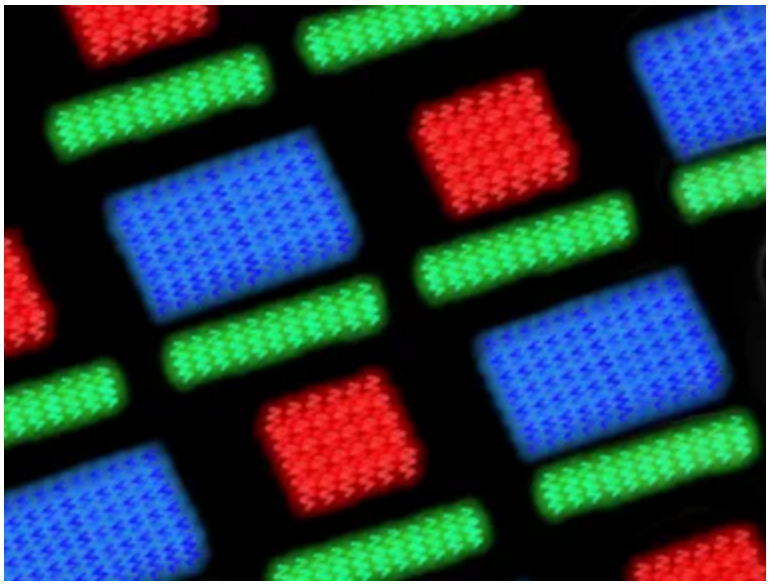


Did you know? That our smart phones use what is called the Organic light emitting diode (OLED) to generate the high-quality images that we see whenever we look at our smartphones. An OLED display is composed of a massive grid of individual pixels and each pixel is composed of a red green, and blue subpixel. This picture shows an OLED grid of microscopic individually controlled dimmable red, green and blue lights . There could be **over 10 million** of these lights on your phone!

