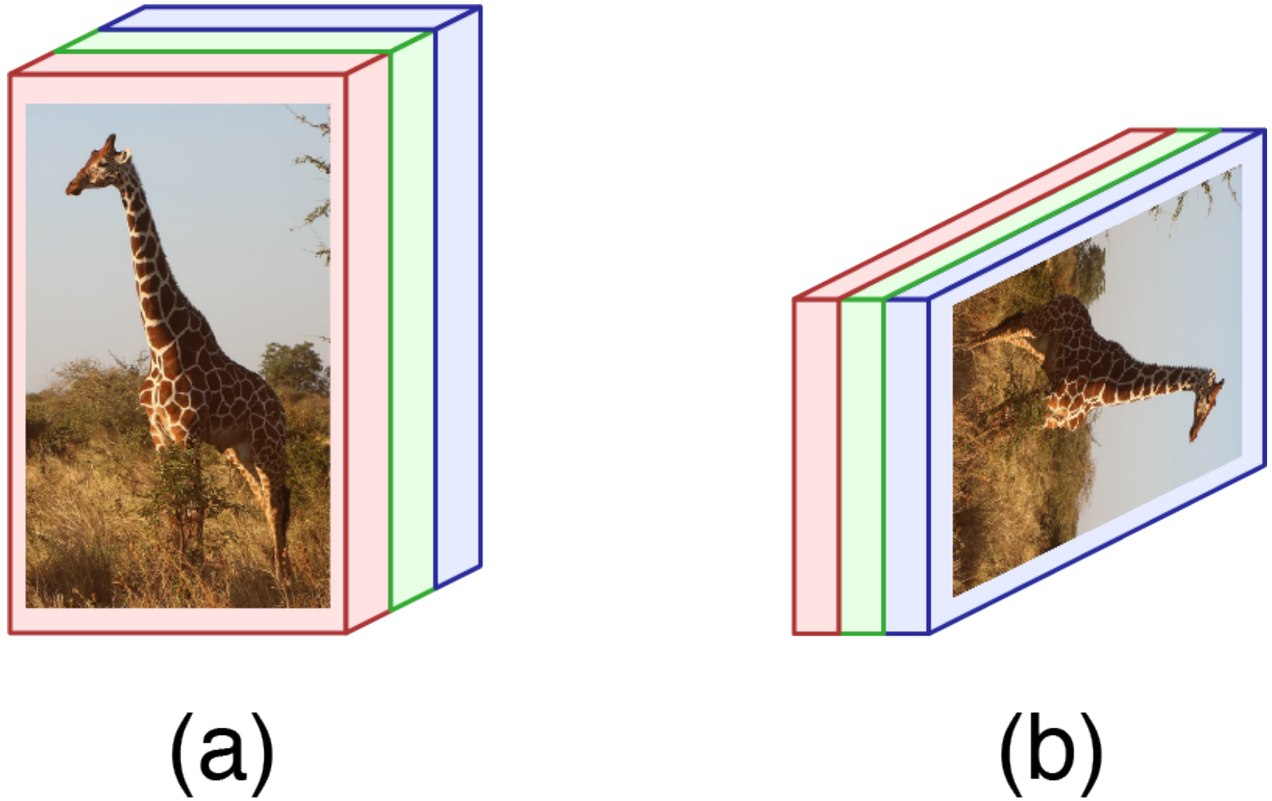
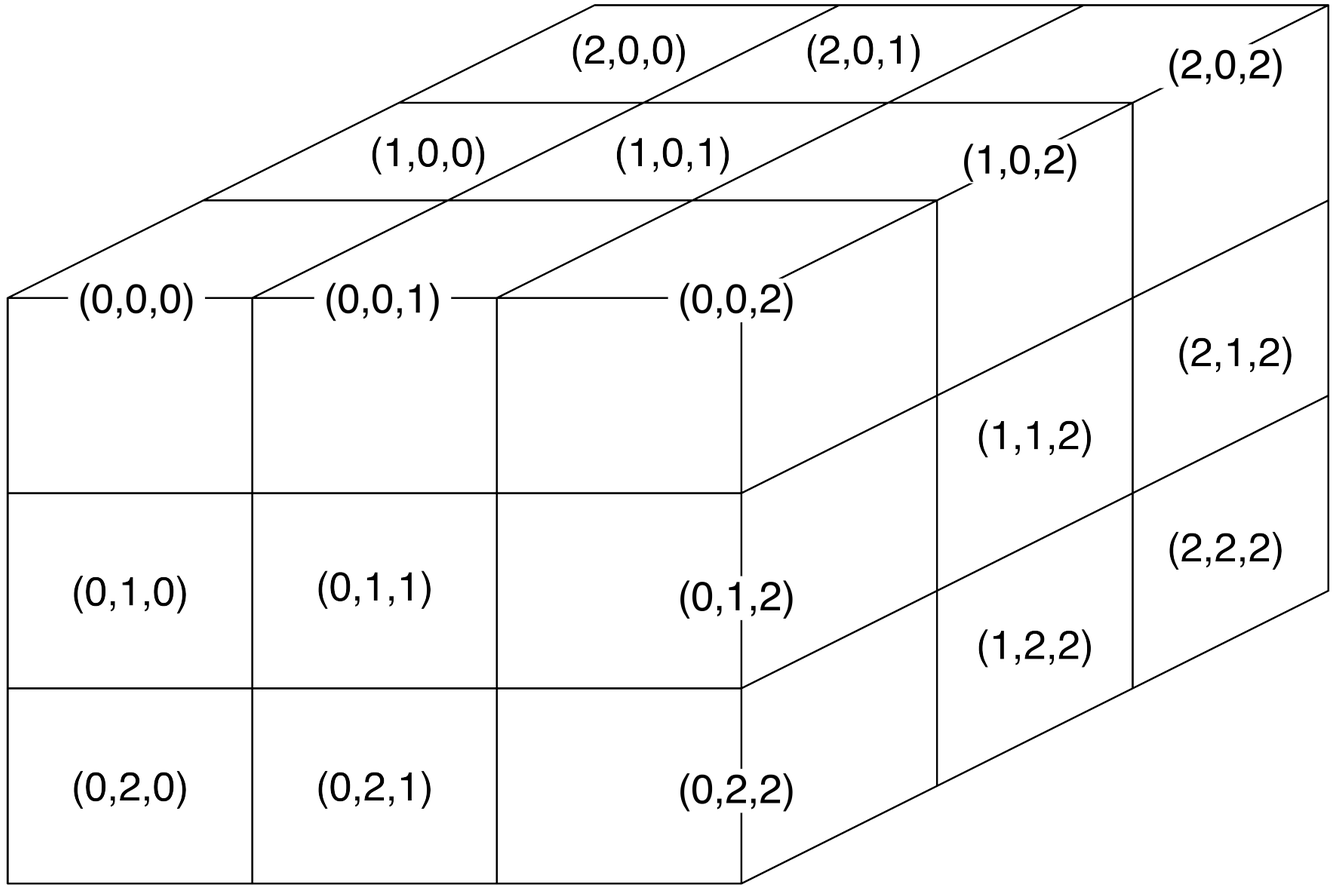
Images

