Assignment-1

AI5100: Deep Learning

Shubham Jain

SM20MTECH12007

Date of submission: 04/02/2021

IIT Hyderabad

Solution-1

1.a) **Convolve Image (I) and filter (F).**

step 1 Rotating Filter by 180°.

$$\begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \qquad [1.]$$

step 2 Passing rotated filter over padded image Matrix.

$$I \times F = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & -1 & 2 \end{bmatrix} \times \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \qquad [2.]$$

step 3 Calculation

$$= \begin{bmatrix} (0\times -1)+(0\times 1)+(0\times -1)+(2\times 1) & (0\times -1)+(0\times 1)+(2\times -1)+(0\times 1) & (0\times -1)+(0\times 1)+(0\times -1)+(1\times 1) \\ (0\times -1)+(2\times 1)+(0\times -1)+(1\times 1) & (2\times -1)+(0\times 1)+(1\times -1)+(-1\times 1) & (0\times -1)+(1\times 1)+(-1\times -1)+(2\times 1) \end{bmatrix}$$

step 4 Results

$$I \times F = \begin{bmatrix} 2 & -2 & 1 \\ 3 & -4 & 4 \end{bmatrix} \qquad [3.]$$

1.b) **For separable filters F can be written as Matrix product of F_1 and F_2: -**

step 1 Using SVD or analytically we can separate F in separate 1D Filters.

$$\underbrace{\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}}_{F} = \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{F_1} \times \underbrace{\begin{bmatrix} -1 & 1 \end{bmatrix}}_{F_2} \qquad [4.]$$

step 2 Passing Filter F_1 over padded image matrix

$$I \times F_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & -1 & 2 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad [5.]$$

step 3 Calculation

$$I \times F_1 = \begin{bmatrix} (1\times 0)+(1\times 0) & (1\times 0)+(1\times 2) & (1\times 0)+(1\times 0) & (1\times 0)+(1\times 1) \\ (1\times 0)+(1\times 0) & (1\times 2)+(1\times 1) & (1\times 0)+(1\times -1) & (1\times 1)+(1\times 2) \end{bmatrix} \qquad [6.]$$

step 4 Intermediate result after passing F_1.

$$I \times F_1 = \begin{bmatrix} 0 & 2 & 0 & 1 \\ 0 & 3 & -1 & 3 \end{bmatrix} \quad [7.]$$

step 5 Passing Filter F_2 over Intermediate result

$$I \times F_1 \times F_2 = \begin{bmatrix} 0 & 2 & 0 & 1 \\ 0 & 3 & -1 & 3 \end{bmatrix} \times \begin{bmatrix} -1 & 1 \end{bmatrix} \quad [8.]$$

step 6 Calculation

$$I \times F_1 \times F_2 = \begin{bmatrix} (-1 \times 0) + (1 \times 2) & (-1 \times 2) + (1 \times 0) & (-1 \times 0) + (1 \times 1) \\ (-1 \times 0) + (1 \times 3) & (-1 \times 3) + (1 \times -1) & (-1 \times -1) + (1 \times 3) \end{bmatrix} \quad [9.]$$

step 7 Result.

$$I \times F_1 \times F_2 = \begin{bmatrix} 2 & -2 & 1 \\ 3 & -4 & 4 \end{bmatrix} \quad [10.]$$

1.c) <u>Proof of $\boldsymbol{F * I = F2 * (F1 * I)}$</u>

Given to us: -

$$(F * I)[i,j] = \sum_{k,l} I[i - k, j - l]F[k,l] \quad [11.]$$

As the filter is separable, it can be written as: -

$$F[k,l] = F_1[k] \times F_2[l] \quad [12.]$$

Put eqn.-12 in eqn.-11.

$$(F * I)[i,j] = \sum_{k,l} I[i - k, j - l] \ (F_1[k] \times F_2[l]) \qquad [13.]$$

If we apply definition of convolution on previous equation, we get.

$$(F * I)[i,j] = \sum_{l} \left\{ \sum_{k} I[i - k, j - l] \ F_1[k] \right\} F_2[l] \qquad [14.]$$

Hence $F * I$ can be written as: -

$$F * I = F2 * (F1 * I) \qquad [15.]$$

Hence Proved.

1.d) <u>**Number of multiplications involved in part (a) and part (b).**</u>

- If we count the number of multiplications made in 1.a) step 3.
  $$6 * 4 = 24$$

- If we count the number of multiplications made in 1.b) step 3 and step 6
  $$(2 * 8) + (2 * 6) = 28$$

Here we get a greater number of multiplications for two 1-D filters as compared to single 2-D direct filter.

