tional architectures to work on new datatypes. For example, graph convolutional architectures allow convolutional networks to be applied to molecular data such as the Tox21 dataset we encountered a few chapters ago! Convolutional architectures are also making a mark in genomics and in text processing and even language translation.

In this chapter, we will introduce the basic concepts of convolutional networks. These will include the basic network components that constitute convolutional architectures and an introduction to the design principles that guide how these pieces are joined together. We will also provide an in-depth example that demonstrates how to use TensorFlow to train a convolutional network. The example code for this chapter was adapted from the TensorFlow documentation tutorial on convolutional neural networks. We encourage you to access the original tutorial on the TensorFlow website if you're curious about the changes we've made. As always, we encourage you to work through the scripts for this chapter in the associated GitHub repo for this book.

Introduction to Convolutional Architectures

Most convolutional architectures are made up of a number of basic primitives. These primitives include layers such as convolutional layers and pooling layers. There's also a set of associated vocabulary including local receptive field size, stride size, and number of filters. In this section, we will give you a brief introduction to the basic vocabulary and concepts underlying convolutional networks.

Local Receptive Fields

The local receptive field concept originates in neuroscience, where the receptive field of a neuron is the part of the body's sensory perception that affects the neuron's firing. Neurons have a certain field of "view" as they process sensory input that the brain sees. This field of view is traditionally called the local receptive field. This "field of view" could correspond to a patch of skin or to a segment of a person's visual field. Figure 6-1 illustrates a neuron's local receptive field.

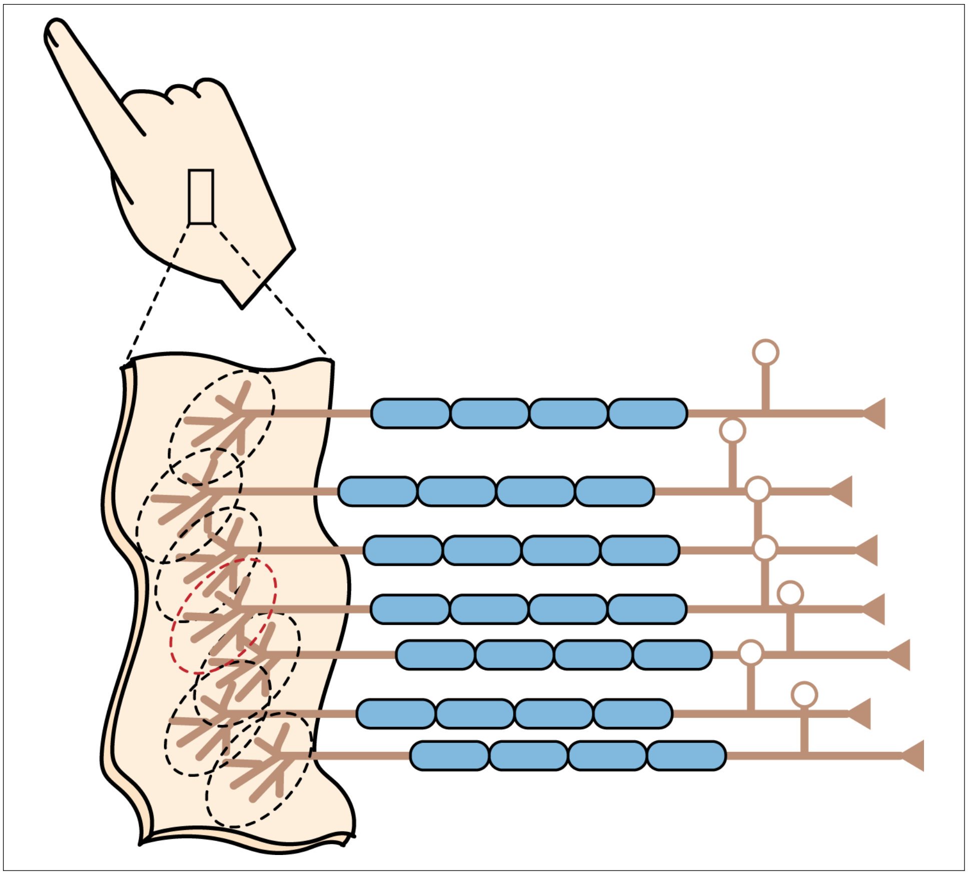

Figure 6-1. An illustration of a neuron's local receptive field.

