ML Practicum: Image Classification

Introducing Convolutional Neural Networks

A breakthrough in building models for image classification came with the discovery that a convolutional neural network (https://wikipedia.org/wiki/Convolutional_neural_network) (CNN) could be used to progressively extract higher- and higher-level representations of the image content. Instead of preprocessing the data to derive features like textures and shapes, a CNN takes just the image's raw pixel data as input and "learns" how to extract these features, and ultimately infer what object they constitute.

To start, the CNN receives an input feature map: a three-dimensional matrix where the size of the first two dimensions corresponds to the length and width of the images in pixels. The size of the third dimension is 3 (corresponding to the 3 channels of a color image: red, green, and blue). The CNN comprises a stack of modules, each of which performs three operations.

1. Convolution

A *convolution* extracts tiles of the input feature map, and applies filters to them to compute new features, producing an output feature map, or *convolved feature* (which may have a different size and depth than the input feature map). Convolutions are defined by two parameters:

- **Size of the tiles that are extracted** (typically 3x3 or 5x5 pixels).
- **The depth of the output feature map**, which corresponds to the number of filters that are applied.

During a convolution, the filters (matrices the same size as the tile size) effectively slide over the input feature map's grid horizontally and vertically, one pixel at a time, extracting each corresponding tile (see Figure 3).

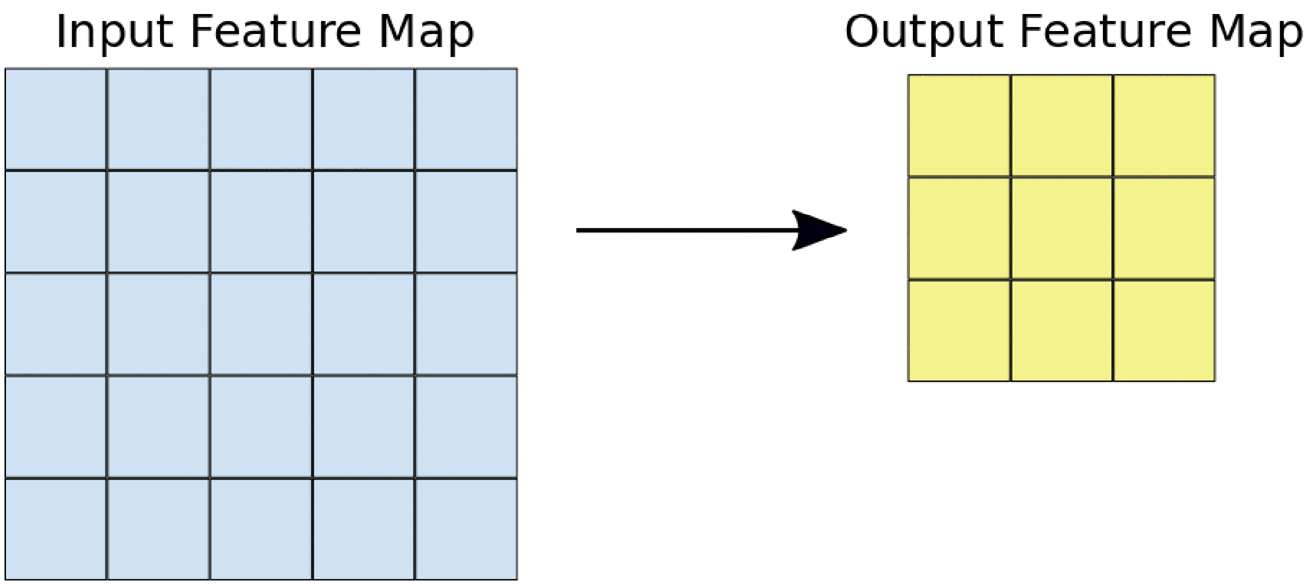

Figure 3. A 3x3 convolution of depth 1 performed over a 5x5 input feature map, also of depth 1. There are nine possible 3x3 locations to extract tiles from the 5x5 feature map, so this convolution produces a 3x3 output feature map.