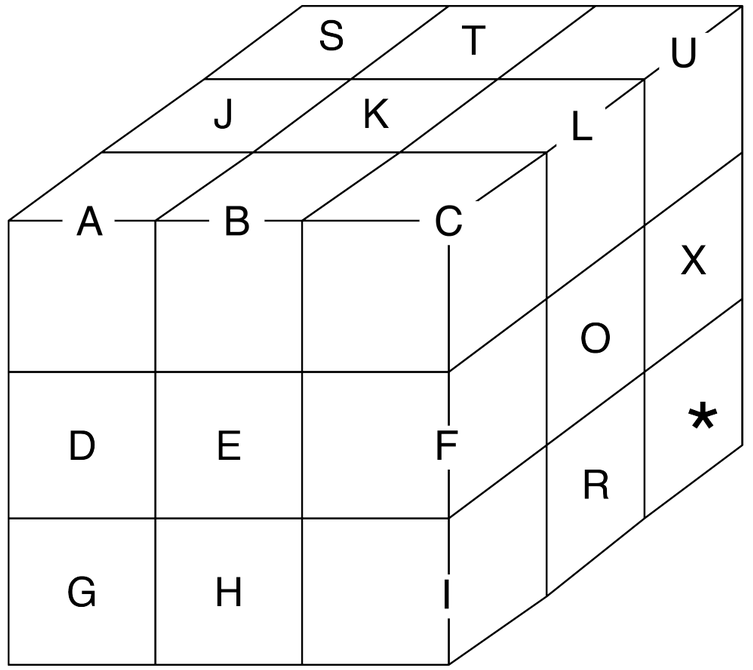


(a) Channels First (Keras) (b) Channels last (pytorch)

Channels first arrangement

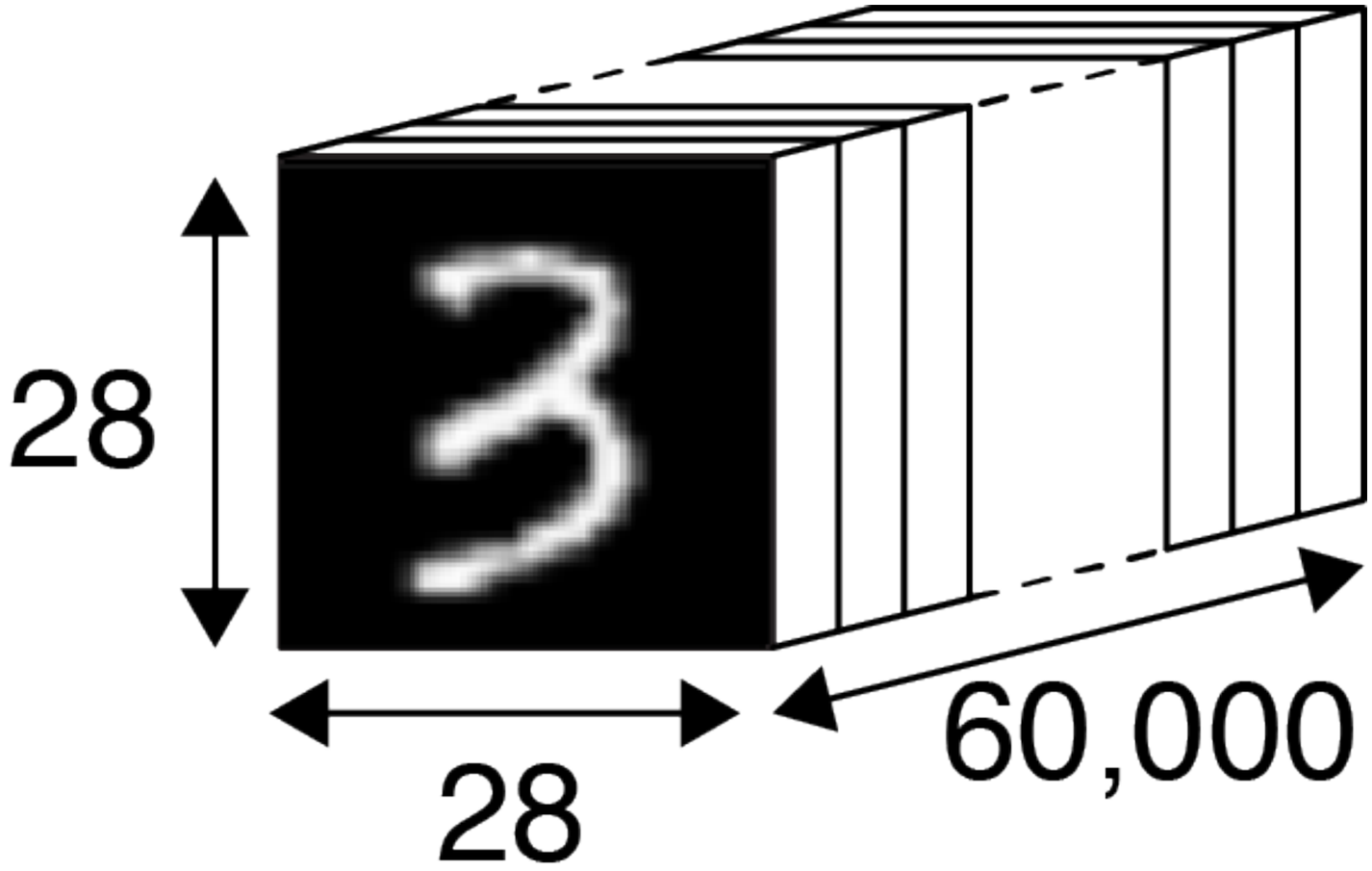


(a)

