{S} \right\rfloor + 1$$

$$n_H^{[1]} = \left\lfloor \frac{n_H^{[0]} - fs + 2PS}{S} \right\rfloor + 1$$

So the hyperparameters for the convolution would be

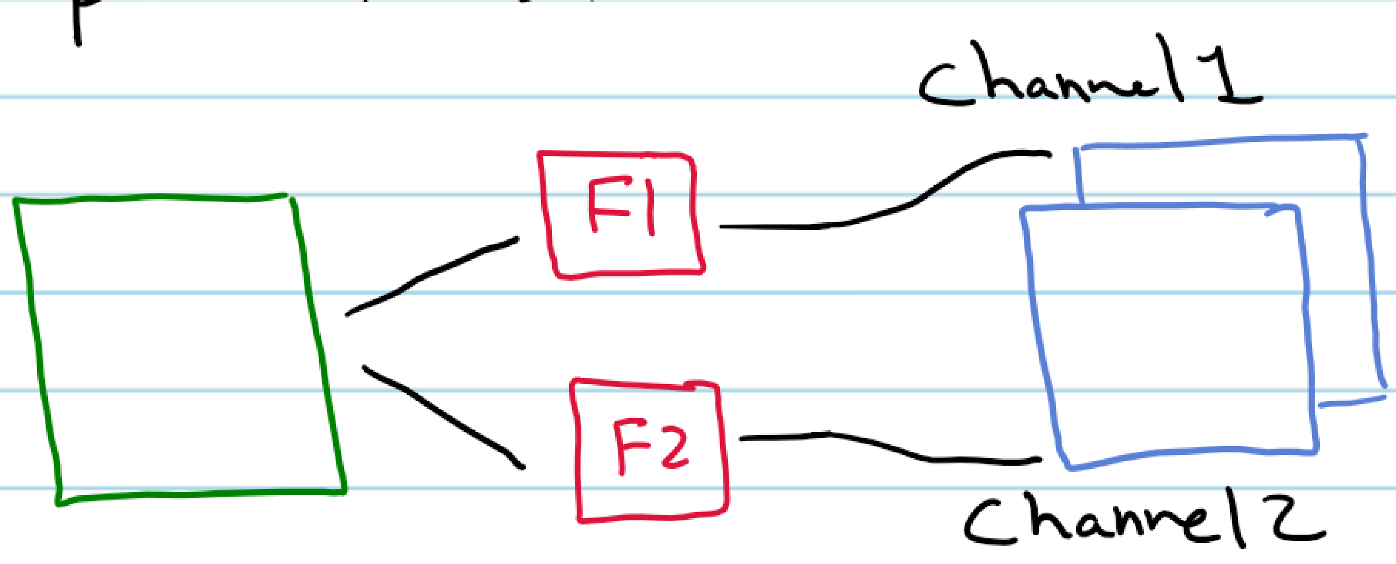
Filter_size : fs — typically 3, 5, 7

Padding size: PS — Typically: "Same", "Valid" ↳ No padding

Strides , S — " 1.

Channels

A filter is optimized to extract a certain feature through the training process. But it may not be enough to rely on only one single feature. For instance, for edge detection, we probably want to have both the vertical & horizontal edges. This cannot be done with a single filter. So we use multiple filters.

