**HW to Chapter 13 “Convolutional Layer”**

**Non-programming Assignment**

1. What is convolution operation and how does it work?
2. Why do we need convolutional layers in neural networks?
3. How are sizes of the original image, the filter, and the resultant convoluted image are related?
4. What is padding and why is it needed?
5. What is strided convolution and why is it needed?
6. **What is convolution operation and how does it work?**

Answer

Convolution is a fundamental operation in image processing and deep learning. It involves sliding a small matrix, called a filter (or kernel), over an input image and computing a weighted sum (dot product) between the filter and the corresponding region of the image.

This operation helps extract meaningful features such as edges, textures, and patterns, producing a feature map that highlights important aspects of the image.

**How Convolution Works:**

* Flip the filter horizontally and vertically (though some implementations omit this step).
* Slide the filter across the image, moving from left to right and top to bottom.
* Perform Hadamard product between the filter and the current region of the image.
* Sum up the results to get a single value in the feature map.
* Repeat this process until the entire image has been covered.

Example Calculation:

If we apply a 2×2 filter to a 4×3 image, the resulting feature map will have a size of:

(4-2+1) × (3-2+1) = 3×2.

Each value in this feature map is computed by applying the filter to a specific region of the image.

For instance, the top-left value in the feature map is computed by multiplying the filter with the top left 2×2 region of the image and summing up the results.

1. **Why do we need convolutional layers in neural networks?**

Answer

Convolutional layers are essential in deep learning models, particularly in Convolutional Neural Networks (CNNs), because they address several key challenges in image processing:

* **Efficient parameter sharing**: Instead of connecting every neuron to every pixel (like in fully connected layers), convolutional layers use filters that are shared across the entire image. This drastically reduces the number of parameters and computational cost.
* **Feature extraction**: These layers automatically detect useful patterns, such as edges, textures, and shapes, which help the model recognize objects more effectively.
* **Hierarchical learning**: Stacking multiple convolutional layers allows the model to learn from simple features (like edges) to complex ones (like object shapes).
* **Preserving spatial structure**: Unlike fully connected layers, which flatten the image, convolutional layers retain important spatial relationships between pixels, making them ideal for image recognition and processing.

1. **How are the sizes of the original image, the filter, and the resultant convoluted image related?**

Answer

The output size of a convoluted image depends on the input size (n₁ × n₂), the filter size (f₁ × f₂), the padding (p), and the stride (s).

Formula for output size

Output size=([n1+2p−f1]/s +1) × ([n2+2p−f2]/s +1)

where:

n1, n2 = input dimensions (height × width)

f1, f2 = filter dimensions

p = padding size

s = stride (step size of the filter)

For no padding (p=0) and stride=1, the output size is:

(n1 − f1 + 1) × (n2 − f2 +1)

This means convolution shrinks the image unless padding is used.

This relationship exists because the filter must fit entirely within the input image during each convolution operation. As the filter moves across the image, it can only slide to positions where it remains fully contained within the image boundaries.

The “+1” in the formula accounts for the initial position, where the filter starts at the top-left corner of the image. Without padding, the final valid position occurs when the right and bottom edges of the filter align with the edges of the input image.

Because of this constraint, convolution without padding naturally reduces the image size—the filter doesn’t get applied to pixels at the very edges, leading to a smaller output. This is why padding is often used when we want to preserve spatial dimensions and ensure all regions of the input are processed equally.

1. What is padding and why is it needed?

Answer

Padding is the process of adding extra pixels (usually zeros) around the edges of an image before performing convolution.

Why use padding?

* Preserves image size: Without padding, repeated convolutions can significantly reduce the size of the image, making it hard to extract deep features.
* Retains border information: Pixels at the edges of the image are often underrepresented in feature detection. Padding ensures they contribute more equally.
* Controls the output size: By adjusting the padding, we can fine-tune the dimensions of the feature map.

Formula for maintaining original size:

To keep the output size the same as the input:

p = [f−1] / 2

(where f is the filter size, assuming a square filter).

1. **What is strided convolution and why is it needed?**

Answer

A strided convolution is a variation of convolution where the filter moves by more than one pixel at each step. Instead of shifting by 1 pixel at a time, the filter moves by s pixels, effectively down sampling the image.

Why use strided convolution?

* **Reduces computation**: By skipping pixels, the model requires fewer operations and less memory.
* **Down samples the image**: It reduces the spatial dimensions of the feature map, similar to max pooling.
* **Captures larger patterns**: A larger stride increases the receptive field, helping the network detect high-level patterns more efficiently.

Formula for output size with stride:

Output size=([n1+2p−f1] / s +1) × ([n2+2p−f2] / s +1)

where s is the stride.

Example:

If we have a 5×5 image, a 3×3 filter, and a stride of 2, the output size is:

(5−3)/2+1=2×2

Compared to a 3×3 output when stride = 1, the larger stride results in a smaller feature map.