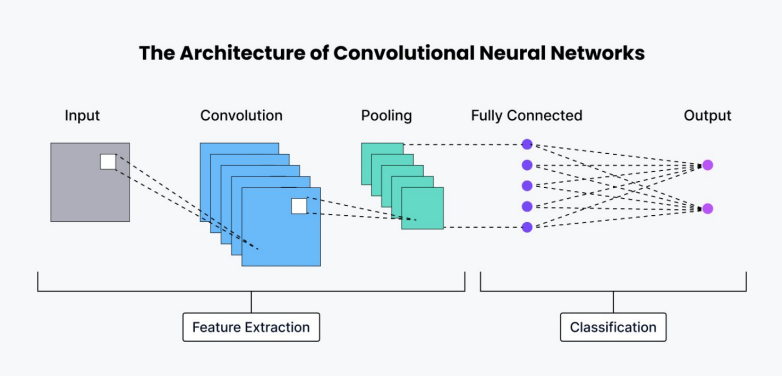
Understanding Adaptive Pooling

—

Divya Bhardwaj

First: Understanding CNNs



Main Components:

- Convolutional layer
  - Input (3 dimensions, height, width and depth, which corresponds to RGB in an image)
  - Kernel/Filter (detects features)
  - Feature Map (final output from the series of dot products from the input and the filter)
- Pooling layer
  - Also known as downsampling, reduces spatial dimensions of the feature map while preserving the most important information
  - Uses a sliding window (or receptive field) to look at a small region of the input feature map and applies an aggregation function to reduce that region to a single value.
  - Advantages: reducing complexity, improving efficiency, limiting risk of overfitting
- Fully-connected (FC) layer
  - Performs the task of classification based on the features extracted through the previous layers and their different filters

Source: IBM: What are Convolutional Neural Networks?

Photo Credit: What is a Convolutional Neural Network? An Engineer's Guide

Source: IBM: What are Convolutional Neural Networks?, Gen AI for visualizing example

Types of Pooling

1. Max Pooling

- Generally more common than average pooling
- Selects maximum value from the receptive field
- Focuses on retaining the strongest/most prominent features

Example

Feature map:

$$\begin{bmatrix} 1 & 3 & 2 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

After Max Pooling

with a 2x2 filter, stride 2:

$$\begin{bmatrix} 6 & 8 \\ 14 & 16 \end{bmatrix}$$

2. Average Pooling

- Calculates average value from the values within the receptive field
- More generalized features, can be used where detail preservation is not as crucial

Example

Feature map:

$$\begin{bmatrix} 1 & 3 & 2 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

After Average Pooling

with 2x2 filter, stride 2:

$$\begin{bmatrix} 3.75 & 5.25 \\ 11.5 & 13.5 \end{bmatrix}$$

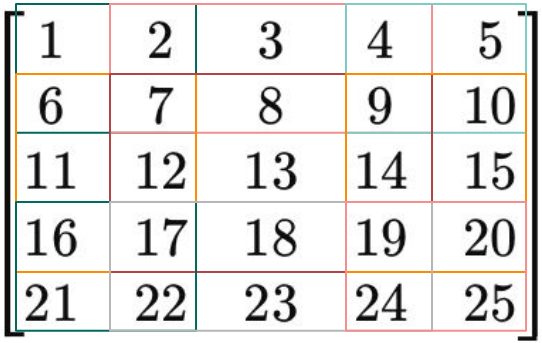
Adaptive (Average) Pooling

- Traditional pooling: the pooling window (e.g., a 2x2 or 3x3 window) is fixed, and the stride is typically the same as the window size, meaning that the size of the output feature map is directly determined by the input size and the pool size.
- Adaptive pooling: takes an input feature map and allows you to specify the desired output size (in terms of height and width), regardless of the input size.
  - This means it calculates the size of the pooling window and stride dynamically to ensure the desired output size will be achieved.
- Then it performs average pooling as previously described
  - Can also do adaptive max pooling
- “Particularly useful in tasks where the input size can vary significantly, such as in image classification, object detection, and semantic segmentation. By ensuring that the output size is consistent, it allows for easier comparison and aggregation of features across different inputs.”

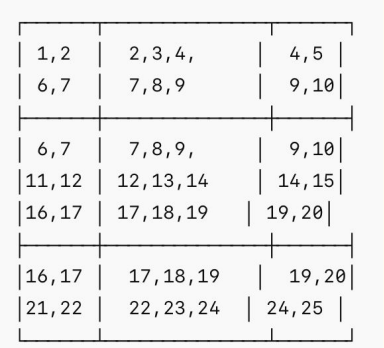
Source: Adaptive Average Pooling Layer

Walking Through an Adaptive Pooling Example

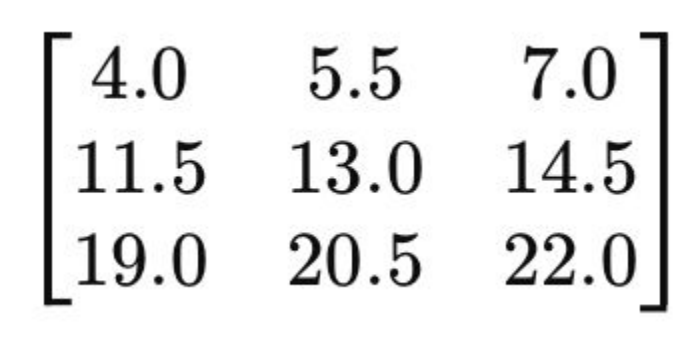
We have a 5x5 matrix and let's say the goal is to output a 3x3



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25



1,2 6,7	2,3,4, 7,8,9	4,5 9,10
6,7 11,12 16,17	7,8,9, 12,13,14 17,18,19	9,10 14,15 19,20
16,17 21,22	17,18,19 22,23,24	19,20 24,25



$$\begin{bmatrix} 4.0 & 5.5 & 7.0 \\ 11.5 & 13.0 & 14.5 \\ 19.0 & 20.5 & 22.0 \end{bmatrix}$$

Source: Adaptive Average Pooling Layer; Gen AI for visualizing the example.

In Our Project

- Current: AdaptiveAvgPool2d((2,2))
- Possible things to try:
  - AdaptiveMaxPool2d
    - Could see if max pooling is better for our images, since this would help preserve most dominant features in a region, can help reduce noise
  - Experiment with increased output size (e.g. (4,4), (8,8)
    - Could see if the increased capacity/more detailed features significantly improve the model and would be worth the increased computation
- Integration of adaptive pooling with attention
  - Could make the pooling window size dynamic based on the attention scores. (e.g., patches with higher attention scores could have a smaller pooling window to preserve more fine-grained features, while patches with lower attention could use larger windows to down-sample them more aggressively)