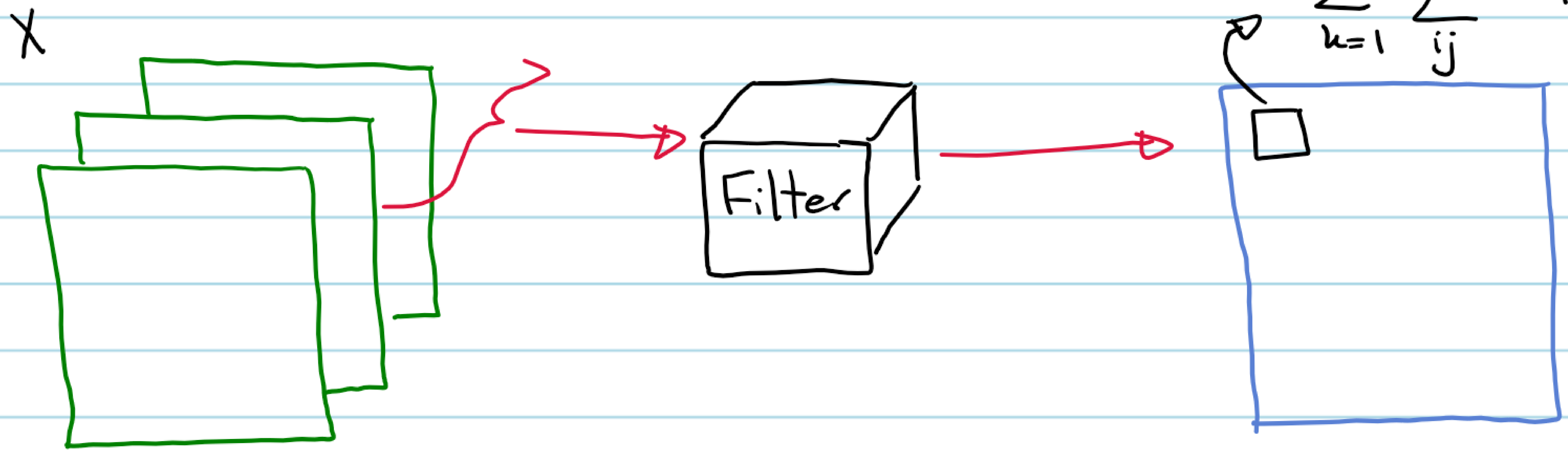


We refer to the # of filters as n_c, the number of channels. This, sort of, represents the number of features that the conv-layer extracts.

Even the input could have (and usually does) multiple channels. For a typical image, these could be RGB channels.

Applying filter to layers with channels:

$$\sum_{k=1}^{n_c} \sum_{ij}^{fs} F_{ijk} \, X_{ijk}$$

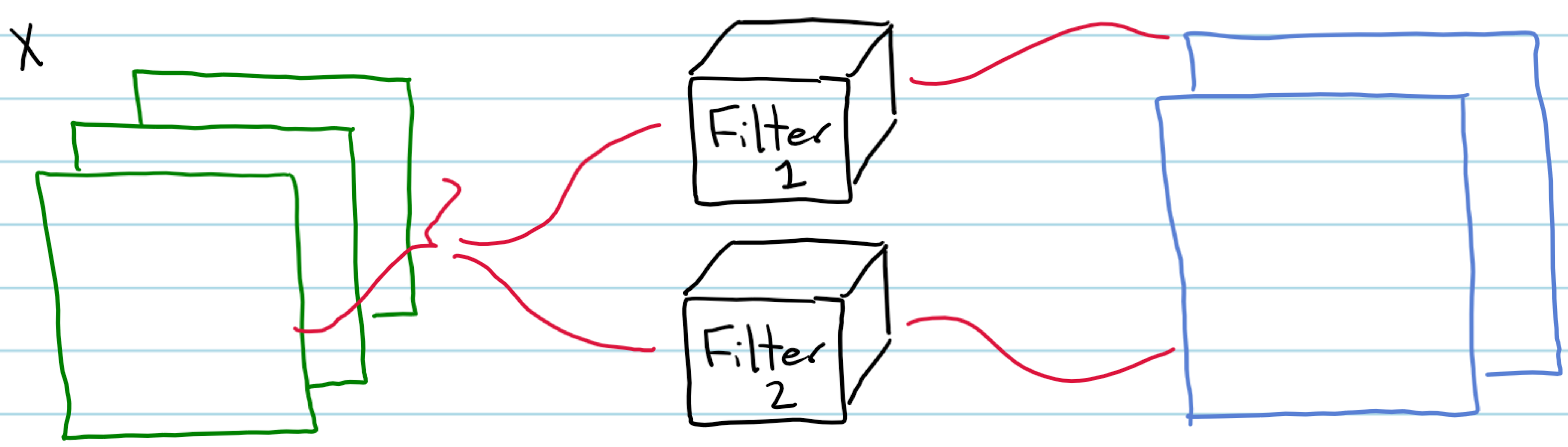


* Filter needs to have the same dimension as the input layer. That means it should be

$$fs \times fs \times n_c^{[l-1]}.$$

* But the outcome would be one channel, i.e. each filter makes one channel.

