POOLING

MUKESH KUMAR

POOLING

- The primary purpose of pooling layers in Convolutional Neural Networks (CNNs) is to reduce the spatial dimensions of the feature maps produced by convolutional layers while maintaining the most relevant information.

- This reduction in spatial dimensions helps to:
  - Increase Efficiency
  - Introduce Translation Invariance
  - Reduce Overfitting
  - Extract More Generalized Features

Increase Efficiency

- Pooling layers reduce the spatial dimensions of feature maps, leading to faster computation and lower memory requirements, making the network more efficient.

Introduce Translation Invariance

- Pooling layers make the network more robust to small translations in the input image by reducing the impact of precise feature positions. This helps the network to focus on more general features rather than specific positions.

Reduce Overfitting

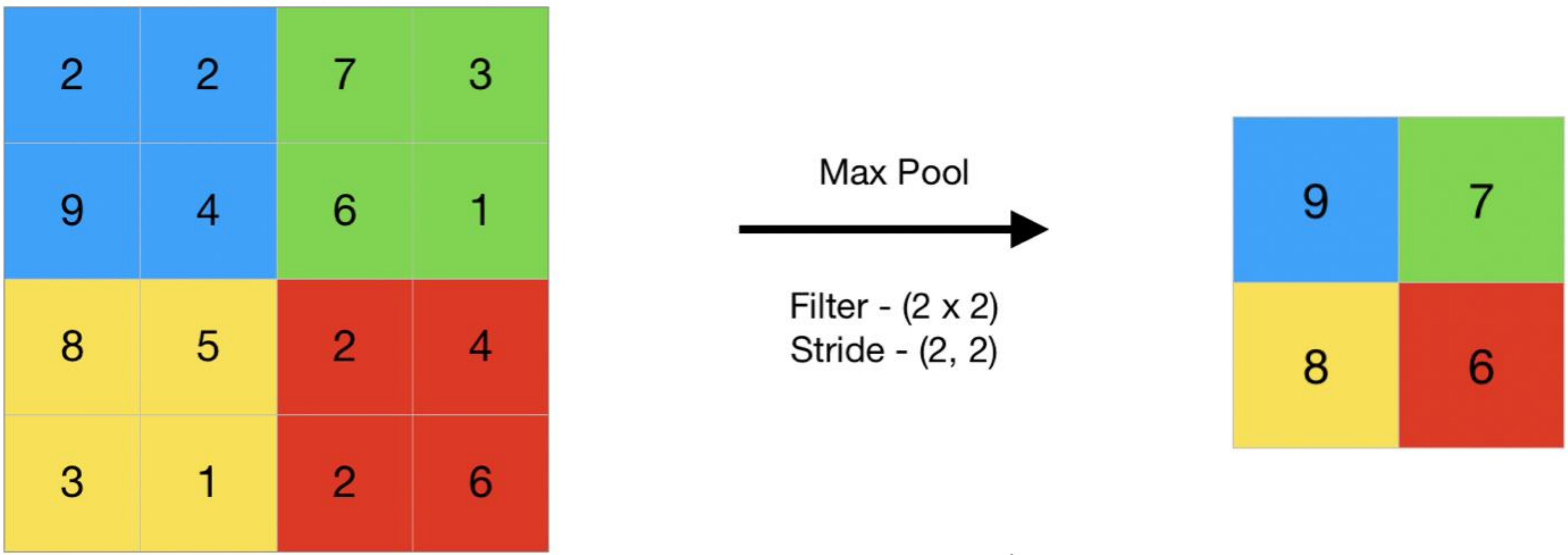
- By decreasing the number of parameters to learn and the amount of computation performed, pooling layers mitigate the risk of overfitting, which can occur when a model is too complex for the training data.

Extract More Generalized Features

- By downsampling the feature maps, pooling layers allow the network to learn more abstract and generalized features that are less sensitive to changes in the input image's illumination, orientation, or perspective.