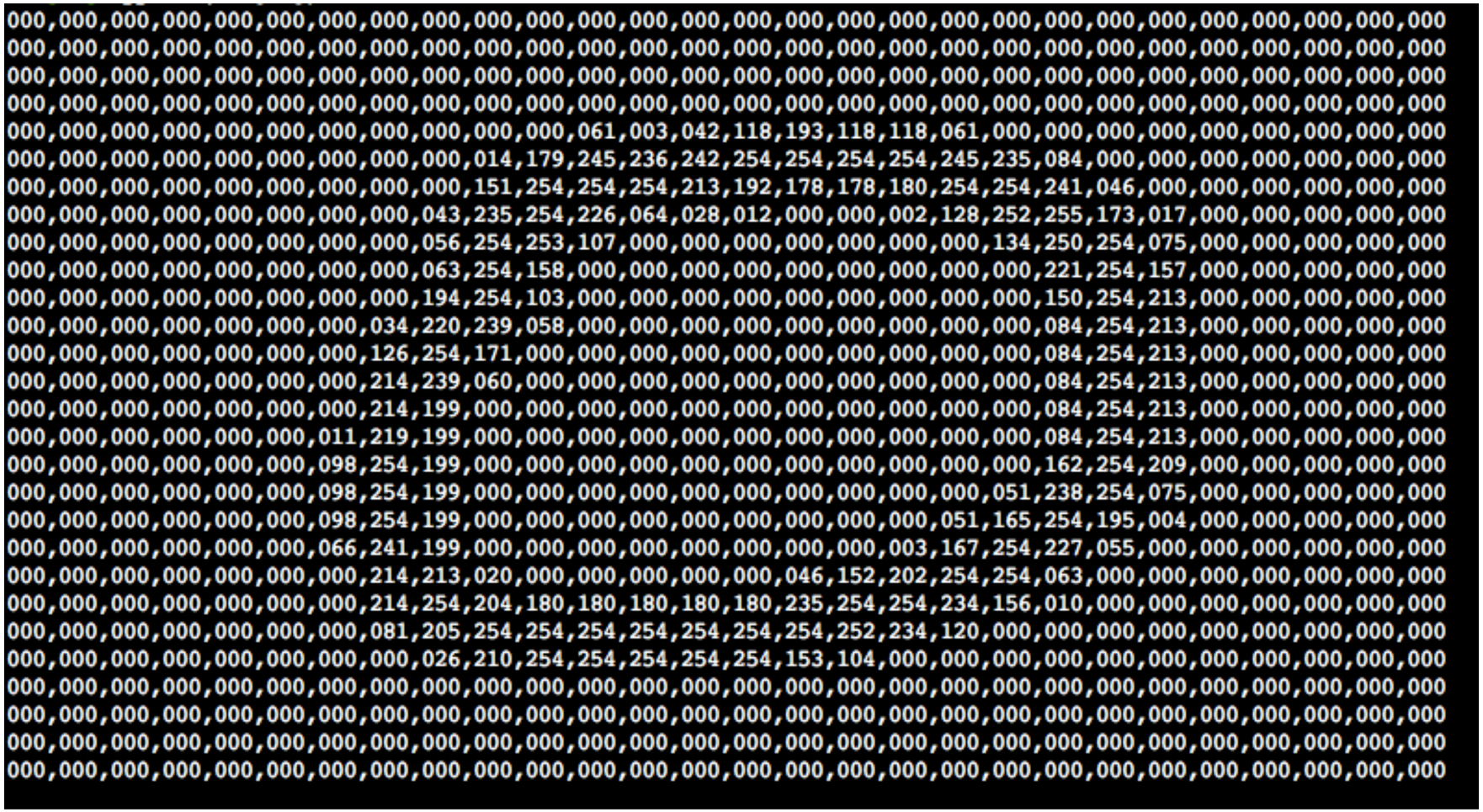


(b)

What about multiple images?



And a single image?

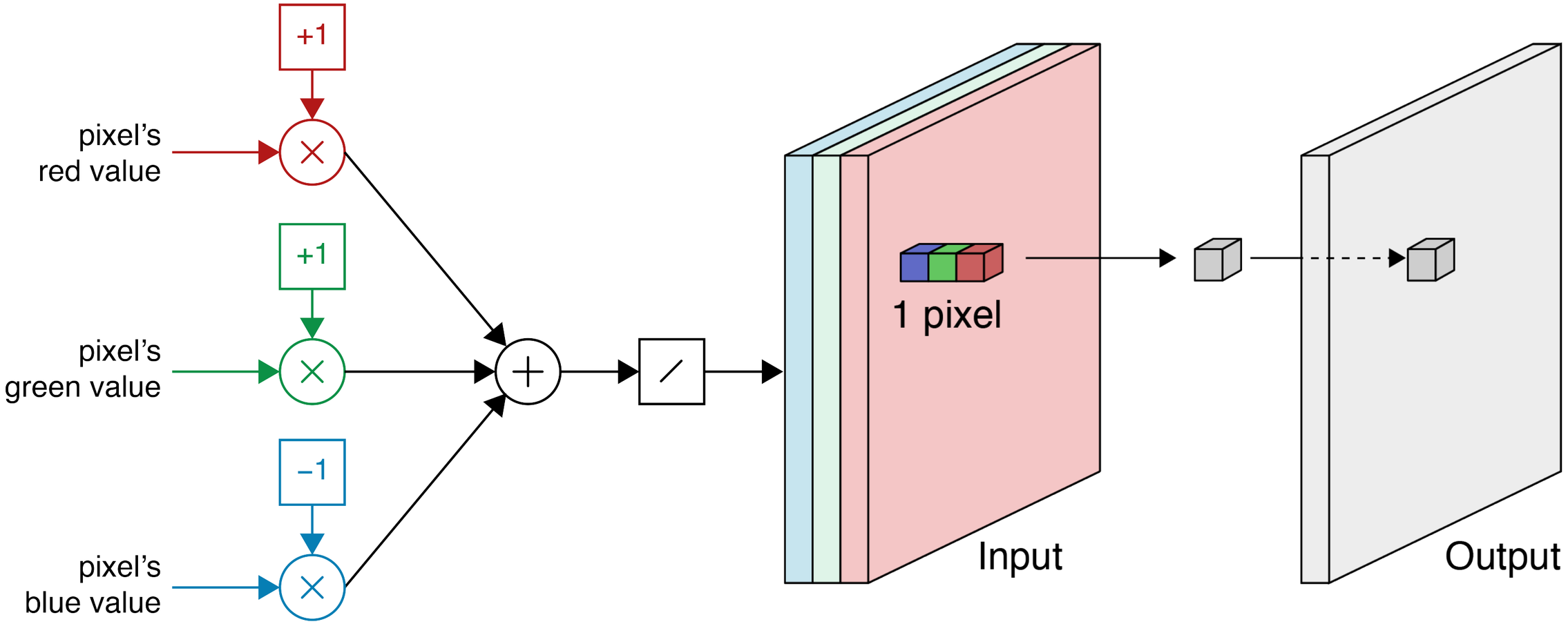


Convolutional Networks

- pay attention to the spatial locality of images
- this is done through the use of "filters"
- thus the representations learnt are spatial and bear a mapping to reality
- and are hierarchical..later layers learn features composed from the previous layers
- perhaps even approximating what the visual cortex does.