In Figure 3, the output feature map (3x3) is smaller than the input feature map (5x5). If you instead want the output feature map to have the same dimensions as the input feature map, you can add *padding* (blank rows/columns with all-zero values) to each side of the input feature map, producing a 7x7 matrix with 5x5 possible locations to extract a 3x3 tile.

For each filter-tile pair, the CNN performs element-wise multiplication of the filter matrix and the tile matrix, and then sums all the elements of the resulting matrix to get a single value. Each of these resulting values for every filter-tile pair is then output in the *convolved feature* matrix (see Figures 4a and 4b).

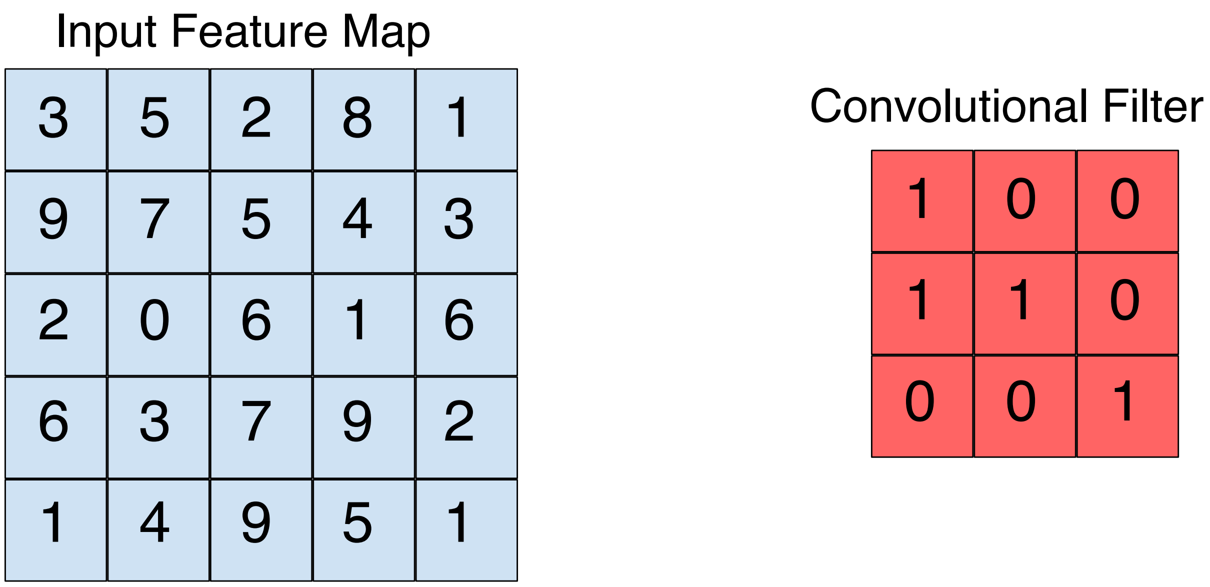

Figure 4a. ***Left****: A 5x5 input feature map (depth 1).* ***Right****: a 3x3 convolution (depth 1).*

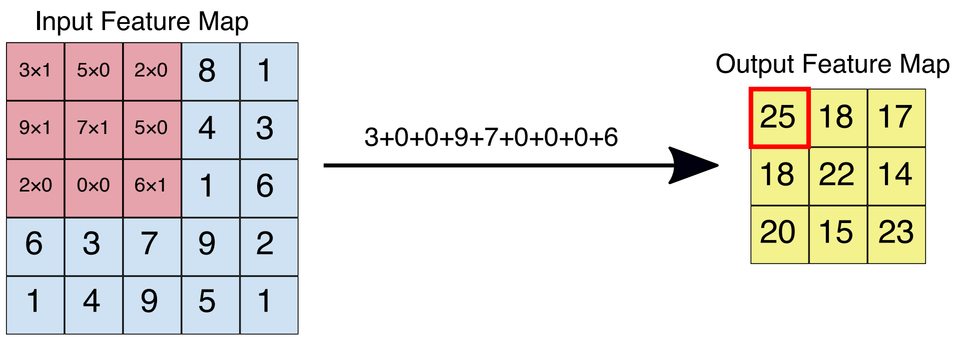

Figure 4b. ***Left****: The 3x3 convolution is performed on the 5x5 input feature map.* ***Right****: the resulting convolved feature. Click on a value in the output feature map to see how it was calculated.*

During training, the CNN "learns" the optimal values for the filter matrices that enable it to extract meaningful features (textures, edges, shapes) from the input feature map. As the number of filters (output feature map depth) applied to the input increases, so does the number of features the CNN can extract. However, the tradeoff is that filters compose the majority of resources expended by the CNN, so training time also increases as more filters are added. Additionally, each filter added to the network provides less incremental value than the previous one, so engineers aim to construct networks that use the minimum number of filters needed to extract the features necessary for accurate image classification.

