Akash Tadwai
Indian Institute of Technology Hyderabad
Deep Learning for Vision
ES18BTECH11019

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Assignment-I

Akash Tadwai - ES18BTECH11019

February 4, 2021

1. Edge Detection

Linear Filters (2+2+2+1+3=10 marks) : In class, we introduced 2D discrete space convolution. Consider an input image $I[i, j]$ and a filter $F[i, j]$. The 2D convolution $F * I$ is defined as

$$(F * I)[i, j] = \sum_{k,l} I[i - k, j - l]F[k, l]$$

(a) Convolve the following I and F (using pen and paper). Assume we use zero-padding where necessary.

$$I = \begin{bmatrix} 2 & 0 & 1 \\ 1 & -1 & 2 \end{bmatrix} F = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}$$

Please DO NOT write programs. It will also be helpful for answering question (d).

A: The output dimension of matrix is $N_h - F_h + 1 \times N_w - F_w + 1$ if input is not padded. But, as we are using "Zero" padding and the kernel is an even sized kernel, I have padded the input with top row and left row. Now using the convolution formula, we get,

$$\mathbb{F}' = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \mathbb{I}' = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & -1 & 2 \end{bmatrix}$$

Let $y[i, j]$ is the output of the convolution we get from the convolution formula that,

$$\begin{aligned} y[0, 0] &= \sum_{k,l} I[0 - k, 0 - l]F[k, l] \\ &= F[0, 0]I[0, 0] + F[0, 1]I[0, -1] + F[1, 1]I[-1, -1] + F[1, 0]I[-1, 0] \\ &= (1)(2) + (1)(0) + (-1)(0) + (-1)(0) \\ &= 2 \end{aligned}$$

Akash Tadwai
Indian Institute of Technology Hyderabad
Deep Learning for Vision
ES18BTECH11019

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

$$
\begin{aligned}
y[0,1] &= \sum_{k,l} I[0-k, 1-l]F[k,l] \\
&= F[0,0]I[0,1] + F[0,1]I[0,0] + F[1,0]I[-1,1] + F[1,1]][-1,0] \\
&= (1)(1) + (1)(2) + (-1)(0) + (-1)(0) \\
&= 3
\end{aligned}
$$

$$
\begin{aligned}
y[1,0] &= \sum_{k,l} I[1-k, 0-1]F[k,l] \\
&= F[0,0]\mathrm{I}[1,0] + F[0,1]I[1,-1] + F[1,1]I[0,-1] + F[1,0]I[0,0] \\
&= (1)(0) + (1)(0) + (-1)(0) + (-1)(2) \\
&= -2
\end{aligned}
$$

$$
\begin{aligned}
y[1,1] &= \sum_{k,l} I[1-k, 1-1]F[k,l] \\
&= F[0,0]I[1,1] + F[0,1]I[1,0] + F[1,0]I[0,1] + F[1,1]I[0,0] \\
&= (1)(-1) + (1)(0) + (-1)(1) + (-1)(2) \\
&= -1 - 1 - 2 = -4
\end{aligned}
$$

$$
\begin{aligned}
y[2,0] &= \sum_{k,l} I[2-k, 0-l]F[k,l] \\
&= F[0,0]I[2,0] + F[1,0]I[1,0] + F[1,1]I[1,-1] + F[0,1]I[2,-1] \\
&= (1)(1) + (-1)(0) + (-1)(0) + (1)(0) \\
&= 1
\end{aligned}
$$

$$
\begin{aligned}
y[2,1] &= \sum_{k,l} I[2-k, 1-l]F[k,l] \\
&= F[0,0]I[2,1] + F[1,0]I[1,0] + F[1,1]I[1,0] + F[0,1]I[2,0] \\
&= (1)(2) + (-1)(0) + (-1)(0) + (1)(2) \\
&= 4
\end{aligned}
$$

$$
\Rightarrow \quad y = \begin{bmatrix} 2 & -2 & 1 \\ 3 & -4 & 4 \end{bmatrix}
$$

(b) Note that the F given in Equation 2 is separable, that is, it can be written as a product of two filters: $F = F_1F_2$. Find F_1 and F_2. Then, compute $(F_1 * I)$ and $F_2 * (F_1 * I)$, i.e., first perform 1D convolution on each column, followed by another 1D convolution on each row. (Please DO NOT write programs. Do it by hand.)

A: We know that if a kernel is separable the *rank* of kernel will be 1 . Here as the second column of the filter $\mathbb{F}$ is a scalar multiple of the first column, the $rank(\mathbb{F}) = 1$.