1. Convolutions Over Volume

1.1 Convolutions on RGB images

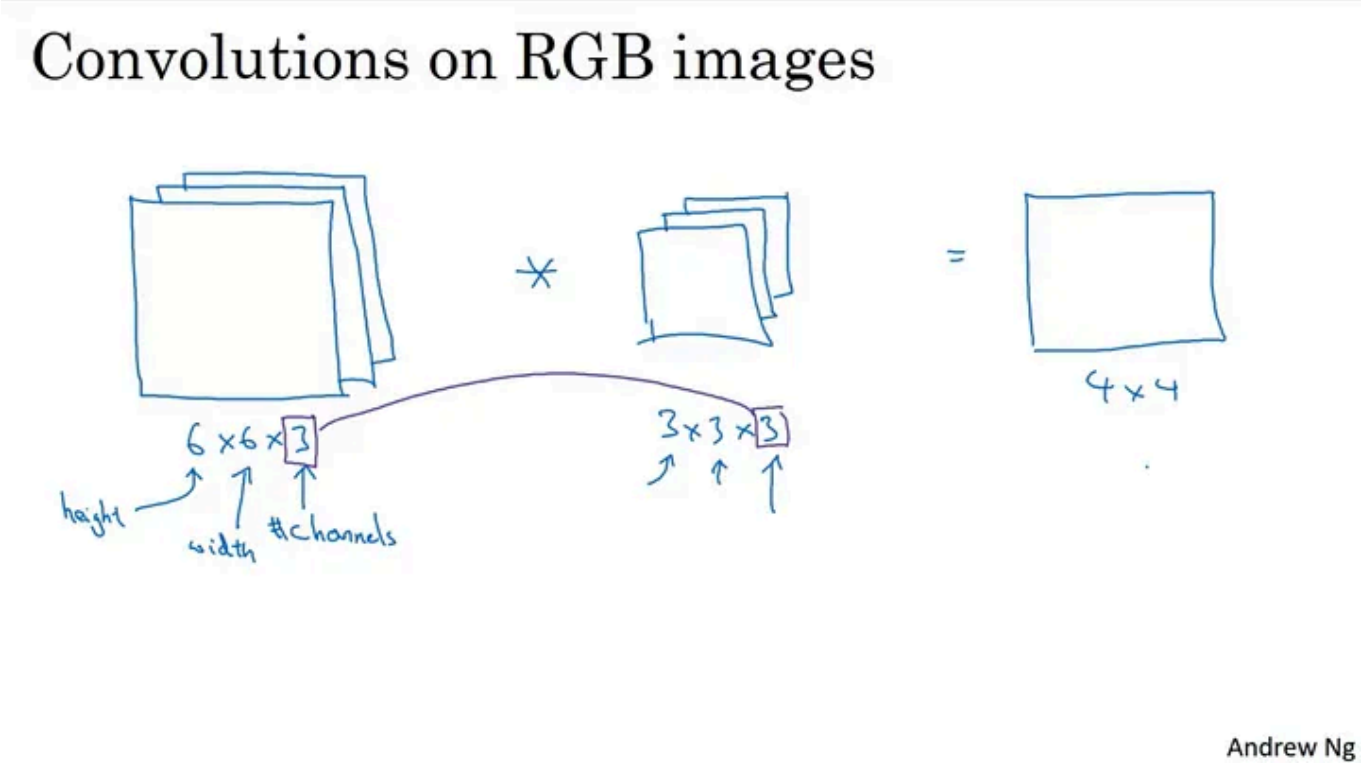


FIGURE 1: Convolving a 6 by 6 by 3 volume with a 3D filter (a 3 by 3 by 3 filter).

An RGB image is represented as a 6 by 6 by 3 volume, where the 3 here responds to the 3 color channels (RGB). In order to detect edges or some other feature in this image, you could convolve the 6 by 6 by 3 volume with a 3D filter (a 3 by 3 by 3 filter) as shown on *figure* 1, not with a 3 by 3 filter, as we have seen in the previous parts. So the filter itself will also have 3 layers corresponding to the red, green, and blue channels. From *figure* 1, notice that the first 6 is the height of the image, the second 6 is the width, and the 3 is the number of channels. And your filter also similarly has a height, a width, and the number of channels. The number of channels in your image must match the number of channels in your filter (notice the last numbers are connected by a curved line on *figure* 1). Also, notice that the output image is a 4 by 4 image (or 4 by 4 by 1 image) instead of a 4 by 4 by 3 image. This will become clear once we look at multiple filters in section 1.2.