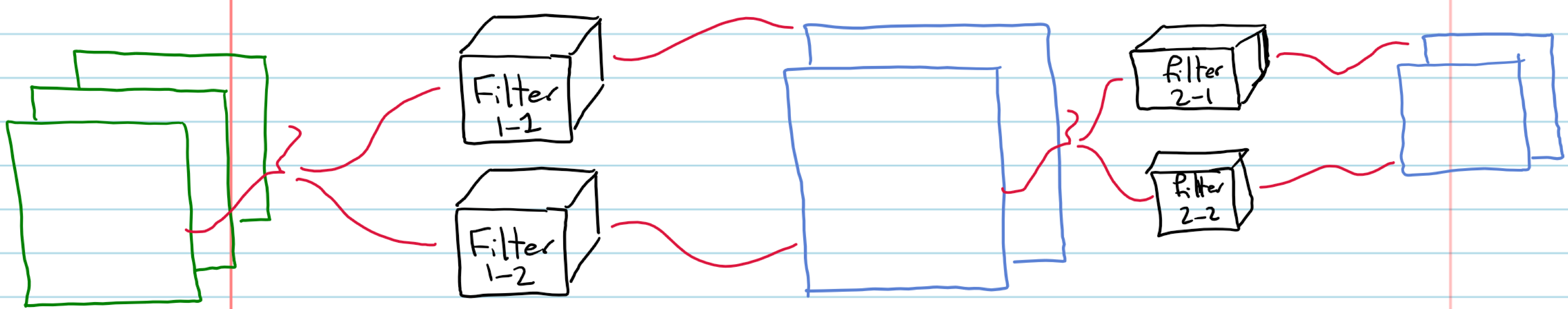
* To increase the # channels, we would need to add more filters.



The number of channels of the hidden layer is given by the # of filters.

Deep conv. net:

We can apply more than just one layer of convolution.



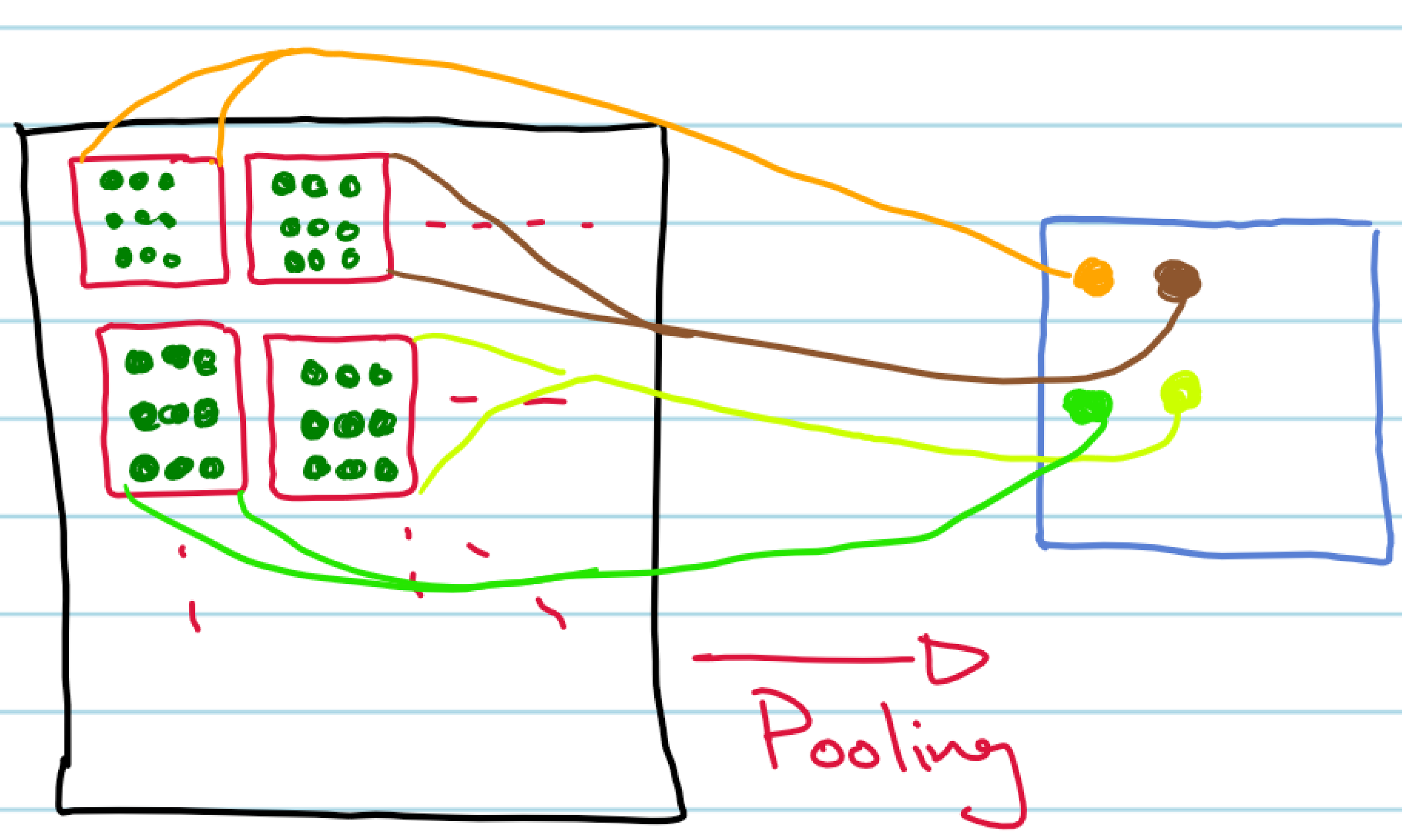
This would mean that the first layer extracts certain features and the following layer builds one top of that to extract more complex features.