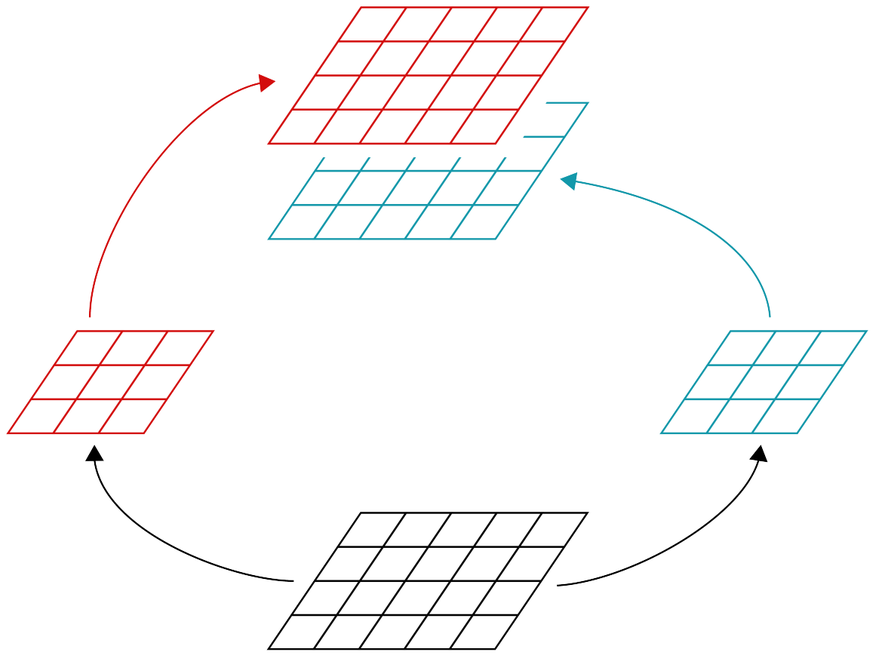
Deep Learning Components

The idea of a filter: detecting yellow

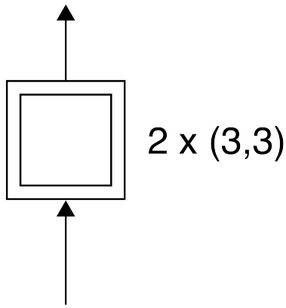


Convolution Layer

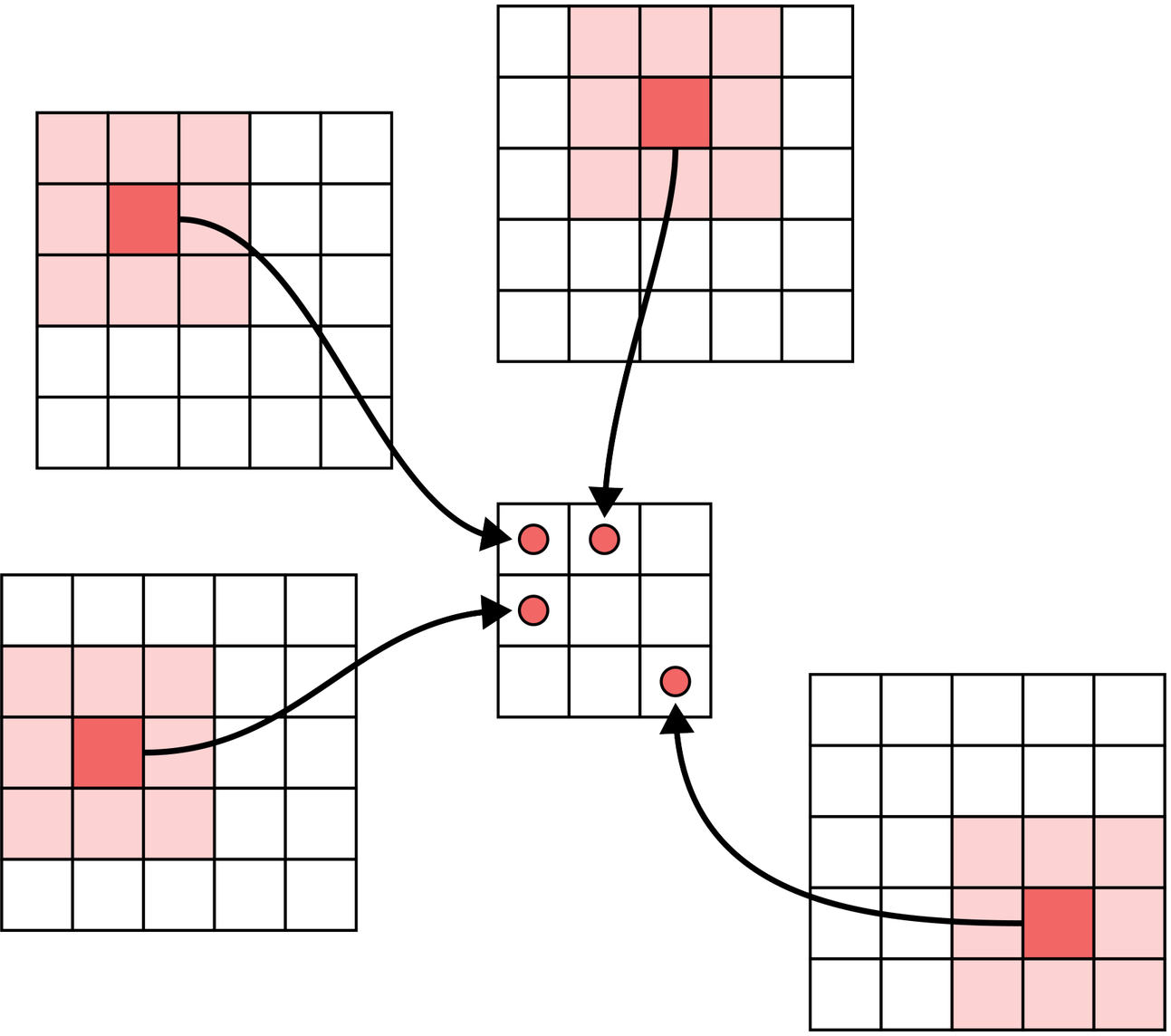
- applies a convolution kernel to an image
- the kernel may have depth, and can use padding on the original image and strides in deciding how to move the kernel window over the image
- the output image(s) depend on these paddings, strides, size of kernel, depth of kernel (channels)
- the kernel usually has weights which multiply the local 'x' in the image and perhaps add bias..in other words, a linear convolution



(a)