Akash Tadwai
Indian Institute of Technology Hyderabad
Deep Learning for Vision
ES18BTECH11019

$$\begin{aligned}\mathbb{F} &= F_1 \cdot F_2 \\ &= \begin{bmatrix}1\\1\end{bmatrix} \times \begin{bmatrix}1 & -1\end{bmatrix}\end{aligned}$$

$$\mathbb{I}' = \begin{bmatrix}0 & 0 & 0\\ 2 & 0 & 1\\ 1 & -1 & 2\end{bmatrix}, F_1 = \begin{bmatrix}1\\1\end{bmatrix}, F_2 = \begin{bmatrix}1 & -1\end{bmatrix}$$

$$(F_1 * I') = \begin{bmatrix}2 & 0 & 1\\ 3 & -1 & 3\end{bmatrix}$$

Now again padding to left so that output dimensions won't change and flipping the kernel and doing correlation,

$$(F_1 * I')' = \begin{bmatrix}0 & 2 & 0 & 1\\ 0 & 3 & -1 & 3\end{bmatrix}$$

$$(F_2 * (F_1 * I')') = \begin{bmatrix}2 & -2 & 1\\ 3 & -4 & 4\end{bmatrix}$$

(c) Prove that for any separable filter $F = F_1F_2$
$F * I = F_2 * (F_1 * I)$ Hint: Expand Equation 1 directly.
A:

$$\begin{aligned}(I * F)[m,n] &= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} I[i,j]\cdot F[m-i,n-j] \\ &= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} I[i,j]\cdot (F_1[m-i]\cdot F_2[n-j]) \quad (F[i,j] = F_1[j] * F2[j]) \\ &= \sum_{j=-\infty}^{\infty} F_2[n-j] \sum_{i=-\infty}^{\infty} I[i,j]\cdot F_1[m-i] \\ &= \sum_{j=-\infty}^{\infty} F_2[n-j]\,(I * F_1) \\ &= F_2 * (I * F_1) \\ &= F_2 * (F_1 * I)\,(\text{Associativity of Convolution})\end{aligned}$$

(d) Carefully count the exact number of multiplications (multiplications only, including those multiplications due to zero-padding) involved in part (a) and part (b). which one of these

requires fewer operations? You may find the computation steps you wrote down for (a) and (b) helpful here.
A: Multiplications in part (a) : 24 $(M1 * N1 * M2 * N2)$
Multiplications in part (b) : 24 $(M1 * N1 * (M2 + N2))$

(e) Consider a more general case: I is an $M_1 \times N_1$ image, and F is an $M_2 \times N_2$ separable filter. i. How many multiplications do you need to do a direct 2D convolution? ii. How many multiplications do you need to do 1D convolution on rows and columns? Hint: For (i) and (ii), we are asking for two functions of M_1, N_1, M_2 and N_2 here, no approximations. iii. Use Big-O notation to argue which one is more efficient in general: direct 2D convolution or two successive 1D convolutions?

A: (i) $(M_1 \times N_1 \times M_2 \times N_2)$ multiplications
(ii) $(M_1 \times N_1 (M_2 + N_2))$ multiplications
(iii) direct 2D convolution is $O(M_1 N_1 M_2 N_2)$ while two 1D convolutions is $O(M_1 N_1 (M_2 + N_2))$
For large $M_2, N_2, M_2 N_2 \gg M_2 + N_2$ So two successive 1D convolutions is more efficient in general.

2. Canny Edge Detector (**2.5+2.5=5 marks**) : Suppose the Canny edge detector successfully detects an edge. The detected edge (shown as the red horizontal line in Figure 2a) is then rotated by θ, where the relationship between a point on the original edge (x, y) and a point on the rotated edge (x', y') is defined as

$$x' = x\cos\theta; y' = x\sin\theta$$

(a) Will the rotated edge be detected using the same Canny edge detector? Provide either a mathematical proof or a counter example. Hint: The detection of an edge by the Canny edge detector depends only on the magnitude of its derivative. The derivative at point (x, y) is determined by its components along the x and y directions. Think about how these magnitudes have changed because of the rotation.
A: Let the magnitude of initial derivative is

$$\mathcal{L} = \sqrt{dx^2 + dy^2} = |dx| \quad (dy = 0 \text{ as point is on X-axis})$$

The magnitude of rotated derivative is

$$\mathcal{L}' = \sqrt{(dx\cos\theta)^2 + (dx\sin\theta)^2} = |dx|$$

So the magnitude doesn't change, it can be detected using the same Canny edge detector.

(b) After running the Canny edge detector on an image, you notice that long edges are broken into short segments separated by gaps. In addition, some spurious edges appear. For each of the two thresholds (low and high) used in hysteresis thresholding, state how you would adjust the threshold (up and down) to address both problems. Assume that a setting exists for the two thresholds that produce the desired result.

A: Parts of the long edges are detected, so the high threshold is low enough for these edges, but the edges are disconnected because the low threshold is too high. Lowering the low threshold will include more pixels of the long edges. Eliminating the spurious edges requires a higher high threshold. The high threshold should be increased only slightly, so as not to make the long edges disappear.

******THE END******