VARIANTS OF THE BASIC CONVOLUTION FUNCTION

Convolution in the context of NN means an operation that consists of many applications of convolution in parallel.

- Kernel K with element giving the connection strength between a unit in channel i of output and a unit in channel j of the input, with an offset of k rows and l columns between the output unit and the input unit.
- Input: with channel i, row j and column k
- Output Z same format as V
- Use 1 as first entry

Full Convolution

0 Padding 1 stride

Unshared convolution. Indices into weight W

- i: the output channel
- j: the output row;
- k: the output column
- l: the input channel
- m: row offset within input

- n: column offset within input

$Z_{i,j,k} = \sum_{l,m,n} [V_{l,i+m-1,j+n-1} W_{i,j,k,l,m,n}]$

Comparison on local connections, convolution and full connection

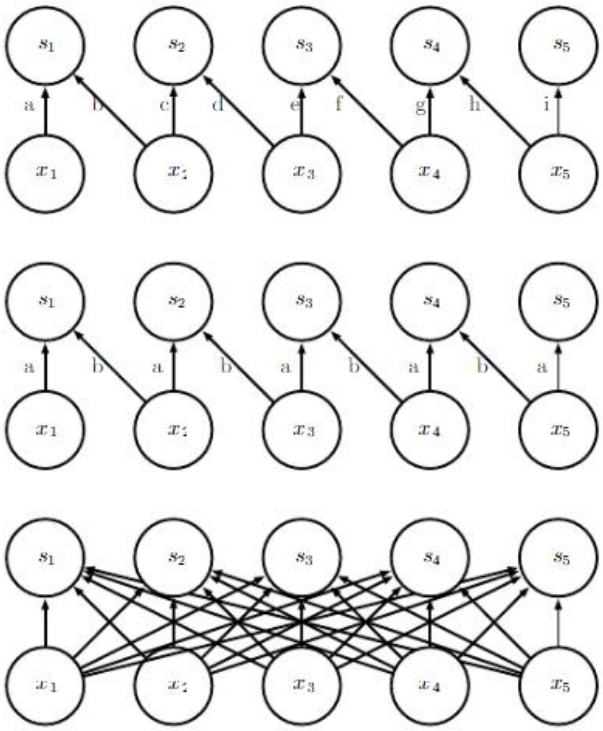

Figure 9.14: Comparison of local connections, convolution, and full connections. *(Top)*A locally connected layer with a patch size of two pixels. Each edge is labeled with a unique letter to show that each edge is associated with its own weight parameter. *(Center)*A convolutional layer with a kernel width of two pixels. This model has exactly the same connectivity as the locally connected layer. The difference lies not in which units interact with each other, but in how the parameters are shared. The locally connected layer has no parameter sharing. The convolutional layer uses the same two weights repeatedly across the entire input, as indicated by the repetition of the letters labeling each edge. *(Bottom)*A fully connected layer resembles a locally connected layer in the sense that each edge has its own parameter (there are too many to label explicitly with letters in this diagram). It does not, however, have the restricted connectivity of the locally connected layer.

Useful when we know that each feature should be a function of a small part of space, but no reason to think that the same feature should occur accross all the space. eg: look for mouth only in the bottom half of the image.

It can be also useful to make versions of convolution or local connected layers in which the connectivity is further restricted, eg: constrain each output channeel i to be a function of only a subset of the input channel.

Adv: * reduce memory consumption * increase statistical efficiency * reduce computation for both forward and backward prop.