1.e) **<u>Number of multiplications for general case:</u>**
<u>I is an $M_1 \times N_1$ image, and F is an $M_2 \times N_2$ separable filter.</u>

Filter size of $= M_2 * N_2$

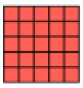

Figure 1: $M_2 * N_2$ filter

Image size of $= (M_1 * N_1)$

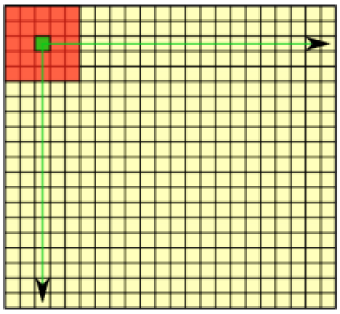

Figure 2: $(M_1 * N_1)$

Resulting image size $= (M_1 - M_2 + 1) * (N_1 - N_2 + 1)$

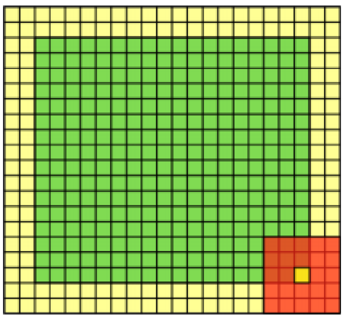

Figure 3: $= (M_1 - M_2 + 1) * (N_1 - N_2 + 1)$

I. Number of Multiplication made to get resulting Image with 2-D filter.

- Multiplications for getting each pixel of resulting image = $M_2 * N_2$
- Number of Pixels in resulting image = $(M_1 - M_2 + 1) * (N_1 - N_2 + 1)$
- Hence, total number of multiplications made =

$$(M_1 - M_2 + 1) * (N_1 - N_2 + 1) * M_2 * N_2 \quad [16.]$$

II. For separable filter F can be broken into 2, 1-D filters.

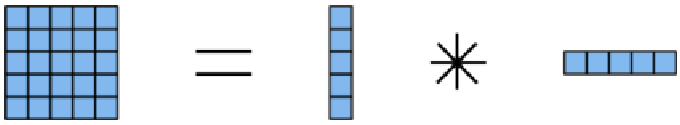

Figure 4: separated filters

now, each 1-D filter can be passed over the image one by one.

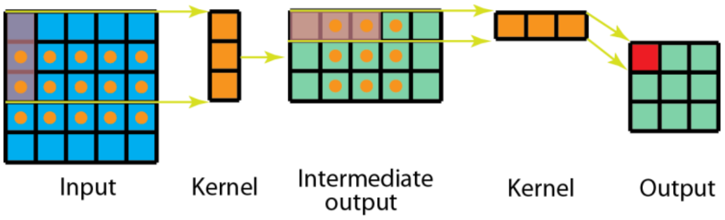

Figure 5: two 1-D convolutions in sequence

step 1 Number of Multiplication made with 1st 1-D filter.

$$= (M_1 - M_2 + 1) * N_1 * M_2$$

Note: - As, we can see the size of matrix after passing of 1st 1-D filter is reduced to $(M_1 - M_2 + 1) * N_1$

step 2 Number of Multiplication made with 2nd 1-D filter.

$$= (M_1 - M_2 + 1) * (N_1 - N_2 + 1) * N_2$$

step 3 Total number of Multiplication with two, 1-D filters.

$$= (M_1 - M_2 + 1) * N_1 * M_2 + (M_1 - M_2 + 1) * (N_1 - N_2 + 1) * N_2 \qquad [17.]$$

III. Efficiency based on Big-O notation.

For non-separable filter of 2-D size, Complexity is given by.

$$= \mathcal{O}(M_1 * N_1 * M_2 * N_2)$$

For separable 1st, 1-D filter, Complexity is given by $= \mathcal{O}(M_1 * N_1 * M_2)$
For separable 2nd, 1-D filter, Complexity is given by $= \mathcal{O}(M_1 * N_1 * N_2)$

We can see that order of time complexity for 2-D direct filter is $\mathcal{O}(n^4)$, where else for the two 1-D filters time complexity is $\mathcal{O}(n^3)$.

Thus, we can say that generally, convolution with two 1-D filters will have less Complexity as compared single 2-D filter.

Solution-2

2.a) Will the rotated edge be detected using the same Canny edge detector?