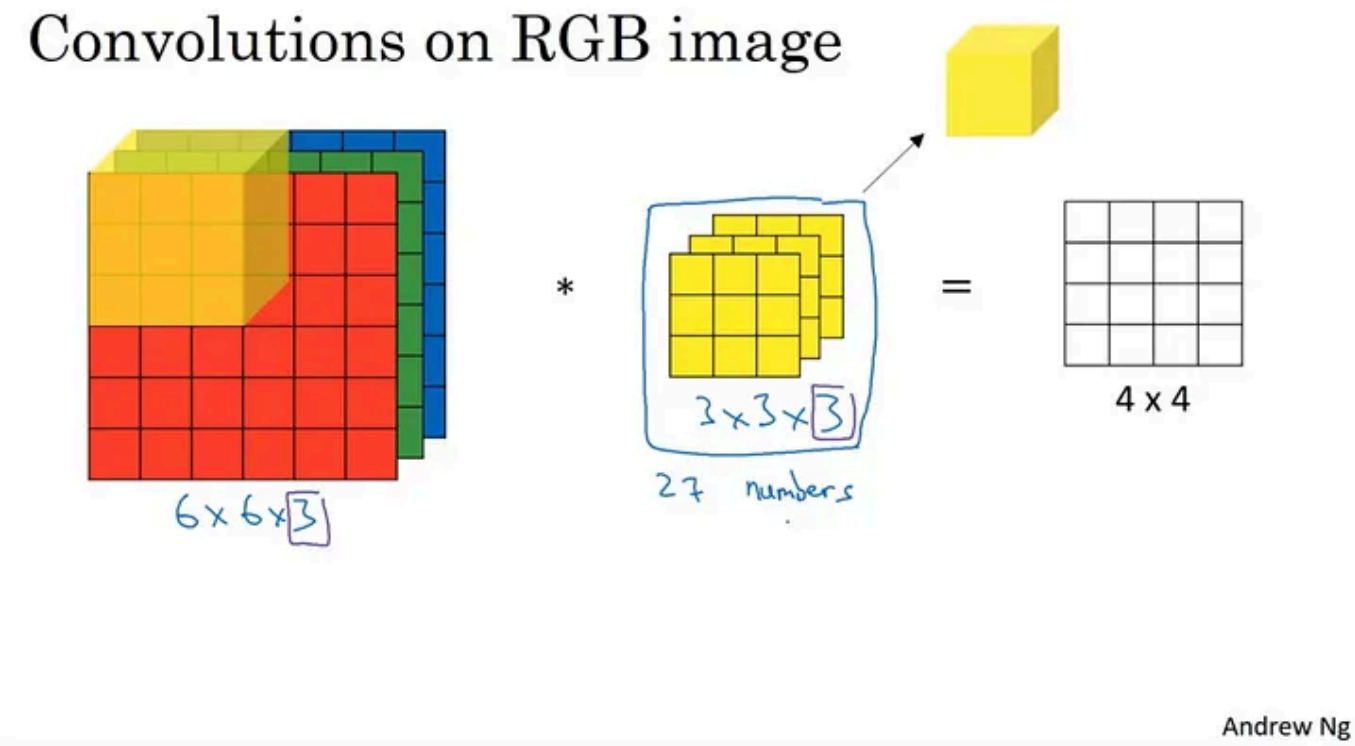
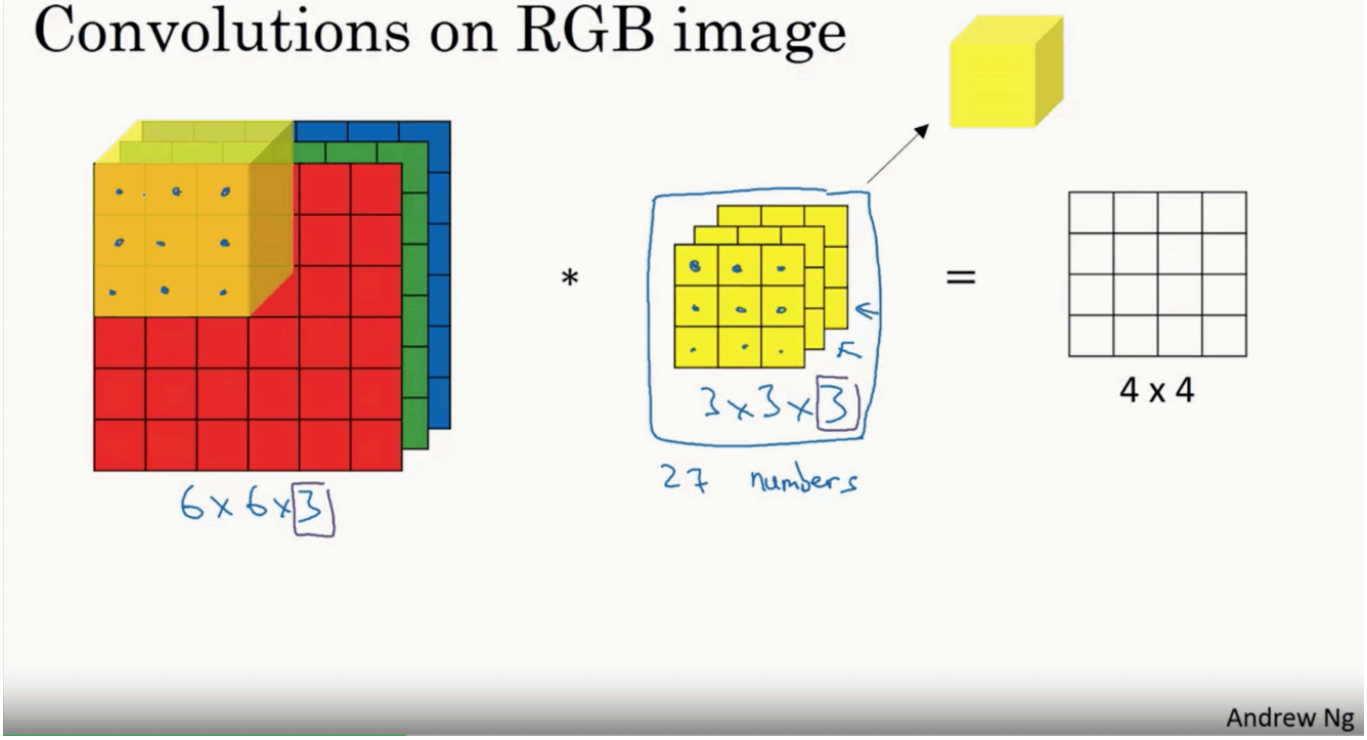


FIGURE 2: A representation of a 3D convolution.

Here's a 6 by 6 by 3 image, and a 3 by 3 by 3 filter once again shown on figure 2. Notice that the 3 by 3 by 3 filter has 27 numbers, or 27 parameters, that's three cubes.

. . .