Types of Pooling

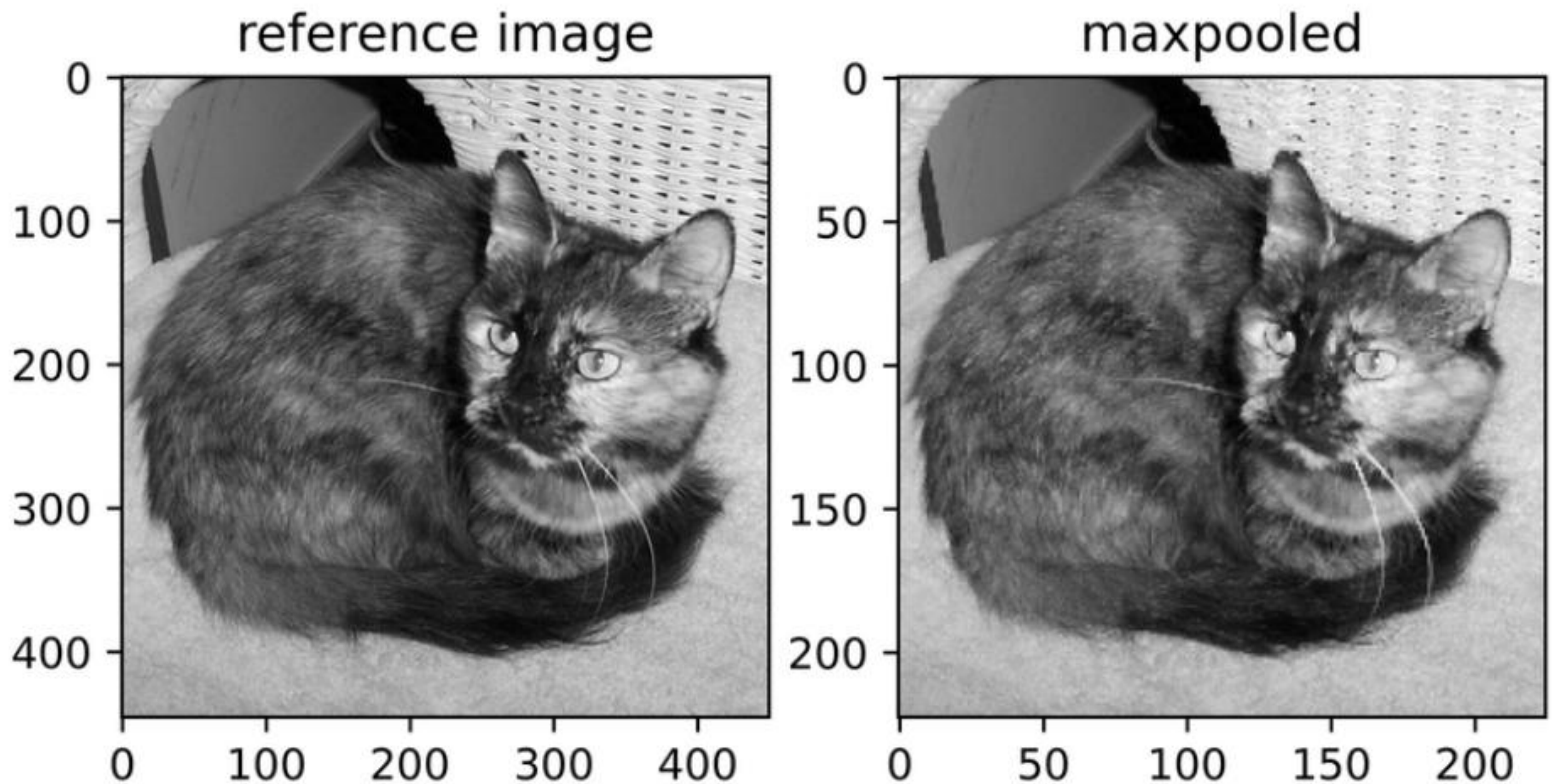
Max Pooling



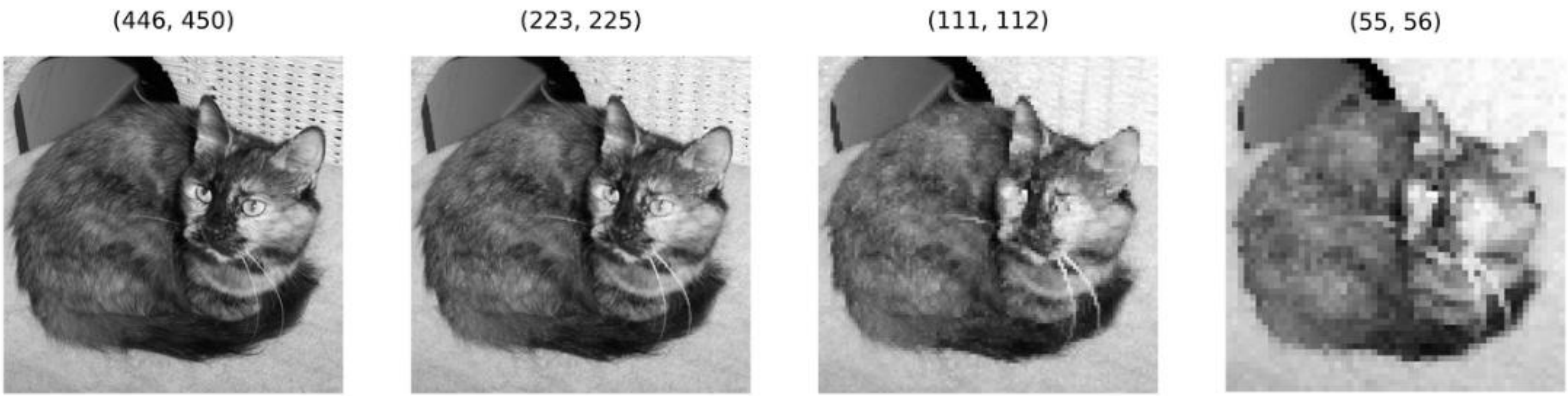
TensorFlow > API > TensorFlow v2.16.1 > Python

