Deep Learning Notes 9.1

Key Words : The Convolution Operation

1. The Convolution Operation

1. **Convolutional networks**, also known as **Convolutional neural networks** or **CNNs**, are a specialized kind of neural network for processing data that has a known, grid-like topology(e.g., time-series data which can be thought if as a 1D grid taking samples at regular time intervals, image data which can be thought of as a 2D grid of pixels).

2. Convolution is a specialized kind of linear operation. **Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.**

3. **Convolution** operation can be defined as follows

$$s(t) = \int x(a)w(t-a)da \Rightarrow s(t) = (x * w)(t) \tag{1}$$

please note that if $t < a$, then $w(t-a)$ must be 0.

4. The function x is often referred to as the **input**, the function w refers to **kernel**, and the output $s(t)$ is sometimes referred to as the **feature map**

5. If we assume that x and w are defined only on integer t, we can define the discrete convolution

$$s(t) = (x * w)(t) = \sum_{\alpha=-\infty}^{\infty} x(a)w(t-a) \tag{2}$$

6. In machine learning applications, **the input is usually a multidimensional array of data** and **the kernel is usually a multidimensional array of parameters that are adapted by the learning algorithm**. We refer to these multidimensional arrays as **tensors**.

7. If I is a two-dimensional image and K is a two-dimensional kernel, then the two-dimensional convolution is defined as follows

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{3}$$

since the convolution is commutative, so we have

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n) \tag{4}$$

Let's take an example. If $I = [1, 2; 3, 4]$, $K = [0.1, 0.2; 0.3, 0.4]$; What is the $I * K$?

1). Flip the kernel

$$\begin{bmatrix} 0.1 & 0.2 \\ 0.3 & 0.4 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.4 & 0.3 \\ 0.2 & 0.1 \end{bmatrix}$$

2). Then calculate every element of the convolution matrix; note that the input and the kernel matrix are all $[2, 2]$ size, so the size the final convolution matrix S would be $[2 + 2 - 1, 2 + 2 - 1]$, i.e., $[3, 3]$.

$$\begin{gathered} S_{11} = 0.1 \times 1 = 0.1 \\ S_{12} = 0.2 \times 1 + 0.1 \times 2 = 0.4 \\ \cdots \\ S_{22} = 0.4 \times 1 + 0.3 \times 2 + 0.2 \times 3 + 0.1 \times 4 = 2 \\ \cdots \\ S_{33} = 0.4 \times 4 = 1.6 \end{gathered}$$

3). The final S is

$$S = \begin{bmatrix} 0.1 & 0.4 & 0.4 \\ 0.6 & 2 & 1.6 \\ 0.9 & 2.4 & 1.6 \end{bmatrix}$$

```
>>  I = [1, 2; 3, 4]; K = [0.1, 0.2; 0.3, 0.4]; conv2(I, K, 'full')

ans =

    0.1000    0.4000    0.4000
    0.6000    2.0000    1.6000
    0.9000    2.4000    1.6000
```

8. It is noteworthy that we restrict the output to only positions where the kernel lies entirely within the image, called “valid” convolution in some contexts.

```
>>  I = [1, 2; 3, 4]; K = [0.1, 0.2; 0.3, 0.4]; conv2(I, K, 'valid')

ans =

     2
```

9. Discrete convolution can be viewed as multiplication by a matrix. However, **the matrix has several entries constrained to be equal to other entries**.

1). for **univariate discrete convolution**, **each row of the matrix is constrained to be equal to the row above shifted by one element**. This is known as a **Toeplitz** matrix. Let’s take an example as follows

$$\mathbf{h} = [h_0, h_1, h_2, h_3] \Rightarrow \mathbf{H} = \begin{bmatrix} h_0 & h_3 & h_2 & h_1 \\ h_1 & h_0 & h_3 & h_2 \\ h_2 & h_1 & h_0 & h_3 \\ h_3 & h_2 & h_1 & h_0 \end{bmatrix}$$

2). **In two dimensions**, a **doubly block circulant matrix** corresponds to convolution. Let’s take an example as follows

$$\mathbf{X} = \begin{bmatrix} \mathbf{H}_0 & \mathbf{H}_3 & \mathbf{H}_2 & \mathbf{H}_1 \\ \mathbf{H}_1 & \mathbf{H}_0 & \mathbf{H}_3 & \mathbf{H}_2 \\ \mathbf{H}_2 & \mathbf{H}_1 & \mathbf{H}_0 & \mathbf{H}_3 \\ \mathbf{H}_3 & \mathbf{H}_2 & \mathbf{H}_1 & \mathbf{H}_0 \end{bmatrix}$$

where

$$\mathbf{h} = [h_0, h_1, h_2, h_3] \Rightarrow \mathbf{H_0} = \begin{bmatrix} h_0 & h_3 & h_2 & h_1 \\ h_1 & h_0 & h_3 & h_2 \\ h_2 & h_1 & h_0 & h_3 \\ h_3 & h_2 & h_1 & h_0 \end{bmatrix}$$

$$\mathbf{h} = [h_3, h_0, h_1, h_2] \Rightarrow \mathbf{H_1} = \begin{bmatrix} h_3 & h_2 & h_1 & h_0 \\ h_0 & h_3 & h_2 & h_1 \\ h_1 & h_0 & h_3 & h_2 \\ h_2 & h_1 & h_0 & h_3 \end{bmatrix}$$

$$\vdots$$