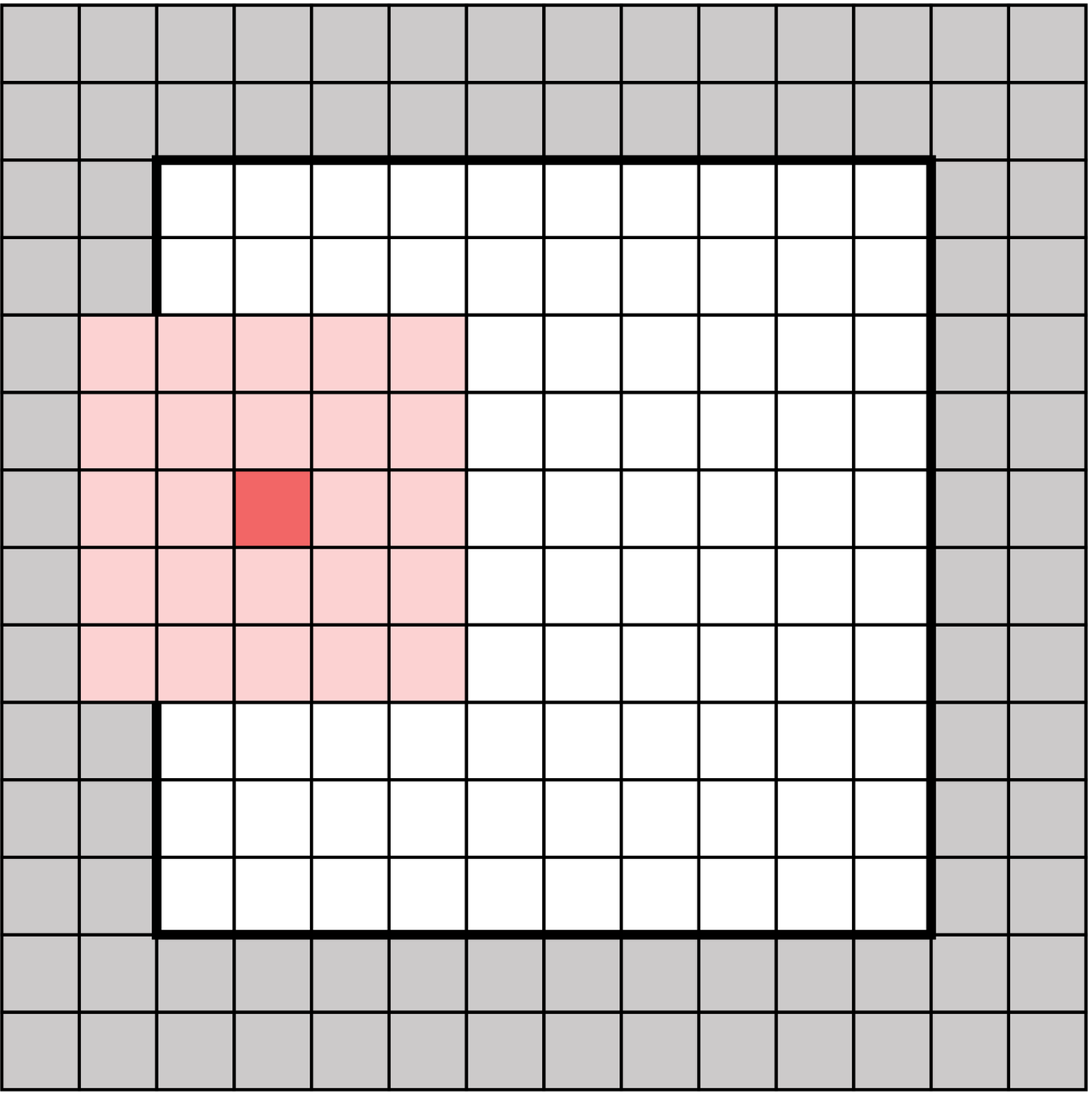


(b)



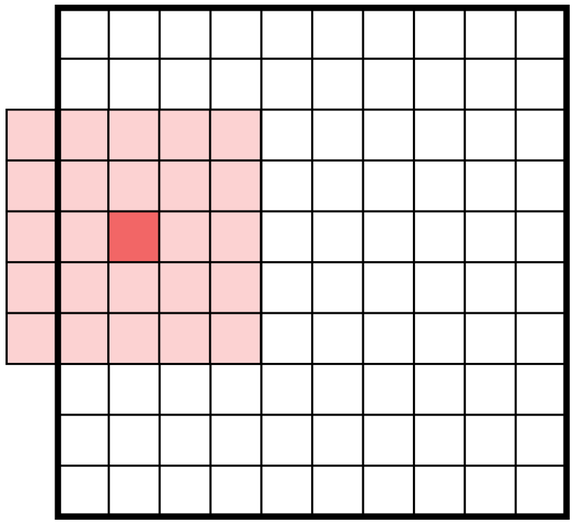
What is a convolution

- slide 3x3 filter across image of size 5x5
- filters must fit into image, thus output will be of smaller size, in this case 3x3
- multiply values in filter by values of pixel and add a bias: $(w, b) \cdot (x, 1)$



Padding

What happens at the edge? We would fall off. Or we cant go to the edge.



- Image size decrease by a convolution is called a "valid" convolution.
- Keeping the same size by 0-padding is called a "same" convolution

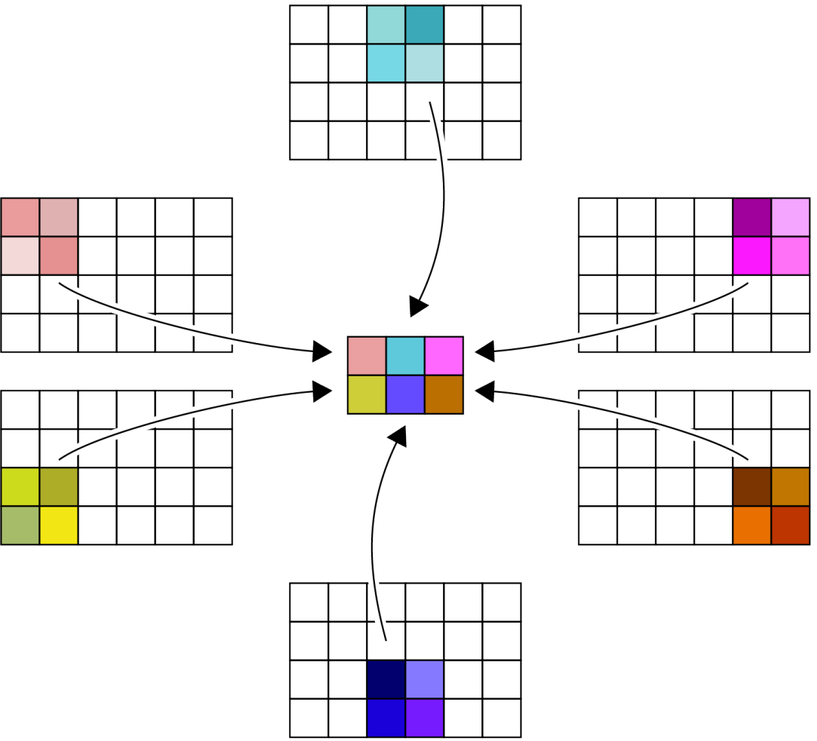
Downsampling

Eventually we want to get to larger length scales across an image, to learn features such as eyes and ears.

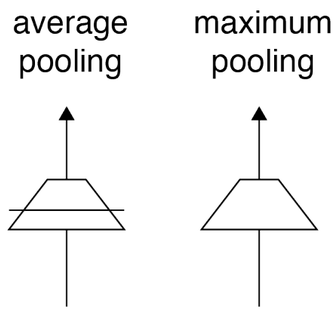
There are two ways to accomplish this:

- pooling, or combining neighboring pixels
- striding, or reducing image size by skipping every nth pixel

