### **Paddings and Strides in CNN**

In **Convolutional Neural Networks (CNNs)**, **paddings** and **strides** are crucial parameters that influence how filters interact with input data. These parameters affect the size of the output feature maps, computational efficiency, and how well features are preserved.

### **1. Padding in CNNs**

**Padding** refers to the process of adding extra pixels (typically zeros) around the edges of the input data before applying the convolution operation. Padding is used to:

1. **Control Output Size**:
   * Without padding, the size of the output decreases after each convolution, which might lead to the loss of important information in deeper layers.
2. **Preserve Spatial Information**:
   * Padding allows the filter to interact with edge and corner pixels more effectively.
3. **Avoid Shrinking Feature Maps**:
   * Especially in deep networks, padding ensures feature maps don't shrink too much as layers progress.

#### **Types of Padding**

1. **Valid Padding**:
   * No padding is applied to the input.
   * Output size is smaller than the input size.
   * Formula for output size: Output Size=Input Size−Filter SizeStride+1\text{Output Size} = \frac{\text{Input Size} - \text{Filter Size}}{\text{Stride}} + 1Output Size=StrideInput Size−Filter Size​+1
2. **Example**:
   * Input: 5×55 \times 55×5
   * Filter: 3×33 \times 33×3
   * Stride: 111
   * Output: (5−3)/1+1=3×3(5 - 3)/1 + 1 = 3 \times 3(5−3)/1+1=3×3
3. **Same Padding**:
   * Adds padding to ensure the output size matches the input size.
   * The amount of padding depends on the filter size and stride.
   * Formula for padding: P=(Filter Size−1)2P = \frac{(\text{Filter Size} - 1)}{2}P=2(Filter Size−1)​
4. **Example**:
   * Input: 5×55 \times 55×5
   * Filter: 3×33 \times 33×3
   * Stride: 111
   * Padding: P=(3−1)/2=1P = (3 - 1)/2 = 1P=(3−1)/2=1
   * Output: 5×55 \times 55×5 (same size as input).

### **2. Strides in CNNs**

**Stride** determines how much the filter moves across the input during the convolution operation. It controls the overlap between filter positions.

1. **Stride = 1**:
   * The filter slides one pixel at a time.
   * Output size is maximized, preserving most of the spatial information.
2. **Stride > 1**:
   * The filter skips pixels as it slides.
   * Reduces the size of the output feature map (downsampling).
   * Useful for reducing computational complexity.

#### **Formula for Output Size (with Padding and Stride)**

Output Height=Input Height−Filter Height+2⋅PaddingStride+1\text{Output Height} = \frac{\text{Input Height} - \text{Filter Height} + 2 \cdot \text{Padding}}{\text{Stride}} + 1Output Height=StrideInput Height−Filter Height+2⋅Padding​+1 Output Width=Input Width−Filter Width+2⋅PaddingStride+1\text{Output Width} = \frac{\text{Input Width} - \text{Filter Width} + 2 \cdot \text{Padding}}{\text{Stride}} + 1Output Width=StrideInput Width−Filter Width+2⋅Padding​+1

### **Practical Examples**

#### **Example 1: Valid Padding, Stride = 1**

Input:

5×55 \times 55×5

Filter:

3×33 \times 33×3

Stride:

111 Output Size=5−31+1=3×3\text{Output Size} = \frac{5 - 3}{1} + 1 = 3 \times 3Output Size=15−3​+1=3×3

#### **Example 2: Same Padding, Stride = 1**

Input:

5×55 \times 55×5

Filter:

3×33 \times 33×3

Stride:

111

Padding:

P=3−12=1P = \frac{3 - 1}{2} = 1P=23−1​=1 Output Size=5−3+2⋅11+1=5×5\text{Output Size} = \frac{5 - 3 + 2 \cdot 1}{1} + 1 = 5 \times 5Output Size=15−3+2⋅1​+1=5×5

#### **Example 3: Same Padding, Stride = 2**

Input:

6×66 \times 66×6

Filter:

3×33 \times 33×3

Stride:

222

Padding:

P=3−12=1P = \frac{3 - 1}{2} = 1P=23−1​=1 Output Size=6−3+2⋅12+1=3×3\text{Output Size} = \frac{6 - 3 + 2 \cdot 1}{2} + 1 = 3 \times 3Output Size=26−3+2⋅1​+1=3×3

### **Impact of Padding and Strides**

| **Parameter** | **Effect on Output** |
| --- | --- |
| **Padding** | Increases the size of the output feature map. Preserves edges. |
| **Stride** | Larger strides reduce the size of the output feature map. |
| **Combination** | The combination of padding and strides determines downsampling or preserving input size. |

### **Code Example: Padding and Strides**

| import tensorflow as tf from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Conv2D  # Example Input input\_data = tf.random.normal([1, 6, 6, 1]) # Batch size 1, 6x6 grayscale image  # Model with Valid Padding model\_valid = Sequential([  Conv2D(filters=1, kernel\_size=(3, 3), strides=(1, 1), padding='valid') ]) output\_valid = model\_valid(input\_data)  # Model with Same Padding model\_same = Sequential([  Conv2D(filters=1, kernel\_size=(3, 3), strides=(1, 1), padding='same') ]) output\_same = model\_same(input\_data)  print(f"Input Shape: {input\_data.shape}") print(f"Output Shape (Valid Padding): {output\_valid.shape}") print(f"Output Shape (Same Padding): {output\_same.shape}") |
| --- |

### **Visualization of Padding and Strides**

#### **Effect of Strides**

With stride 1:

* The filter slides across every pixel.

With stride 2:

* The filter skips every other pixel, reducing spatial resolution.

#### **Effect of Padding**

* **No Padding (Valid)**:
  + Filters are constrained by the input boundaries, reducing output size.
* **Same Padding**:
  + Pads the input to ensure the output size matches the input size.

### **Key Takeaways**

1. **Padding**:
   * Controls whether edges are preserved or reduced in size.
   * Common types: **Valid** (no padding) and **Same** (output size matches input size).
2. **Stride**:
   * Determines how much the filter moves across the input.
   * Larger strides result in smaller outputs (downsampling).
3. **Choosing Padding and Strides**:
   * Use **Same Padding** for deep networks where spatial dimensions need to be preserved.
   * Use **Strides > 1** for reducing feature map size or computational cost.