GIF 1: An illustration of a 3D convolution

So how do you convolve this RGB image the 3D filter? What you do is take each of the 27 numbers of the filter and multiply them with the corresponding numbers from the red, green, and blue channels of the image, i.e take the first 9 numbers from red channel, then the 3 beneath it to the green channel, then the three beneath it to the blue channel, and multiply it with the corresponding 27 numbers that gets covered by this yellow cube show on the left. Then add up all those numbers and this gives you this first number in the output, and then to compute the next output you take this cube and slide it over by one, and again, due to 27 multiplications,

add up the 27 numbers, that gives you this next output, do it for the next number over, for the next position over, that gives the third output and so on.

. . .

1.2 Multiple Filters

Now that you know how to convolve on volumes, what if we didn't just want to detect vertical edges? What if we wanted to detect vertical edges and horizontal edges, maybe 45 degree edges, 70 degree edges as well, but in other words, what if you want to use multiple filters at the same time?

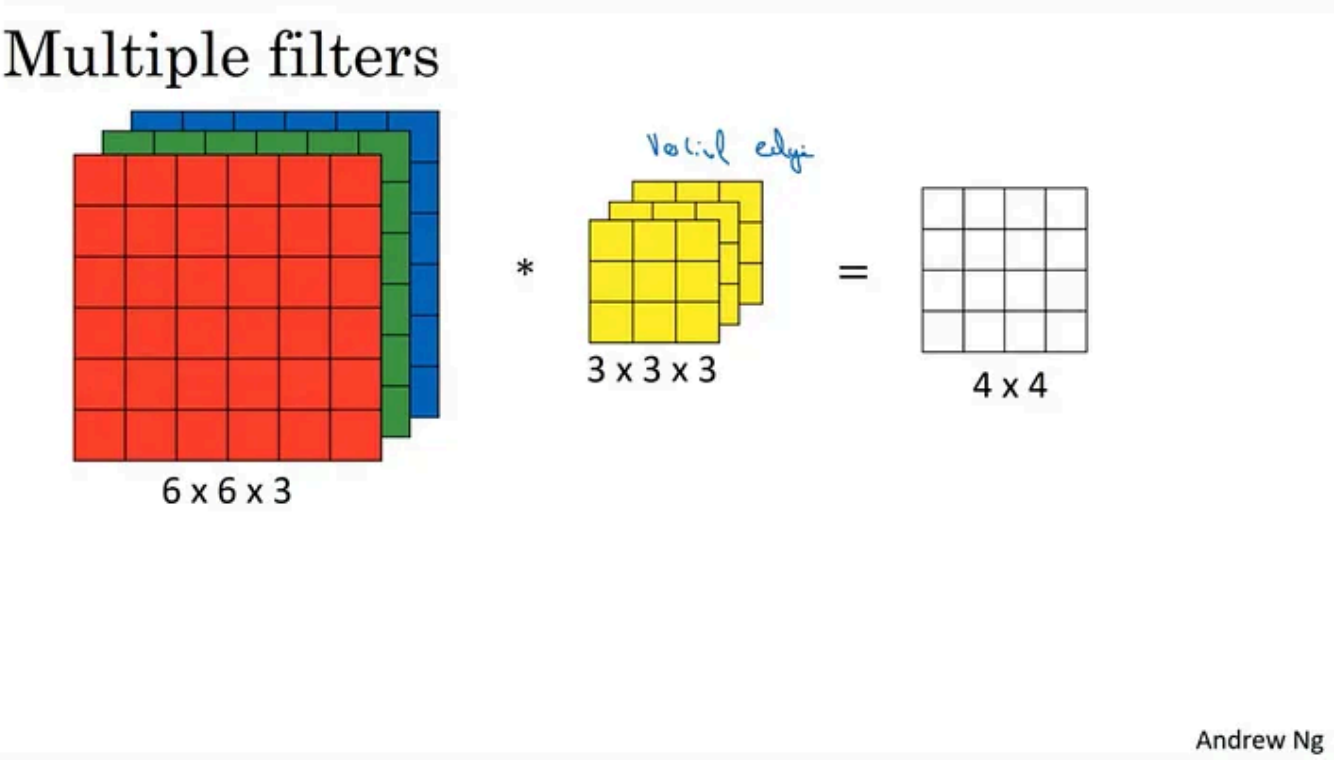


FIGURE 3: a representation of a 3D convolution that we have seen in the first section. The yellow filter is now thought of to be a vertical edge filter.