Pooling Layer

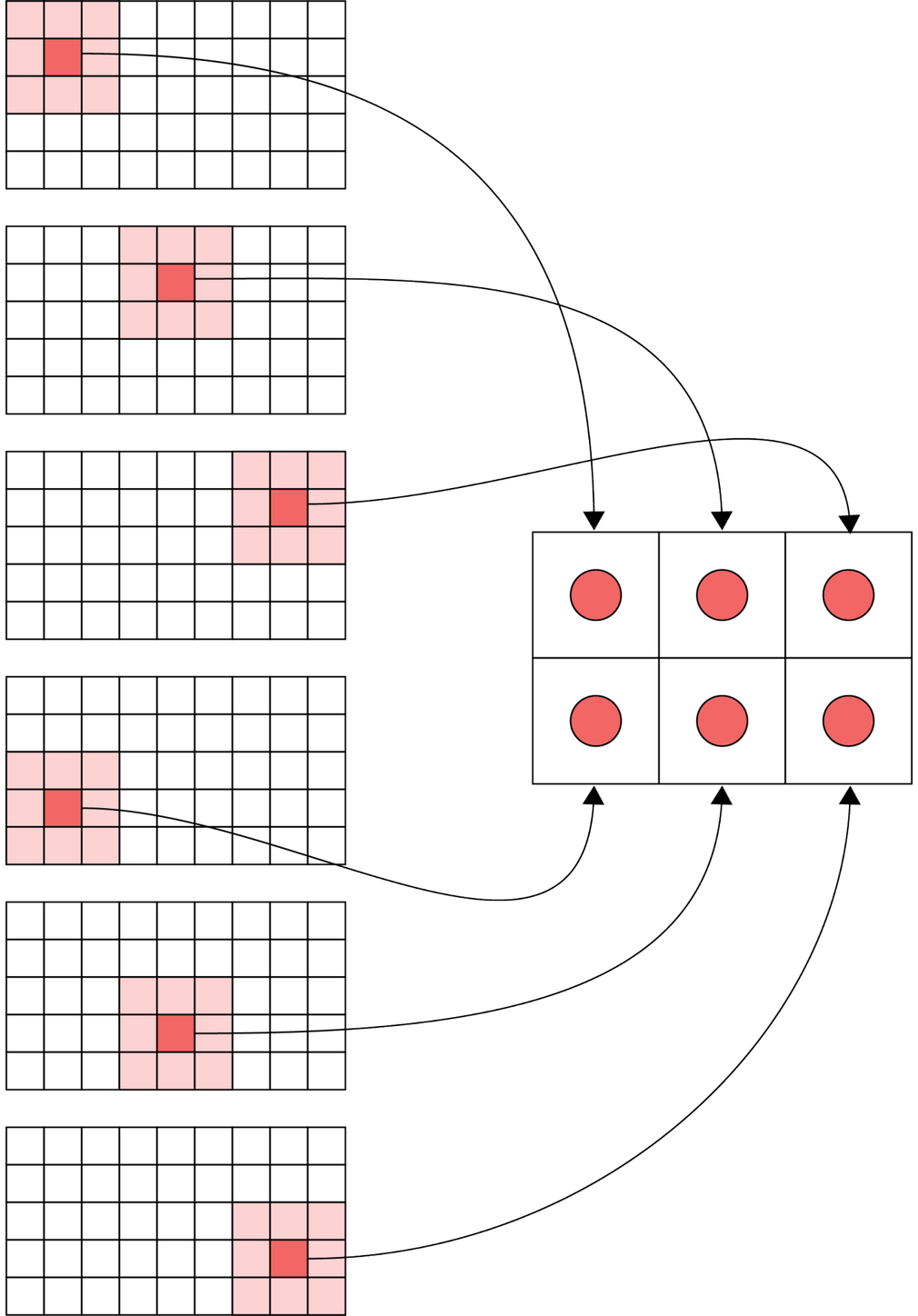


(a)



(b)

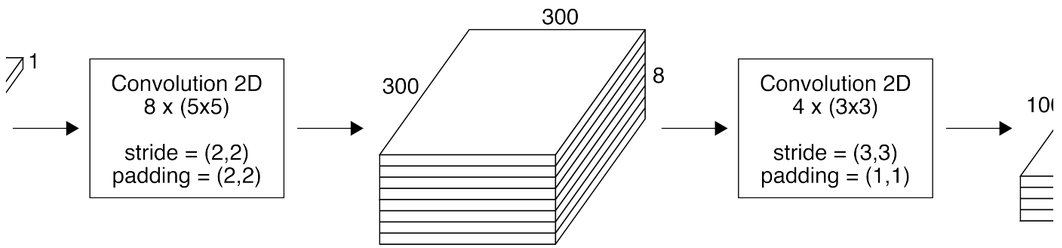
- down samples images, usually using a non-overlapping stride
- 2 common types, max-pooling and average-pooling. Former better for detecting sharp edges
- makes images smaller for manageability, and also helps in learning at the next level in the hierarchy



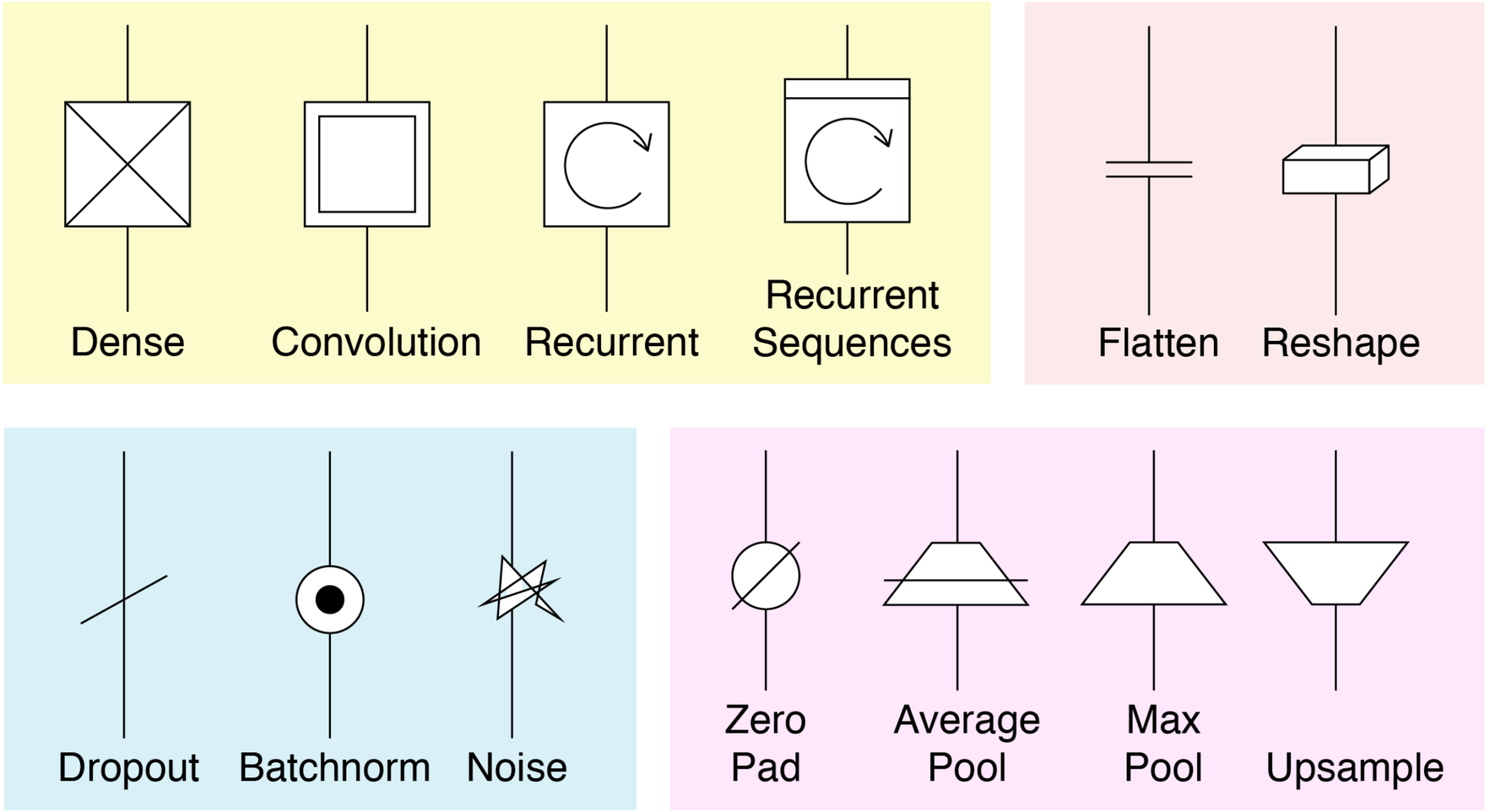
Striding

In recent years, experience has shown that striding while convolving is faster and often gives results that are just as good, or better, than pooling