Pooling:

This is fairly similar to coarse-graining.

The idea is that we want to reduce the size of the hidden layers as we go deeper in the neural net, i.e. n_H & n_W should decrease. This is b/c we are looking for a few key features that help with the classification task.

With max pooling, we take a window and move it over the layer and pick the max value in that window.



In some sense, the max pooling checks if a certain property exists anywhere in the window of the layer.

We can also take the average which is known as "averge pooling".

Hyperparameters of Pooling

Windows size :	ws	typically 2	3	max vs avg
Stride		2	2	

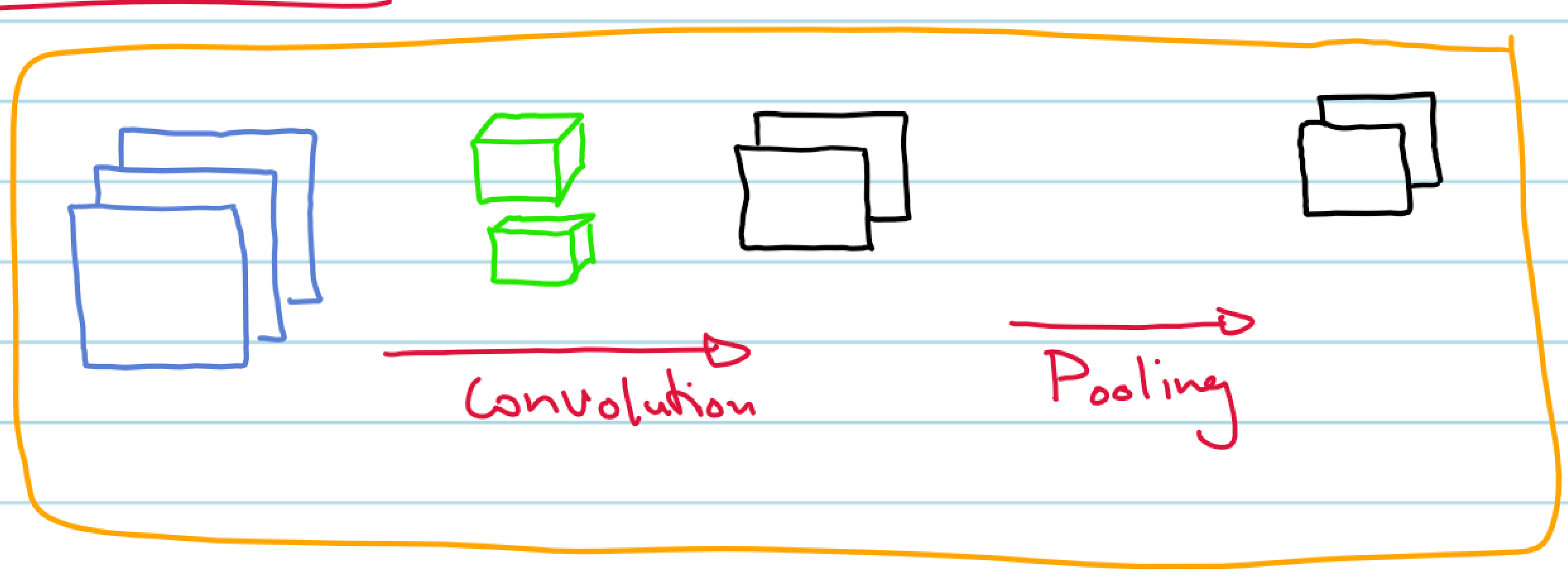
But no learnable parameter.

***Note** Pooling is applied to channels seperately and does not change n_c.

Size $n_w \times n_H \times n_c$

$$\left\lfloor \frac{n_w - ws}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_w - ws}{s} + 1 \right\rfloor \times n_c$$

One layer



One unit of convolution

Fully Connected Layer

Often a full conv_net is composed of several convolution units followed by a couple of fully connected layers.

One can think of the convolution units as feature extractors and the FC layers as the classifier that works/uses the extracted features for classification.

Full Network