# CSE 428

Image Processing

## Assignment - I

Name : Udoy Saha

ID : 23341134

Section : 02

Submitted to:

Saiful Bari Iftu [SDQ]

Submission date:

5 - October - 2024

Ans to the ques no :- 1

$\boxed{a}$

Here, input image has, width, $w = 4$
height, $h = 4$

$\therefore$ size $= 4 \times 4$

kernel size $= 3 \times 3$

If the height and width remains the same, lets consider the new height and width $h'$ and $w'$.

So, $h' = \frac{h - k + 2P}{S} + 1$

$\Rightarrow h = \frac{h - k + 2P}{S} + 1 \quad [\because h' = h]$

$\Rightarrow S.h = h - k + 2P + S$

$\Rightarrow 2P = Sh - h + k - S$

$\therefore P = \frac{S.h - h + k - S}{2} = \frac{1 \times 4 - 4 + 3 - 1}{2}$

$= \frac{4 - 4 + 3 - 1}{2}$

$= 1$

Same for width, the padding will be 1.

(Ans)

$\boxed{b}$

Input image = $\begin{bmatrix} 6 & 2 & 1 & 9 \\ 2 & 6 & 2 & 4 \\ 1 & 3 & 9 & 5 \\ 2 & 0 & 1 & 2 \end{bmatrix}$

Kernel = $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$

$\therefore$ Blurred image (without clipping) = $\begin{bmatrix} 2.94 & 3.8125 \\ 3.94 & 4.75 \end{bmatrix}$

$\therefore$ After rounding,

Blurred image = $\begin{bmatrix} 3 & 4 \\ 4 & 5 \end{bmatrix}$

(Ans)

$\boxed{c}$

Here, the mask $= \begin{bmatrix} 3 & 4 \\ 4 & 5 \end{bmatrix}$

Masked part from the original image $= \begin{bmatrix} 0 & 2 \\ 9 & 9 \end{bmatrix}$

$\therefore$ Unsharp mask = Original image − mask

$$= \begin{bmatrix} 0 & 2 \\ 9 & 9 \end{bmatrix} - \begin{bmatrix} 3 & 4 \\ 4 & 5 \end{bmatrix}$$

$$= \begin{bmatrix} -3 & -2 \\ 5 & 4 \end{bmatrix}$$

(Ans)

~~Adding the unsharp mask with~~

$\boxed{d}$

Sharpened image = Original image + k × Unsharp mask

$$= \begin{bmatrix} 0 & 2 \\ 9 & 9 \end{bmatrix} + 1 \times \begin{bmatrix} -3 & -2 \\ 5 & 4 \end{bmatrix}$$

$$= \begin{bmatrix} -3 & 0 \\ 14 & 13 \end{bmatrix}$$

After clipping $= \begin{bmatrix} 0 & 0 \\ 14 & 13 \end{bmatrix}$

(Ans)

—o—x—o—

## Ans to the ques no:-2

The picture has many edges in different directions. Only detecting horizontal and vertical edges wont be sufficient here. Also the image contains a lot of grain.

We have two good option for this image to detect edges. First one is Robinson Compass filter which detects edges by convolving multiple filters dedicated for different directions. Second one is Laplacian operator. But, since the image contains a lot of grain, the edges produced by Laplacian operator wont be so good.

Therefore, ~~B~~ the best technique will be applying Robinson Compass operators. Employing these filters will produce the best result:

1. Horizontal = $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

2. Vertical = $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

3. North-East = $\begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$

4. North-west = $\begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ 2 & -1 & 0 \end{bmatrix}$

—o—x—o—

Ans to the ques no:-3

This picture shows an Eagle, and it is noticeable that there is not much grain present in the picture. The color of one region is almost consistent throughout the region.

A Robinson Compass Operator would work nicely in this case. But to cover different directions, we will need to employ at-least 4 filters. Since, the grains present is considerably less in this picture, the Laplacian operator would also work great in this case.

So, a slightly modified or enhanced version of Laplacian Positive operator will be enough to cover all edges. The filter is as follows:-

$$\begin{bmatrix} 0 & 4 & 0 \\ 4 & -16 & 4 \\ 0 & 4 & 0 \end{bmatrix}$$

Ans to the ques no:- 4

| Layer | Input Dimension | Filter Size | # Filters or # Neurons | Padding | Output Dimension | # Params |
| --- | --- | --- | --- | --- | --- | --- |
| Conv 1 | 128×128×3 | 7×7 | 8 | 2 | 126×126×8 | 8×(7×7×3+1) = 1184 |
| MaxPool 1 | 126×126×8 | 2×2 | — | 0 | 63×63×8 | 0 |
| Conv 2 | 63×63×8 | 5×5 | 16 | 2 | 63×63×16 | 16×(5×5×8+1) = 3216 |
| MaxPool 2 | 63×63×16 | 2×2 | — | 0 | 31×31×16 | 0 |
| Conv 3 | 31×31×16 | 3×3 | 32 | 0 | 29×29×32 | 32×(3×3×16+1) = 4640 |
| AvgPool 3 | 29×29×32 | 4×4 | — | 0 | 7×7×32 | 0 |
| Flatten | 7×7×32 | — | — | — | 1568 | 0 |
| FC | 1568 | — | 256 | — | 256 | (1568+1)×256 = 401664 |
| FC | 256 | — | 128 | — | 128 | (256+1)×128 = 32896 |
| FC (output) | 128 | — | 4 | — | 4 | (128+1)×4 = 516 |

a

In the final layer,

number of classes = 4

activation function = softmax

b

Done in table.

c

Done in the table.

## d

| Layer | Input dimension | Output dimension |
|---|---|---|
| Conv 1 | 32×128×128×3 | 32×126×126×8 |
| MaxPool 1 | 32×126×126×8 | 32×63×63×8 |
| Conv 2 | 32×63×63×8 | 32×63×63×16 |
| Maxpool 2 | 32×63×63×16 | 32×31×31×16 |
| Conv 3 | 32×31×31×16 | 32×29×29×32 |
| AvgPool 3 | 32×29×29×32 | 32×7×7×32 |
| Flatten | 32×7×7×32 | 32×1568 |
| FC | 32×1568 | 32×256 |
| FC | 32×256 | 32×128 |
| FC (output) | 32×128 | 32×4 |

## e

Yes, adding BatchNormalization layers will change the total number of trainable parameters. It will add:

| | | | |
|---|---|---|---|
| Batch Normalization Conv1 → | 8×4 | = | 32 |
| " Conv 2 → | 16×4 | = | 64 |
| " Conv 3 → | 32×4 | = | 128 |
| " FC → | 256×4 | = | 1024 |
| " FC → | 128×4 | = | 512 |
| | | | 1760 |

So, it will add 1760 more parameters. (Ans)

—o— x —o—