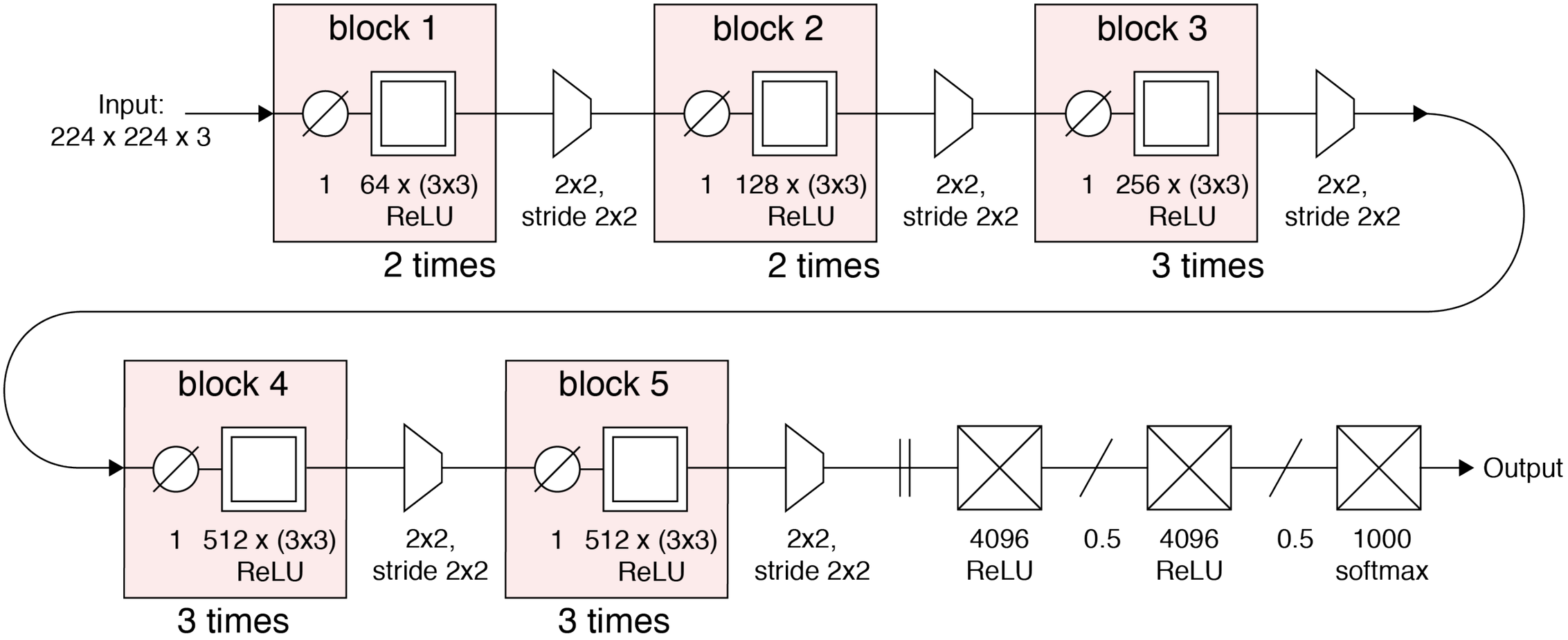
- Image is scaled down at each step,
- Filters work with lower-res versions
- filters can more easily look for larger features