So, here's the picture we had from the previous in section 1.1 on *figure* 3. Let's say, that the yellowish 3 by 3 by 3 filter is maybe a vertical edge detector

(or maybe it's intended to detect some other feature).

. . .

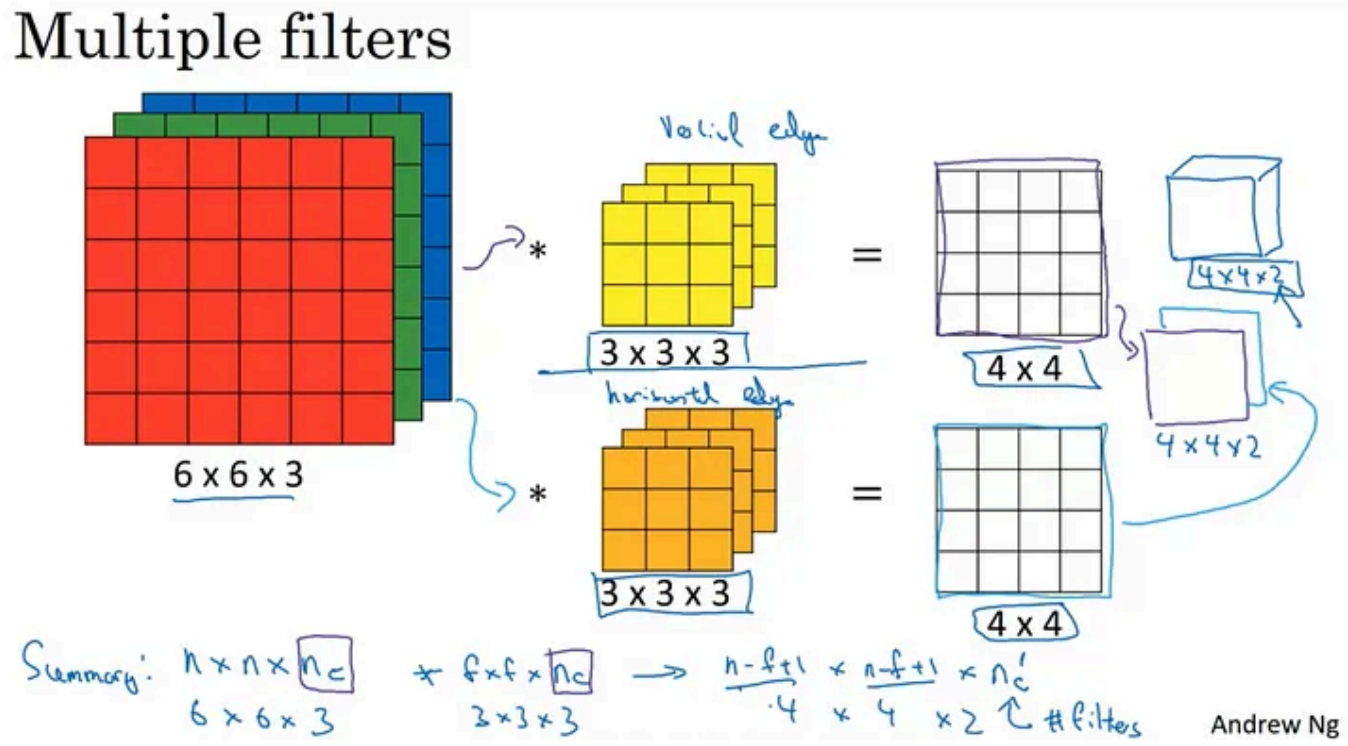


FIGURE 4: the orange filter at the bottom is thought to be a horizontal edge filter.

Now, let's say that there's a second 3 by 3 by 3 filter denoted by an orange color, is a horizontal edge detector as shown of *figure* 4. So, convolving the image with the the yellow filter gives you a 4 by 4 output and convolving with the orange filter gives you a different 4 by 4 output. Now, what we can do is then take these two 4 by 4 outputs and stack them to get a 4 by 4 by 2 output. Notice that if the image was first convolved with the yellow filter , then we take the image & yellow filter's output to be the first one (at the front) and you can then take the image & orange second filter's output and stack it at the back to end up with the 4 by 4 by 2 output image shown on the right in *figure* 4. Notice that the 2 comes from the fact that we used two different filters.

. . .

