Pooling Layers

In the previous section, we introduced the notion of convolutional kernels. These kernels apply learnable nonlinear transformations to local patches of inputs. These transformations are learnable, and by the universal approximation theorem, capable of learning arbitrarily complex input transformations on local patches. This flexibility gives convolutional kernels much of their power. But at the same time, having many learnable weights in a deep convolutional network can slow training.

Instead of using a learnable transformation, it's possible to instead use a fixed nonlinear transformation in order to reduce the computational cost of training a convolutional network. A popular fixed nonlinearity is "max pooling." Such layers select and output the maximally activating input within each local receptive patch. Figure 6-6 demonstrates this process. Pooling layers are useful for reducing the dimensionality of input data in a structured fashion. More mathematically, they take a local receptive field and replace the nonlinear activation function at each portion of the field with the max (or min or average) function.

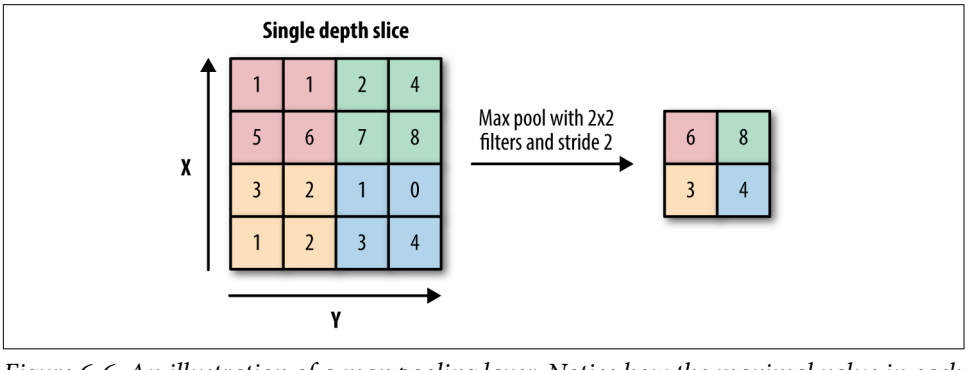

Figure 6-6. An illustration of a max pooling layer. Notice how the maximal value in each colored region (each local receptive field) is reported in the output.

Pooling layers have become less useful as hardware has improved. While pooling is still useful as a dimensionality reduction technique, recent research tends to avoid using pooling layers due to their inherent lossiness (it's not possible to back out of pooled data which pixel in the input originated the reported activation). Nonetheless, pooling appears in many standard convolutional architectures so it's worth understanding.

Constructing Convolutional Networks

A simple convolutional architecture applies a series of convolutional layers and pooling layers to its input to learn a complex function on the input image data. There are a lot of details in forming these networks, but at its heart, architecture design is sim-