2. ReLU

Following each convolution operation, the CNN applies a Rectified Linear Unit (ReLU) transformation to the convolved feature, in order to introduce nonlinearity into the model. The ReLU function, $F(x) = max(0, x)$, returns *x* for all values of *x* > 0, and returns 0 for all values of *x* ≤ 0.

ReLU is used as an activation function in a variety of neural networks; for more background, see Introduction to Neural Networks (https://developers.google.com/machine-learning/crash-course/introduction-to-neural-networks/) in Machine Learning Crash Course (https://developers.google.com/machine-learning/crash-course/).

3. Pooling

After ReLU comes a pooling step, in which the CNN downsamples the convolved feature (to save on processing time), reducing the number of dimensions of the feature map, while still preserving the most critical feature information. A common algorithm used for this process is called max pooling (https://wikipedia.org/wiki/Convolutional_neural_network#Pooling_layer).

Max pooling operates in a similar fashion to convolution. We slide over the feature map and extract tiles of a specified size. For each tile, the maximum value is output to a new feature map, and all other values are discarded. Max pooling operations take two parameters:

- **Size** of the max-pooling filter (typically 2x2 pixels)
- **Stride**: the distance, in pixels, separating each extracted tile. Unlike with convolution, where filters slide over the feature map pixel by pixel, in max pooling, the stride determines the locations where each tile is extracted. For a 2x2 filter, a stride of 2 specifies that the max pooling operation will extract all nonoverlapping 2x2 tiles from the feature map (see Figure 5).