2. The Convolutional Layer

2.1 An Example

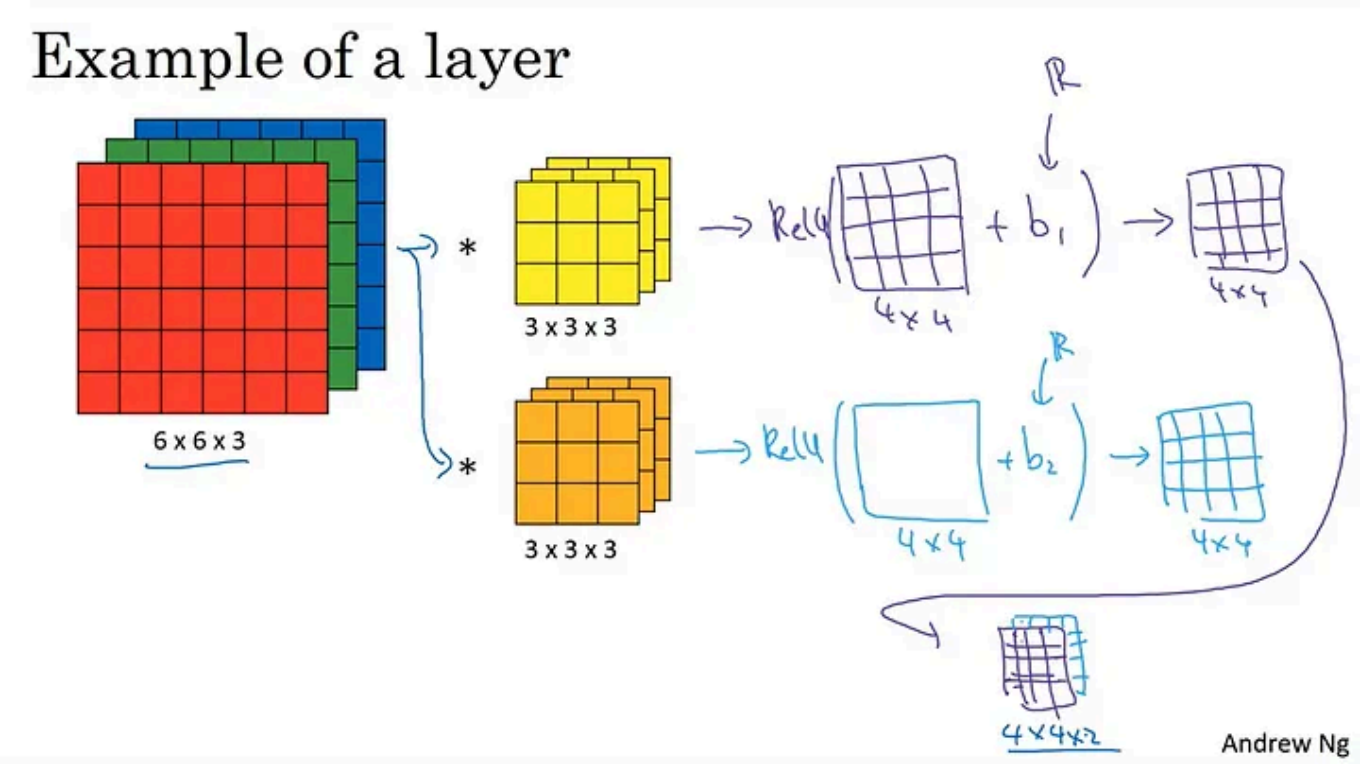


FIGURE 5: In this computation going from a 6 by 6 by 3 input image at the top left corner to a 4 by 4 by 2 output image at the bottom right corner is what's called **one layer of a convolutional network**.

The final thing to turn this into a convolutional neural net layer, is that for each of these we're going to add it bias, so this is going to be a real number. And where python broadcasting, you kind of have to add the same number so every one of these 16 elements. And then apply a non-linearity function , which for this illustration let's say a **ReLU** (**Rectifier Linear Unit**) non-linearity, and this gives you a 4 by 4 output, all right? After applying the bias and the non-linearity. We also apply the ReLU non-linearity to the the orange filter (after adding the bias) and get a different 4 by 4 output. Then same as we did before, if we take this and stack it up as follows, so we ends up with a

4 by 4 by 2 outputs. Then this computation where you come from a 6 by 6 by 3 to a 4 by 4 by 2, this is **one layer of a convolutional neural network.**