tf.keras.layers.MaxPool2D

MaxPooled

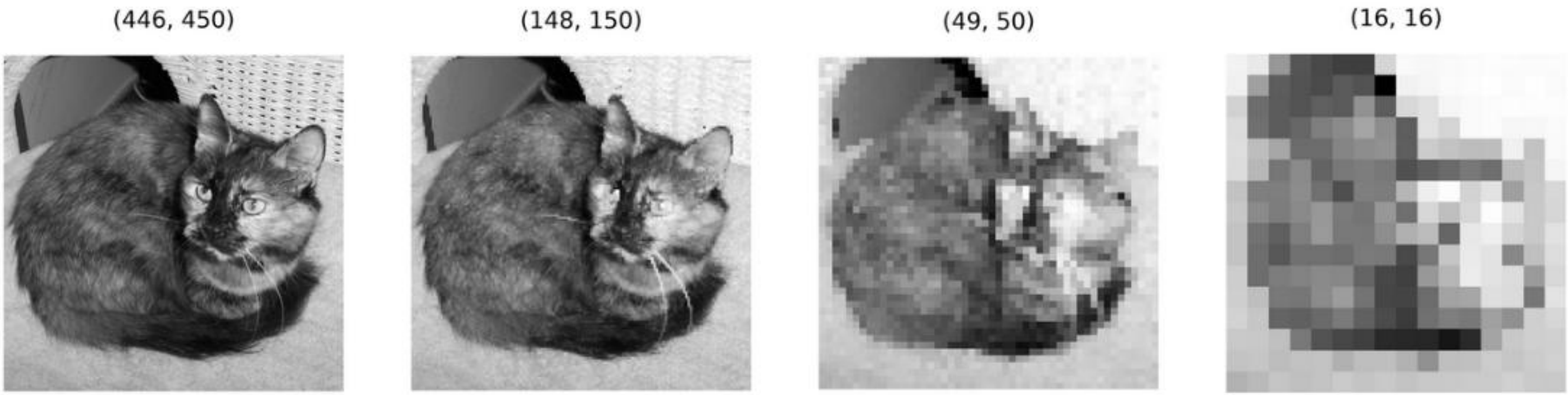


MaxPooled

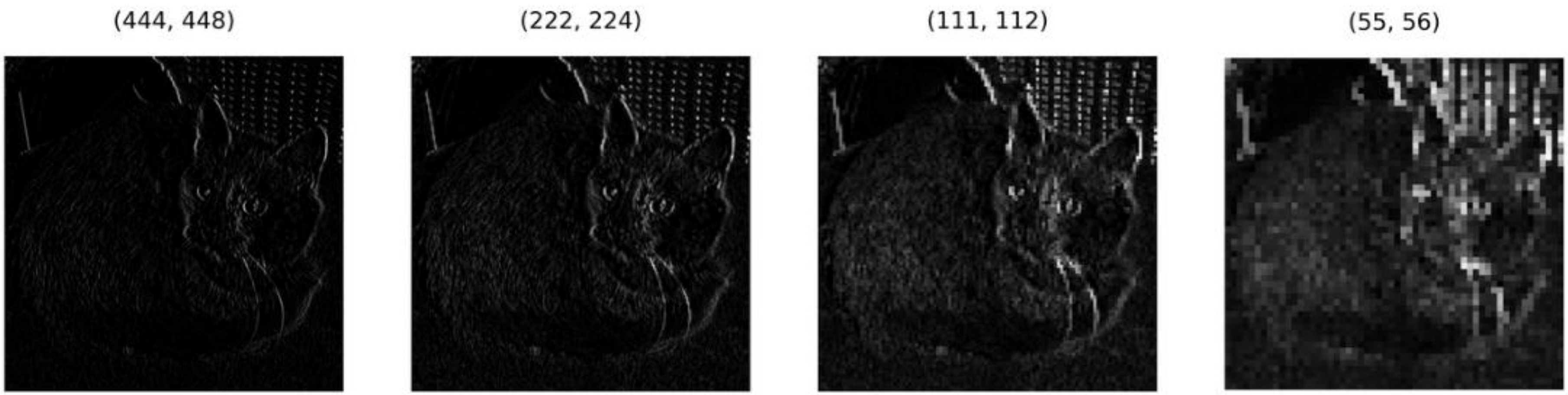


Reference image through 3 progressive