Convolutional architectures borrow this latter concept with the computational notion of "local receptive fields." Figure 6-2 provides a pictorial representation of the local receptive field concept applied to image data. Each local receptive field corresponds to a patch of pixels in the image and is handled by a separate "neuron." These "neurons" are directly analogous to those in fully connected layers. As with fully connected layers, a nonlinear transformation is applied to incoming data (which originates from the local receptive image patch).

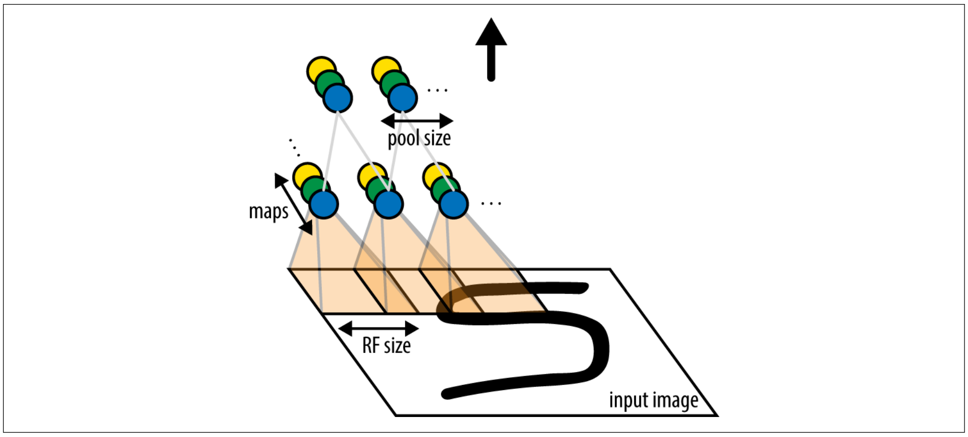

Figure 6-2. The local receptive field (RF) of a "neuron" in a convolutional network.

A layer of such "convolutional neurons" can be combined into a convolutional layer. This layer can viewed as a transformation of one spatial region into another. In the case of images, one batch of images is transformed into another by a convolutional layer. Figure 6-3 illustrates such a transformation. In the next section, we will show you more details about how a convolutional layer is constructed.

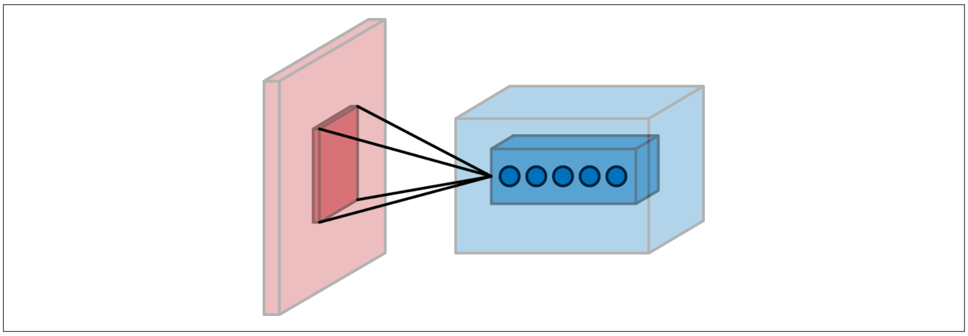

Figure 6-3. A convolutional layer performs an image transformation.

It's worth emphasizing that local receptive fields don't have to be limited to image data. For example, in stacked convolutional architectures, where the output of one convolutional layer feeds into the input of the next, the local receptive field will correspond to a "patch" of processed feature data.

Convolutional Kernels

In the last section, we mentioned that a convolutional layer applies nonlinear function to a local receptive field in its input. This locally applied nonlinearity is at the heart of convolutional architectures, but it's not the only piece. The second part of the