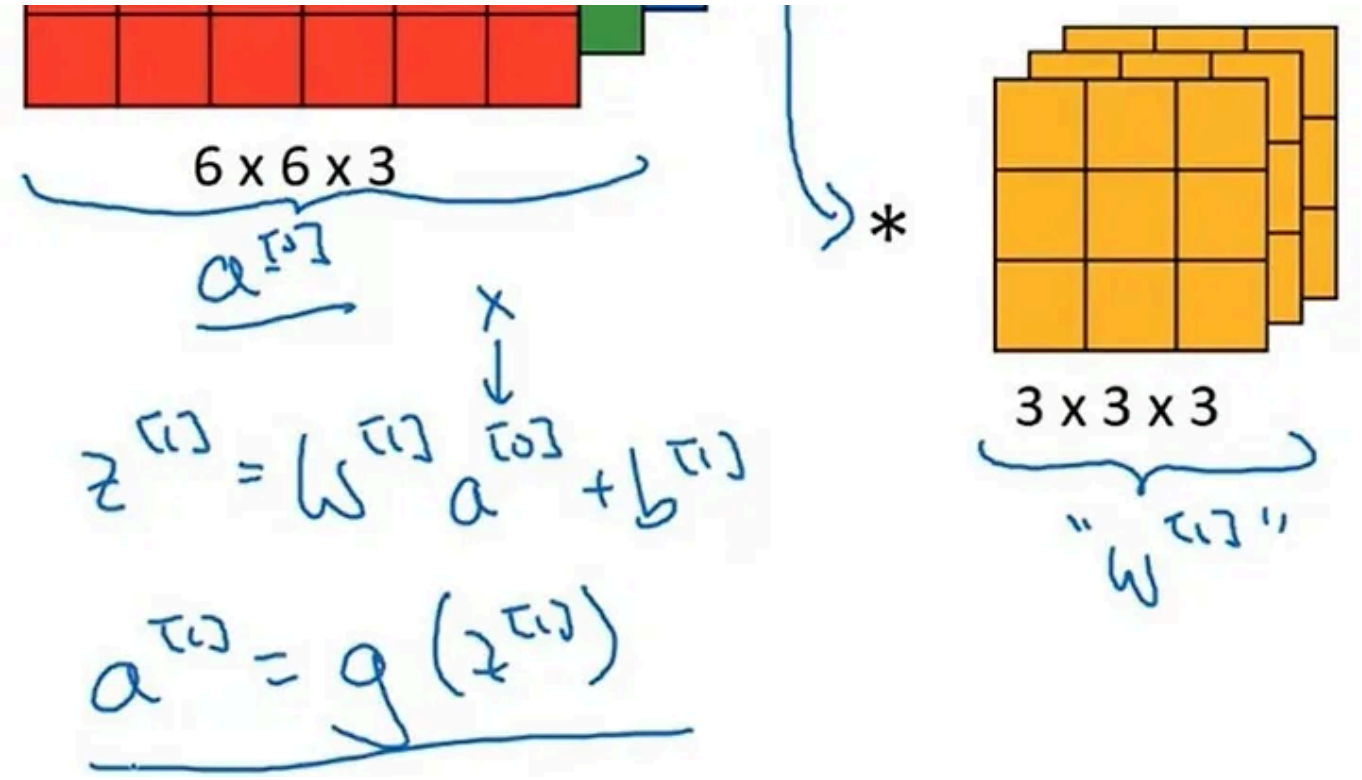


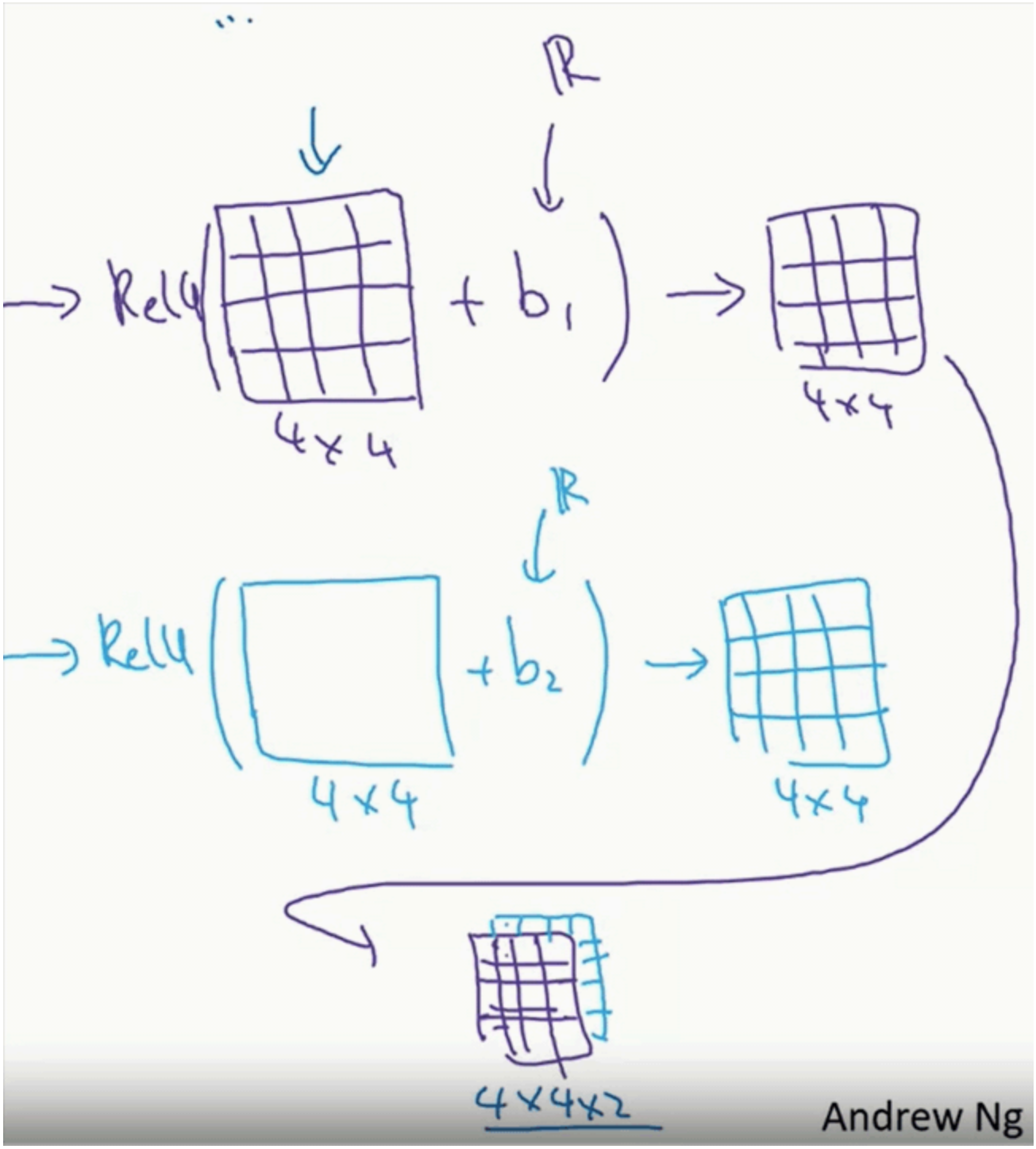
FIGURE 6: forward propagation and matrix convolution analogy.

So to map this back to one layer of forward propagation in the standard neural network, in a non-convolutional neural network. Remember that one step of forward propagation was $z^{[1]} = w^{[1]} a^{[0]} + b^{[1]}$. And you apply the non-linearity to get $a[1]$, so that's $g(z^{[1]})$ just as underlined in *figure* 6. By analogy, The input image here is $a^{[0]}$, this is $x^{[3]}$. The filters here, play a role similar to $w^{[1]}$. So you're really computing a linear function to get the 4 by 4 output matrix which is $a^{[1]} = g(z^{[1]})$.

. . .

2.2 Summary

I'll reiterate points just discussed in this section from another perspective just for clarity.



GIF 2: Illustration for clarity.

The 4 by 4 matrix output matrix, the output of the convolution operation, plays a role similar to $w^{[1]}$ $a^{[0]}$. and the output image plays a role similar

to $a^{[1]} = g(z^{[1]})$, where g is the non-linearity function such as the ReLU and $z^{[1]} = w^{[1]} a^{[0]} + b^{[1]}$. So we are really just going from $a^{[0]}$ to $a^{[1]}$, where a^[0] is the 6 by 6 by 3 input image and a^[1] is the 4 by 4 by 2 output image. See GIF 2.

Also notice that if for example we instead had 10 filters instead of 2, then we would have a 4 by 4 by 10 dimensional output volume.

. . .

Below is a summary of the notation just in case my future posts get a little confusing or if you forget what the notation means. Just come back to this image for the notation. I have also used curved lines to help you remember how these formulas are linked to each other. I think it's better to understand where such formulas come from than to just memorize their characters.