iterations of max pooling using a (2, 2) kernel.

MaxPooled

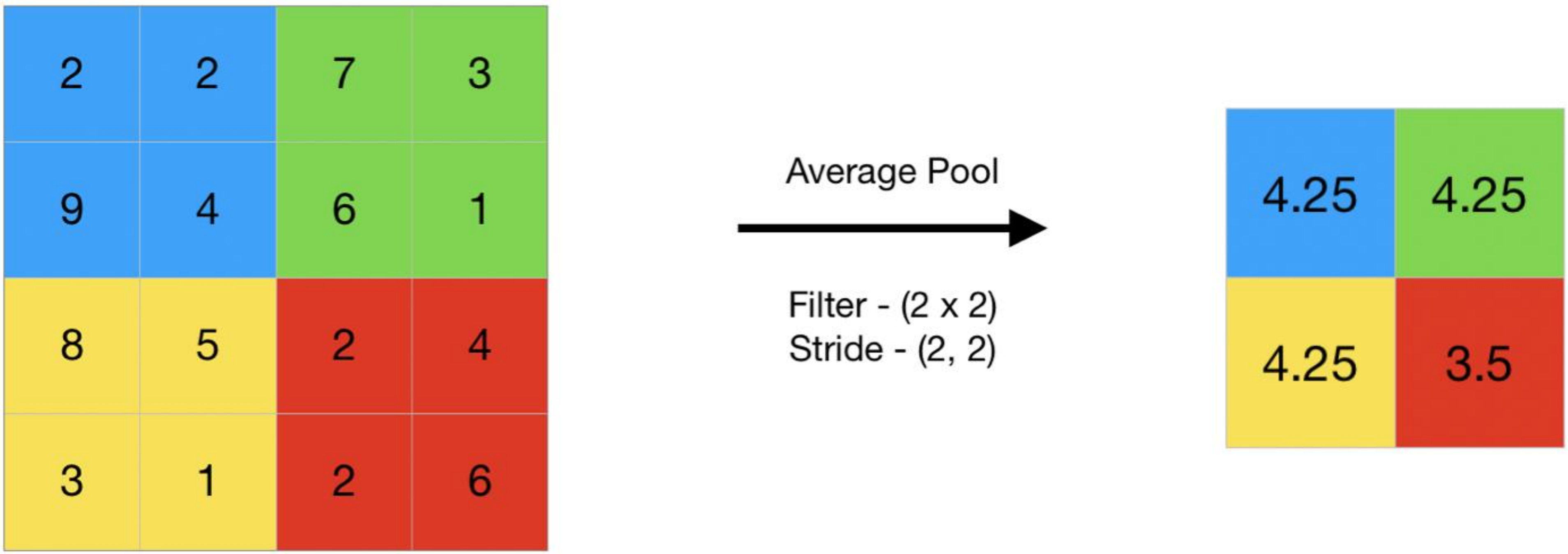


Reference image through 3 iterations of max pooling using a (3, 3) kernel.