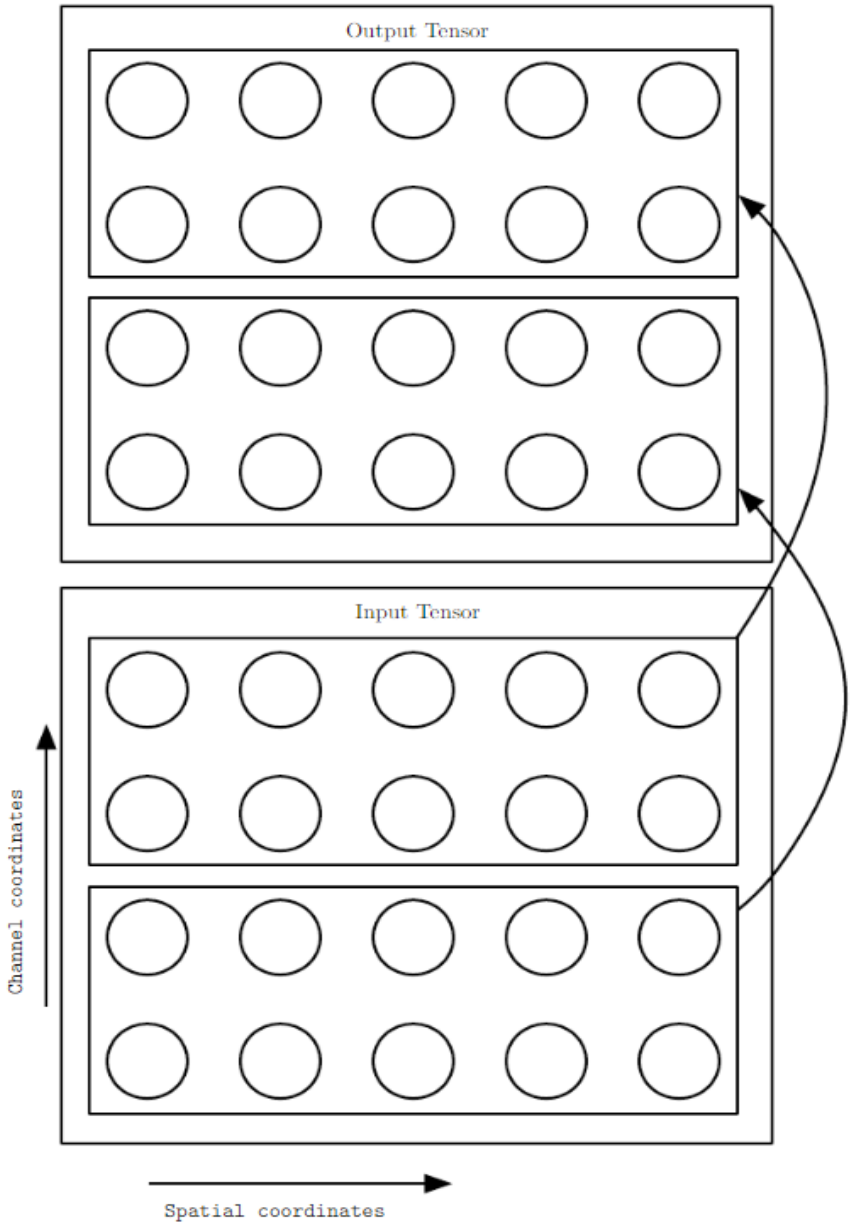

Figure 9.15: A convolutional network with the first two output channels connected to only the first two input channels, and the second two output channels connected to only the second two input channels.

Tiled Convolution

Learn a set of kernels that we rotate through as we move through space. Immediately neighboring locations will have different filters, but the memory requirement for storing the parameters will increase by a factor of the size of this set of kernels. Comparison on locally connected layers, tiled convolution and stardard convolution:

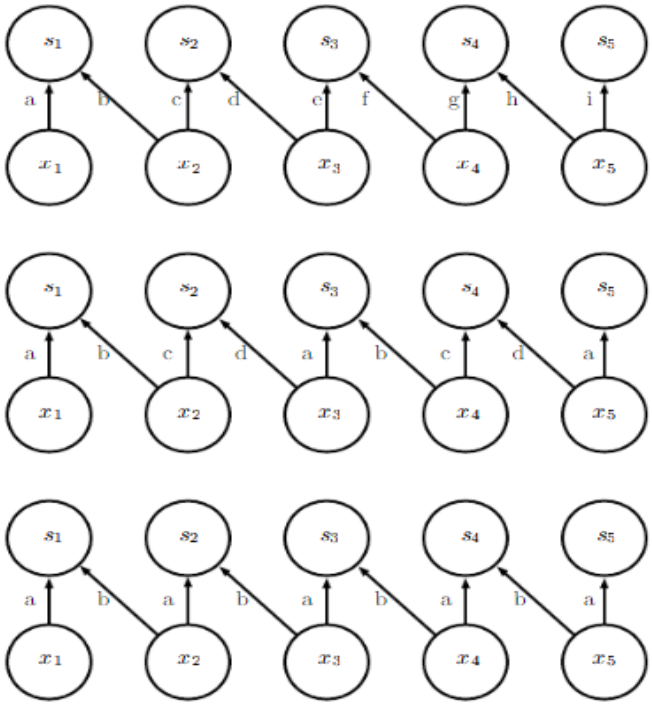

Figure 9.16: A comparison of locally connected layers, tiled convolution, and standard convolution. All three have the same sets of connections between units, when the same size of kernel is used. This diagram illustrates the use of a kernel that is two pixels wide. The differences between the methods lies in how they share parameters. *(Top)* A locally connected layer has no sharing at all. We indicate that each connection has its own weight by labeling each connection with a unique letter. *(Center)* Tiled convolution has a set of t different kernels. Here we illustrate the case of $t = 2$. One of these kernels has edges labeled "a" and "b," while the other has edges labeled "c" and "d." Each time we move one pixel to the right in the output, we move on to using a different kernel. This means that, like the locally connected layer, neighboring units in the output have different parameters. Unlike the locally connected layer, after we have gone through all t available kernels, we cycle back to the first kernel. If two output units are separated by a multiple of t steps, then they share parameters. *(Bottom)* Traditional convolution is equivalent to tiled convolution with $t = 1$. There is only one kernel, and it is applied everywhere, as indicated in the diagram by using the kernel with weights labeled "a" and "b" everywhere.

