Now you know all the building blocks to create a convolutional neural network. Let's see how to assemble them.

CNN Architectures

Typical CNN architectures stack a few convolutional layers (each one generally followed by a ReLU layer), then a pooling layer, then another few convolutional layers (+ReLU), then another pooling layer, and so on. The image gets smaller and smaller as it progresses through the network, but it also typically gets deeper and deeper (i.e., with more feature maps) thanks to the convolutional layers (see Figure 13-9). At the top of the stack, a regular feedforward neural network is added, composed of a few fully connected layers (+ReLUs), and the final layer outputs the prediction (e.g., a softmax layer that outputs estimated class probabilities).

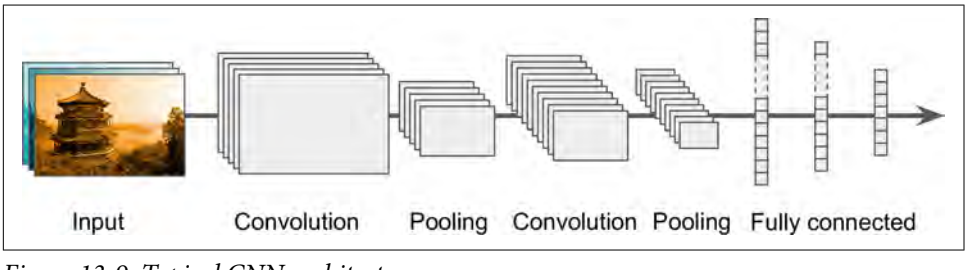

Figure 13-9. Typical CNN architecture

A common mistake is to use convolution kernels that are too large. You can often get the same effect as a 9 × 9 kernel by stacking two 3 × 3 kernels on top of each other, for a lot less compute.

Over the years, variants of this fundamental architecture have been developed, leading to amazing advances in the field. A good measure of this progress is the error rate in competitions such as the ILSVRC ImageNet challenge. In this competition the top-5 error rate for image classification fell from over 26% to barely over 3% in just five years. The top-five error rate is the number of test images for which the system's top 5 predictions did not include the correct answer. The images are large (256 pixels high) and there are 1,000 classes, some of which are really subtle (try distinguishing 120 dog breeds). Looking at the evolution of the winning entries is a good way to understand how CNNs work.

We will first look at the classical LeNet-5 architecture (1998), then three of the winners of the ILSVRC challenge: AlexNet (2012), GoogLeNet (2014), and ResNet (2015).

Other Visual Tasks

There was stunning progress as well in other visual tasks such as object detection and localization, and image segmentation. In object detection and localization, the neural network typically outputs a sequence of bounding boxes around various objects in the image. For example, see Maxine Oquab et al.'s 2015 paper that outputs a heat map for each object class, or Russell Stewart et al.'s 2015 paper that uses a combination of a CNN to detect faces and a recurrent neural network to output a sequence of bounding boxes around them. In image segmentation, the net outputs an image (usually of the same size as the input) where each pixel indicates the class of the object to which the corresponding input pixel belongs. For example, check out Evan Shelhamer et al.'s 2016 paper.

LeNet-5

The LeNet-5 architecture is perhaps the most widely known CNN architecture. As mentioned earlier, it was created by Yann LeCun in 1998 and widely used for handwritten digit recognition (MNIST). It is composed of the layers shown in Table 13-1.

Table 13-1. LeNet-5 architecture

Layer	Type	Maps	Size	Kernel size	Stride	Activation
Out	Fully Connected	–	10	–	–	RBF
F6	Fully Connected	–	84	–	–	tanh
C5	Convolution	120	1×1	5×5	1	tanh
S4	Avg Pooling	16	5×5	2×2	2	tanh
C3	Convolution	16	10×10	5×5	1	tanh
S2	Avg Pooling	6	14×14	2×2	2	tanh
C1	Convolution	6	28×28	5×5	1	tanh
In	Input	1	32×32	–	–	–

There are a few extra details to be noted:

- MNIST images are 28×28 pixels, but they are zero-padded to 32×32 pixels and normalized before being fed to the network. The rest of the network does not use any padding, which is why the size keeps shrinking as the image progresses through the network.
- The average pooling layers are slightly more complex than usual: each neuron computes the mean of its inputs, then multiplies the result by a learnable coefficient (one per map) and adds a learnable bias term (again, one per map), then finally applies the activation function.