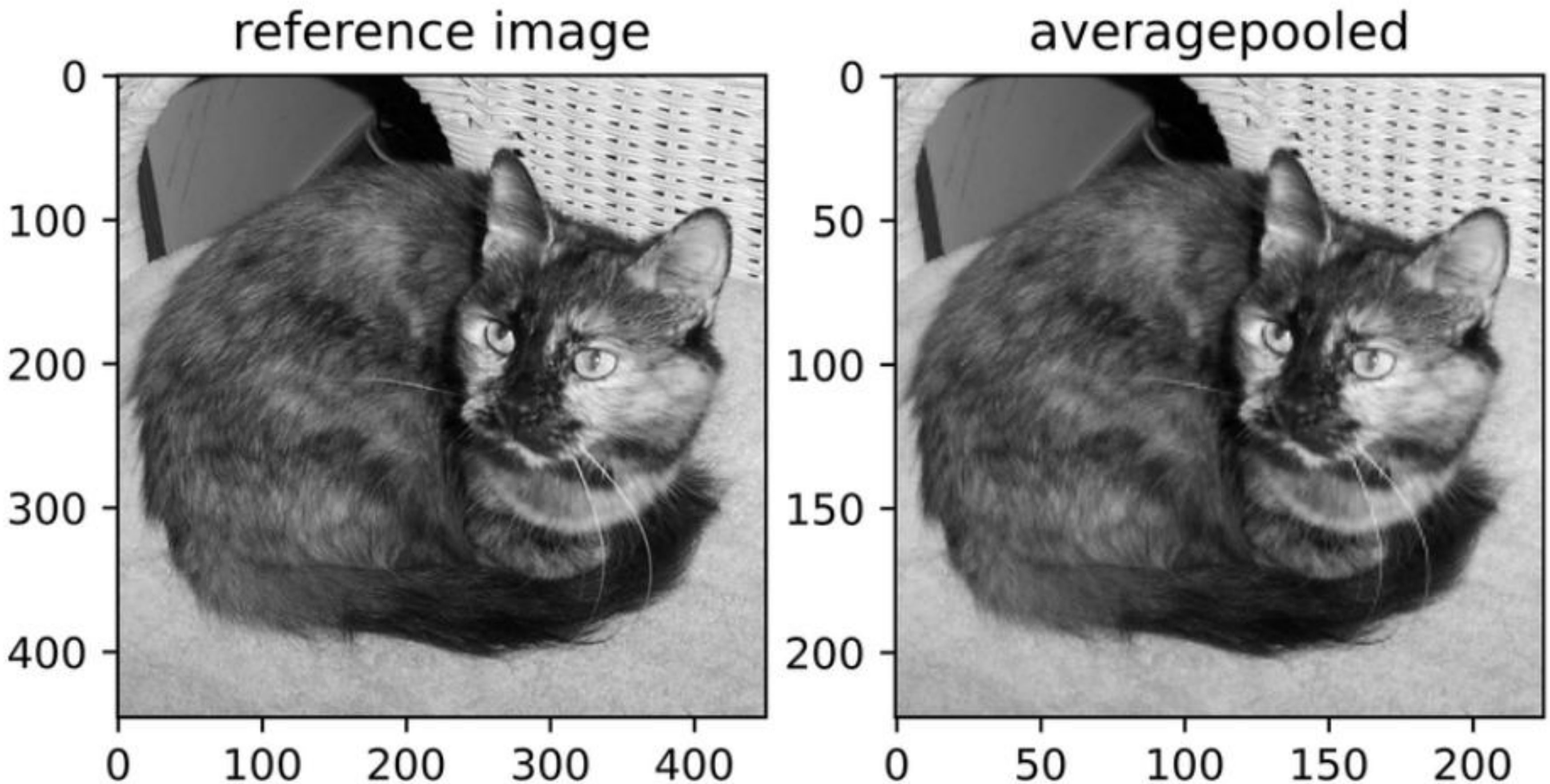
Max pooling over detected edges.

