Chpt8 Using convolutions to generalize

Sep 5, 2021

Covers:

1. Understanding convolution
2. Building a convolutional neural network
3. Creating custom nn.Module subclasses
4. Design choices for neural network

Due to the fully connected setup needed to detect the various possible translations of the bird or airplane in the image, we have both too many parameters and no position independence. As discussed in the last chapter, we could argument our training data by using a wide variety of recropped images to try to force generalization, but that won't address the issue of having too many parameters. There is a better way. It consists of replacing the dense, fully connected affine transformation in our neural network unit with a different linear operation: convolution.

8.1 The case for convolutions

In this section, we'll see how convolutions deliver locality and translation invariance. We could compute the weighted sum of a pixel with its intermediate neighbors. This would be equivalent to building weight matrices, one per output feature and output pixel location, in which all weights beyond the certain distance are zero. This will still be a weighted sum: that is, a linear operation.

We would like these localized patterns to have an effect on the output regardless of their location in the image: that is, to be **translation invariant**.

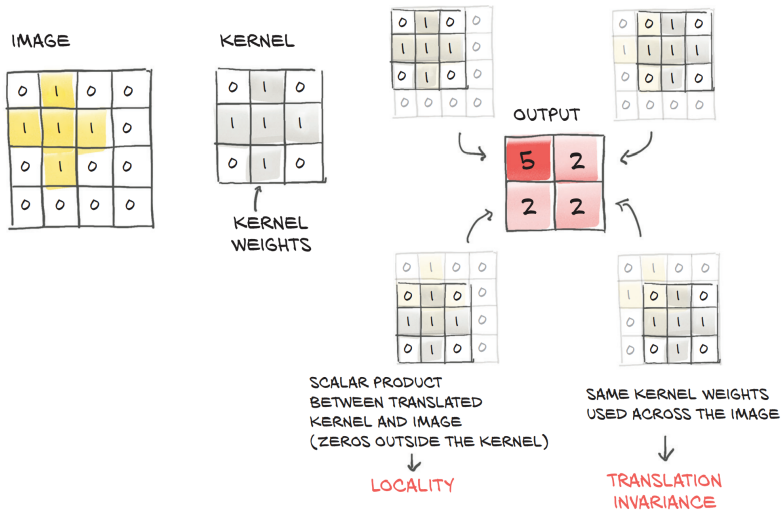

Figure 8.1 Convolution: locality and translation invariance