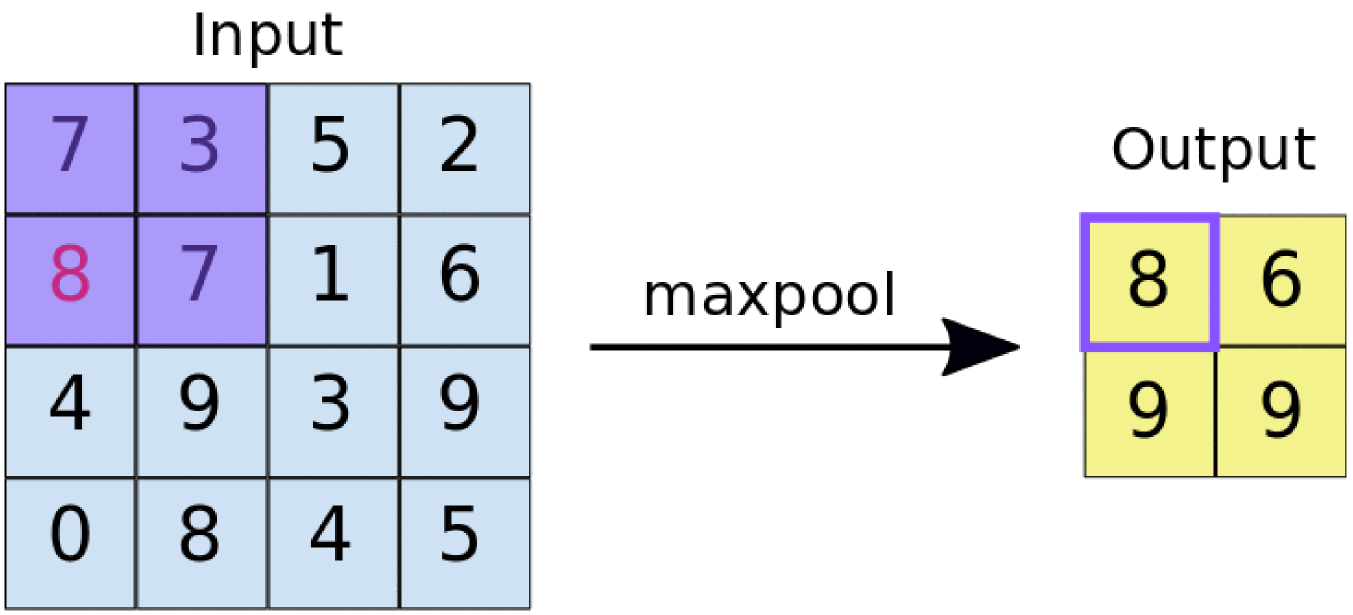

*Figure 5. **Left**: Max pooling performed over a 4x4 feature map with a 2x2 filter and stride of 2. **Right**: the output of the max pooling operation. Note the resulting feature map is now 2x2, preserving only the maximum values from each tile.*

Fully Connected Layers

At the end of a convolutional neural network are one or more fully connected layers (when two layers are "fully connected," every node in the first layer is connected to every node in the second layer). Their job is to perform classification based on the features extracted by the convolutions. Typically, the final fully connected layer contains a softmax activation function, which outputs a probability value from 0 to 1 for each of the classification labels the model is trying to predict.

For more on softmax and multi-class classification, see Multi-Class Neural Networks (https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/) in Machine Learning Crash Course (https://developers.google.com/machine-learning/crash-course/).

Figure 6 illustrates the end-to-end structure of a convolutional neural network.

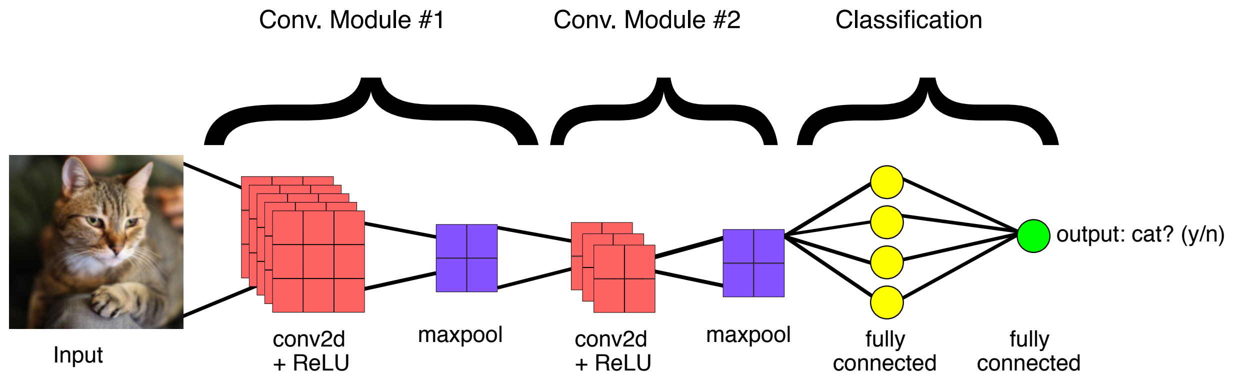

Figure 6. The CNN shown here contains two convolution modules (convolution + ReLU + pooling) for feature extraction, and two fully connected layers for classification. Other CNNs may contain larger or smaller numbers of convolutional modules, and greater or fewer fully connected layers. Engineers often experiment to figure out the configuration that produces the best results for their model.

Key Terms

- convolutional filter (https://developers.google.com/machine-learning/glossary#convolutional_filter)
- convolutional operation (https://developers.google.com/machine-learning/glossary#convolutional_operation)
- ReLU (https://developers.google.com/machine-learning/glossary#ReLU)
- convolutional neural network (https://developers.google.com/machine-learning/glossary#convolutional_neural_network)
- pooling (https://developers.google.com/machine-learning/glossary#pooling)
- stride (https://developers.google.com/machine-learning/glossary#stride)

Previous
← Introduction (https://developers.google.com/machine-learning/practica/image-classification)

Next
Check Your Understanding: Convolution → (https://developers.google.com/machine-learning/practica/image-classification/check-your-understanding)