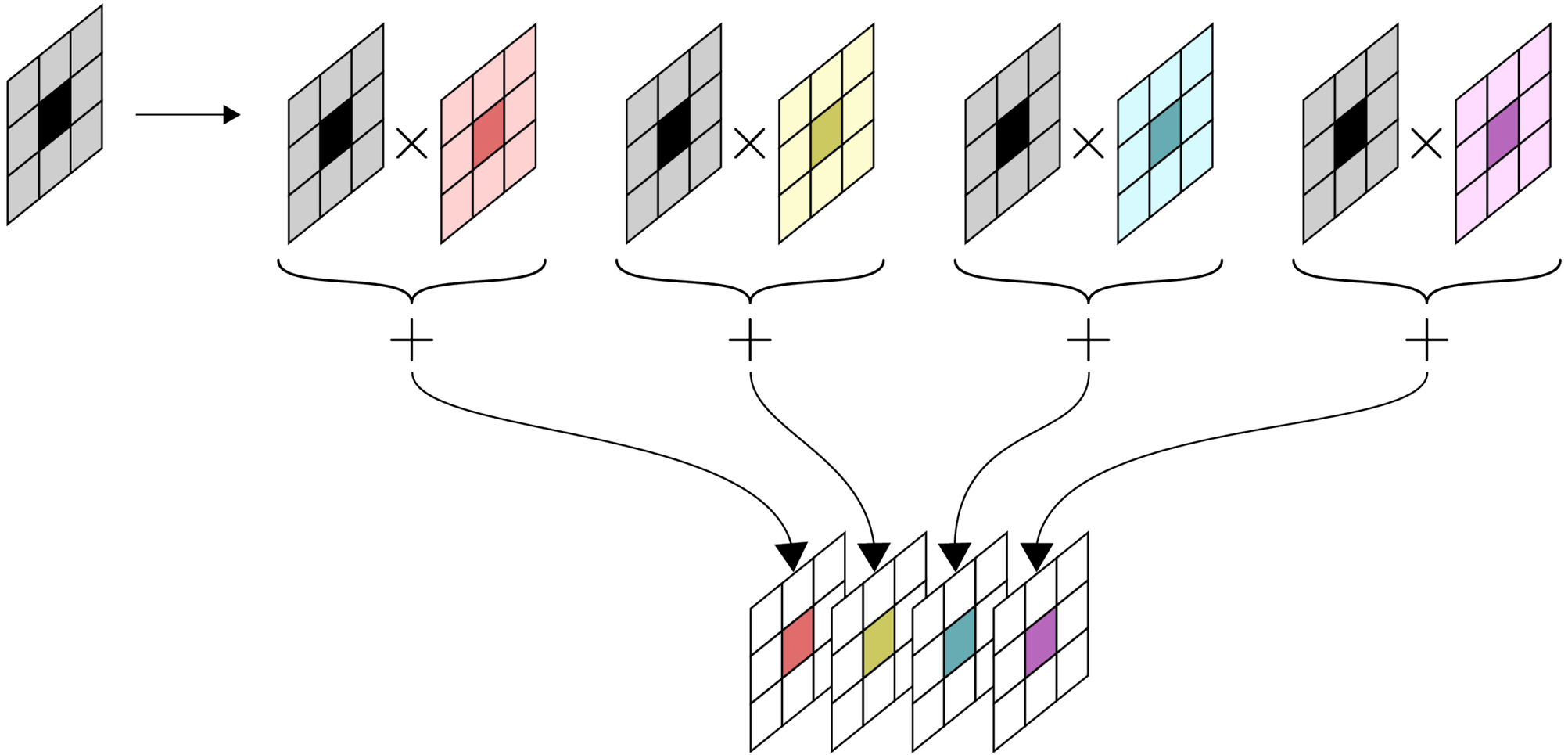
Layer types schematic



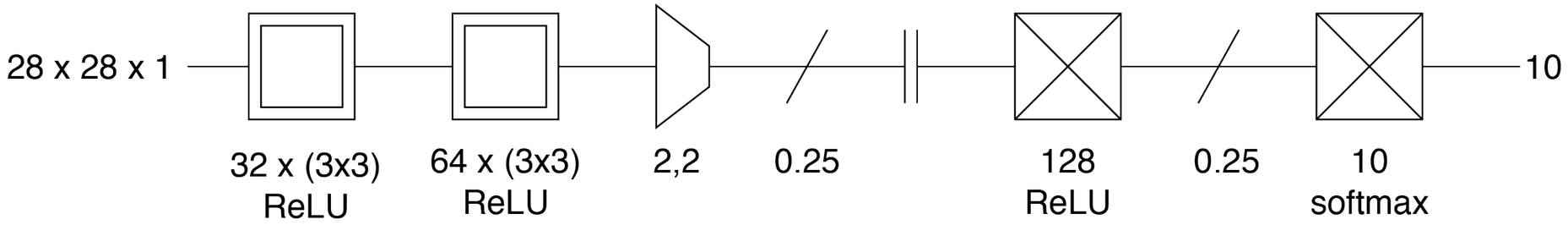
VGG16 Image Classification Example

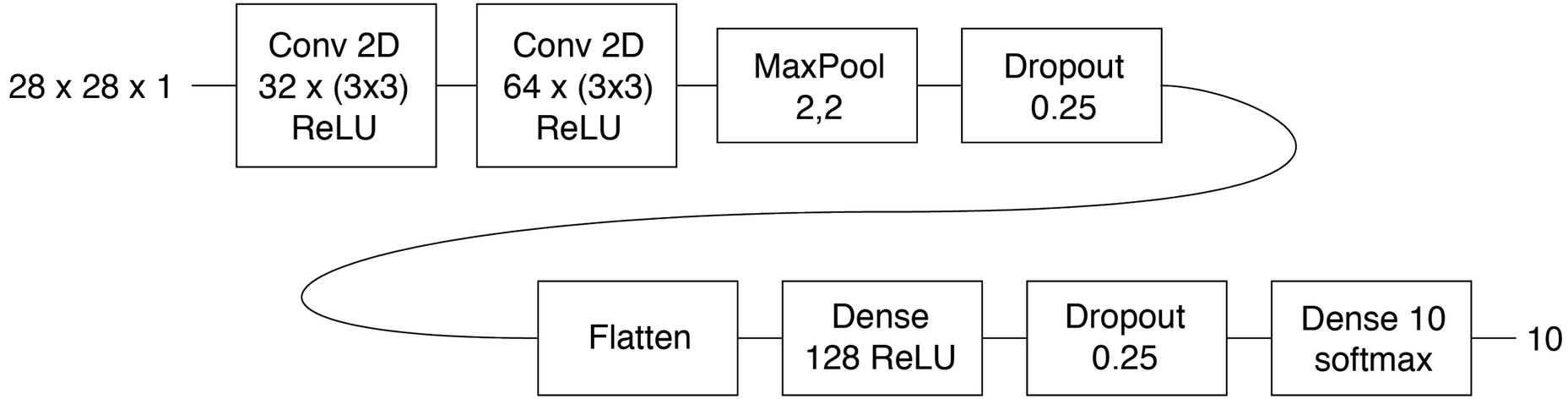


How Channels work: each color a different feature

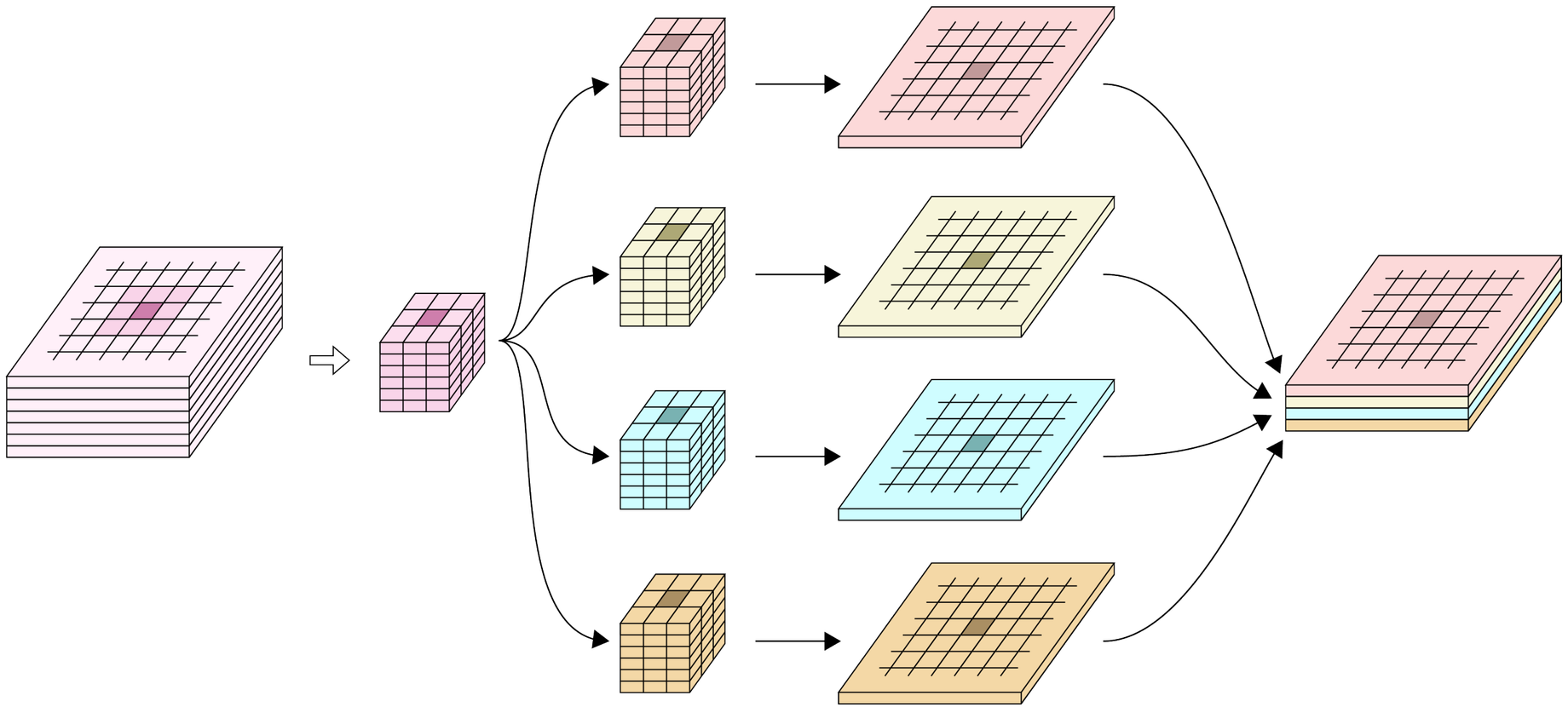


MNIST architecture from Keras Docs





Channel Arithmetic



input is (say) 26x26x6, so filters MUST have 6 channels and we have 4 new featuremaps: `Conv2D(4,(3,3))`