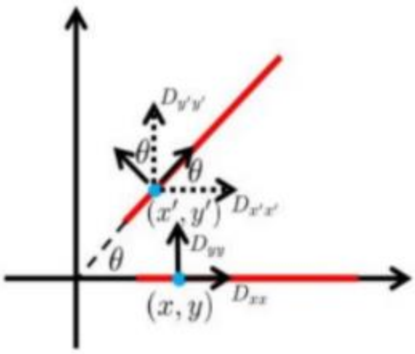

Figure 6: Rotation of image

The image is now rotated at certain angle θ and the relationship between coordinates of new image and originally given will have coordinates as below: -

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

$D_{xx} = derrivate\ in\ x\ direction\ for\ orriginal\ image$

$D_{yy} = derrivate\ in\ y\ direction\ for\ orriginal\ image$
$= 0(as\ no\ vertical\ edge)$

$D_{x\prime x\prime} = derrivate\ in\ x\ direction\ for\ rotated\ image = D_{xx}\ \cos\theta$

$D_{y\prime y\prime} = derrivate\ in\ y\ direction\ for\ rotated\ image = D_{xx}\ \sin\theta$

Case-1 Magnitudes in original image: -

$$\sqrt{D_{xx}^2 + D_{yy}^2} = \sqrt{D_{xx}^2 + 0} = D_{xx}$$

Case-2 Magnitudes in rotated image: -

$$\sqrt{D_{x'x'}^2 + D_{y'y'}^2} = \sqrt{(D_{xx}\ \cos\theta)^2 + (D_{xx}\ \sin\theta)^2}$$
$$= \sqrt{{D_{xx}}^2 \times (\cos^2\theta + \sin^2\theta)} = D_{xx}$$

Magnitudes in both the cases are exactly same. Hence, we proved mathematically that rotated edge will be detected using the same Canny edge detector.

2.b) **How would you adjust the threshold to address Broken Edges and spurious edges problem?**

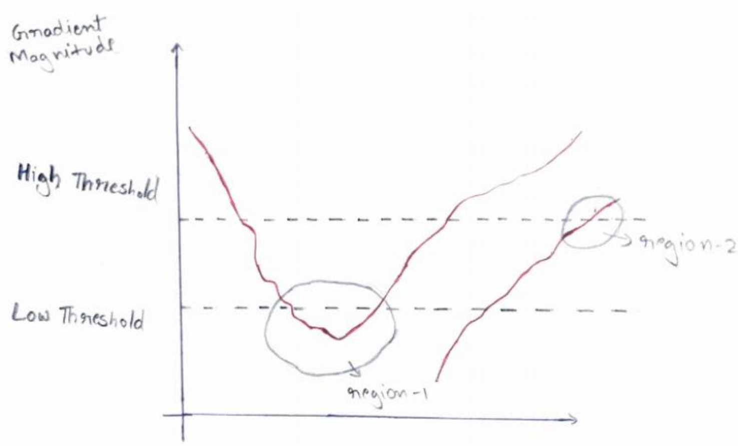

Figure 7 gradient magnitudes with low High-threshold and high Low-Threshold.

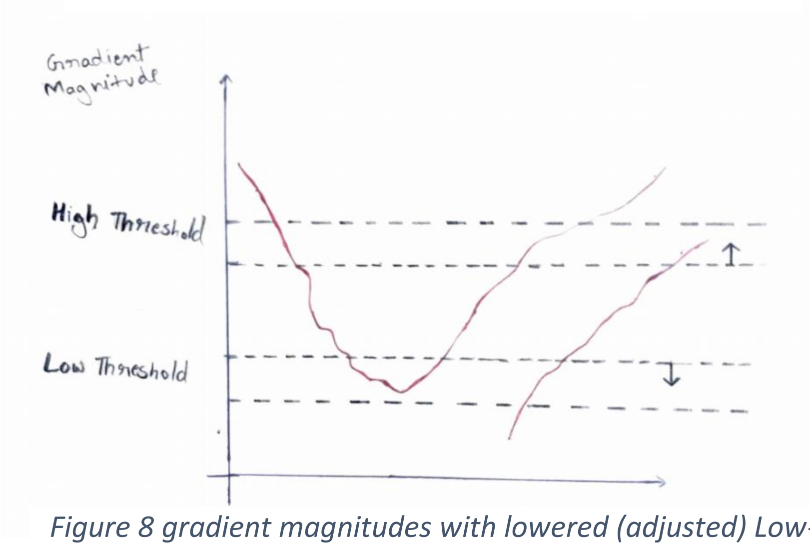

Figure 8 gradient magnitudes with lowered (adjusted) Low-threshold and higher (adjusted) High-Threshold.

Broken Edges: - The reason for breaking of long edges into short segments separated by gaps is because of higher Low-Threshold. As we can see in figure-7, region-1 that the part of gradient magnitude slips below Low-Threshold and thus part of long edges with low magnitude disappear. This issue can be solved by lowering the Low-Threshold slightly, taking care that the lowering low-threshold does not affect classification of non-edge pixels as shown in figure-8.

Spurious Edges: - The reason for spurious edges is low High-Threshold. In figure-7, region-2 we can see that the gradient magnitude value for non-edge pixels lie above high threshold making the resulting feature map with spurious edges. This can be resolved by marking the High-Threshold slightly higher, while also taking care that edge pixels are not wrongly classified as non-edge pixel as shown in figure-8.