Average Pooling

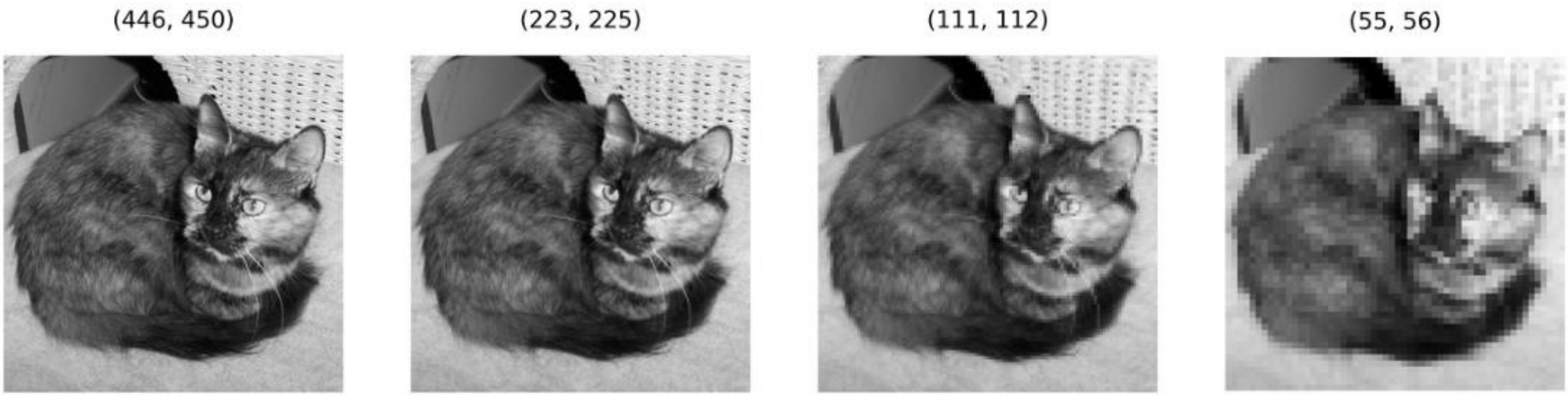


TensorFlow > API > TensorFlow v2.16.1 > Python

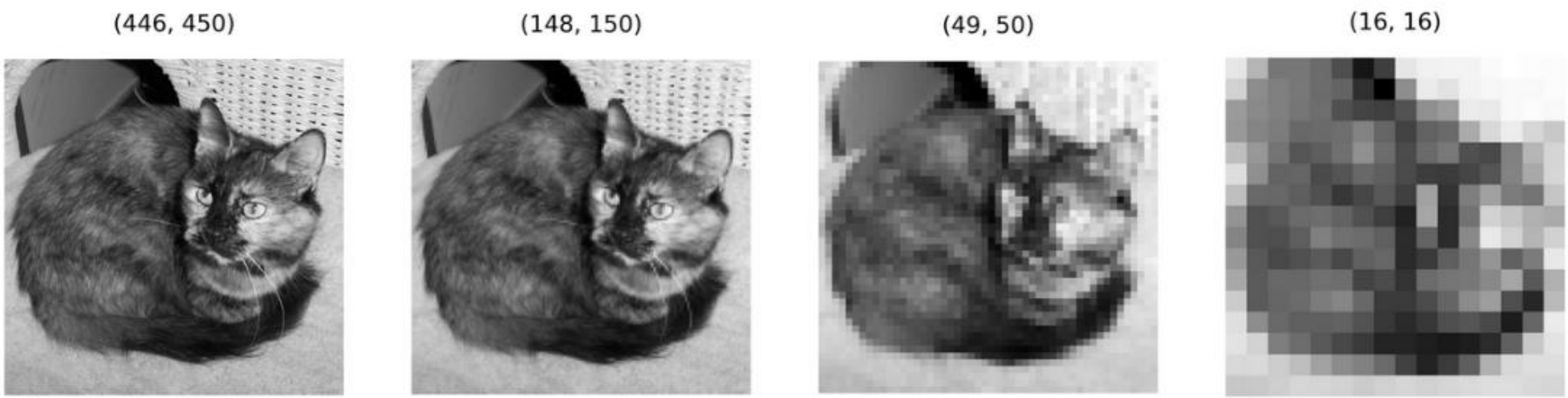
tf.keras.layers.AveragePooling2D



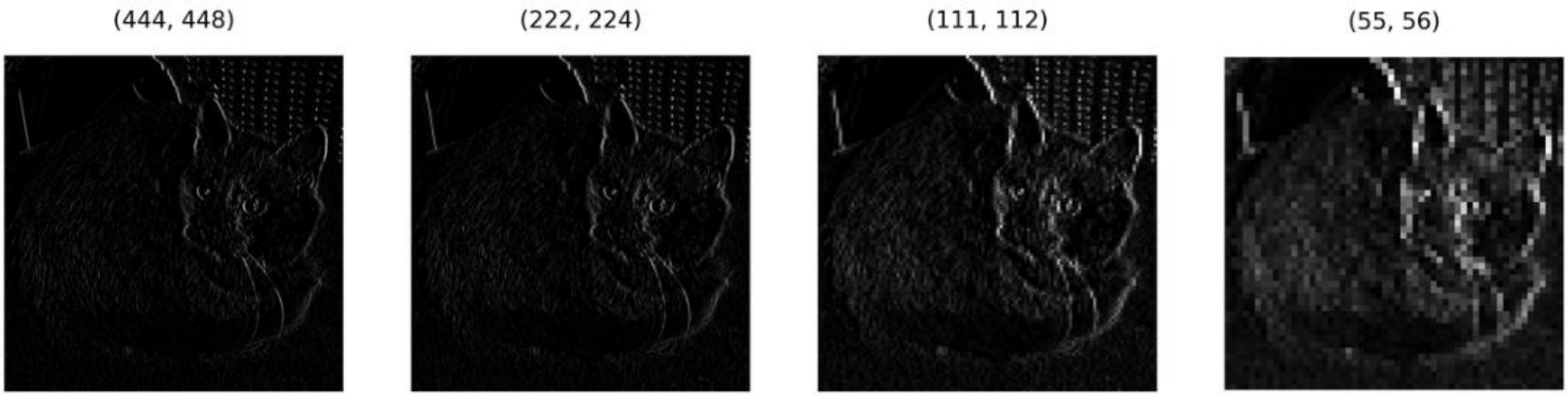
Producing an average pooled representation.



Reference image through 3 iterations of average pooling using a (2, 2) kernel.



Reference image through 3 iterations of average pooling using a (3, 3) kernel.



Average pooling over detected edges.

tf.keras.layers.GlobalAveragePooling2D

Feature map size calculations

- After a pooling layer, the size of the feature map can be calculated using the following formula, which takes into account the input size, the pooling window size, stride, and padding.
- The general formula for calculating the output size of a pooling operation is:

$$\text{Output size} = \left\lfloor \frac{\text{Input size} - \text{Pool size}}{\text{Stride}} \right\rfloor + 1$$

- Here's how you calculate it step-by-step:
- **Input Size (H_in or W_in)**: This is the height or width of the input feature map.
- **Pool Size (K)**: This is the size of the pooling window (e.g., 2 for a 2x2 pooling window).
- **Stride (S)**: This is the step size with which the pooling window moves across the input.
- **Padding (P)**: This is the number of padding pixels added to the input feature map.

Without Padding

If no padding is applied (i.e., padding = 0), the output size is calculated as follows:

$$\text{Output size} = \left\lfloor \frac{\text{Input size} - \text{Pool size}}{\text{Stride}} \right\rfloor + 1$$

With Padding

If padding is applied, the output size is adjusted accordingly. Padding adds additional pixels around the input feature map.

$$\text{Output size} = \left\lfloor \frac{\text{Input size} + 2 \times \text{Padding} - \text{Pool size}}{\text{Stride}} \right\rfloor + 1$$