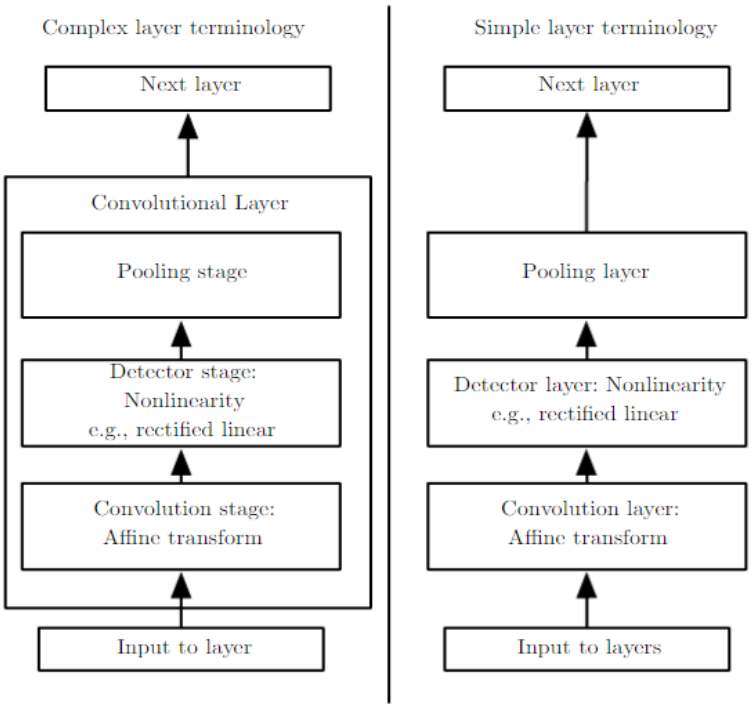

Figure 9.7: The components of a typical convolutional neural network layer. There are two commonly used sets of terminology for describing these layers. *(Left)*In this terminology, the convolutional net is viewed as a small number of relatively complex layers, with each layer having many "stages." In this terminology, there is a one-to-one mapping between kernel tensors and network layers. In this book we generally use this terminology. *(Right)*In this terminology, the convolutional net is viewed as a larger number of simple layers; every step of processing is regarded as a layer in its own right. This means that not every "layer" has parameters.

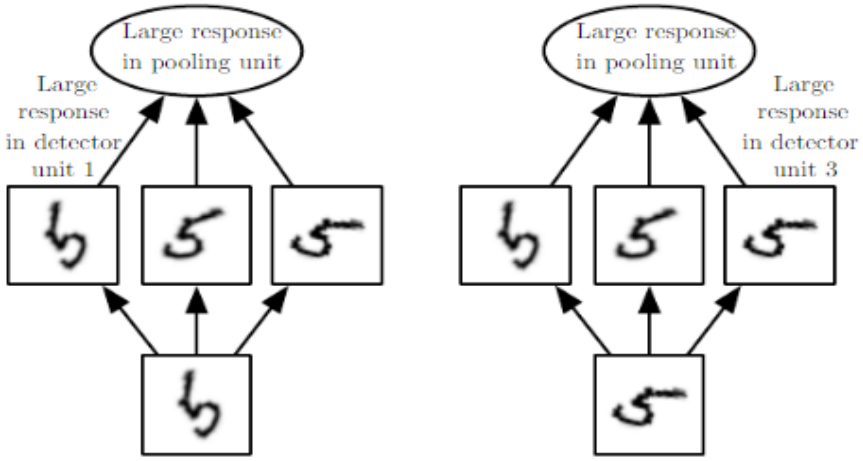

Figure 9.9: Example of learned invariances. A pooling unit that pools over multiple features that are learned with separate parameters can learn to be invariant to transformations of the input. Here we show how a set of three learned filters and a max pooling unit can learn to become invariant to rotation. All three filters are intended to detect a hand written 5. Each filter attempts to match a slightly different orientation of the 5. When a 5 appears in the input, the corresponding filter will match it and cause a large activation in a detector unit. The max pooling unit then has a large activation regardless of which detector unit was activated. We show here how the network processes two different inputs, resulting in two different detector units being activated. The effect on the pooling unit is roughly the same either way. This principle is leveraged by maxout networks (Goodfellow *et al.*, 2013a) and other convolutional networks. Max pooling over spatial positions is naturally invariant to translation; this multichannel approach is only necessary for learning other transformations.