Assignment 2 - Image processing

TDT4196 Visual Computing Fundamentals

Lars Martin Moe

November, 2019

Task 1: CNN Theory

a)

The equations for calculating the height and width are given as:

$$H_{i+1} = (H_i - F_H + 2P_H)/S_H + 1 \tag{1}$$
$$W_{i+1} = (W_i - F_W + 2P_W)/S_W + 1 \tag{2}$$

The given parameters are:

$$\begin{aligned} F_W &= F_H = 5 \\ S_W &= S_H = 1 \\ W_1 &= W_2 = W \\ H_1 &= H_2 = H \end{aligned}$$

By inserting these into eq (1) and eq (2) and rearranging, we get:

$$\begin{aligned} W_2 &= (W_1 - F_W + 2P_W)/S_W + 1 \\ W &= W - 4 + 2P_W \\ P_W &= 2 \end{aligned}$$

We will get the same result for the height, given the symmetry. This means that we should be padding with $P_W = P_H = 2$ for width and height, respectively.

b)

The given parameters for the image and first layer are:

$$\begin{aligned} P_H &= P_W = 0 \\ S_H &= S_W = 1 \\ H_1 &= W_1 = 512 \\ H_2 &= W_2 = 504 \end{aligned}$$

By using the same method as in a), we get:

$$
\begin{aligned}
H_2 &= (H_1 - F_H + 2 \cdot P_H)/S_H + 1 \\
504 &= (512 - F_H + 2 \cdot 0)/1 + 1 \\
F_H &= 512 + 1 - 504 = 9
\end{aligned}
$$

and

$$
\begin{aligned}
W_2 &= (W_1 - F_W + 2 \cdot P_W)/S_W + 1 \\
504 &= (512 - F_W + 2 \cdot 0)/1 + 1 \\
F_W &= 512 + 1 - 504 = 9
\end{aligned}
$$

Thus the kernel has a spatial dimension of 9×9. This is a square kernel with odd dimensions, as assumed.

c)

For the subsampling with neighbourhoods, we have the parameters:

$$
\begin{aligned}
H_1 &= W_1 = 512 \\
P_H &= P_W = 0 \\
F_H &= F_W = 2 \\
S_H &= S_W = 2
\end{aligned}
$$

Using the same method as in a) and b), we get:

$$
\begin{aligned}
H_2 &= (H_1 - F_H + 2 \cdot P_H)/S_H + 1 \\
&= (512 - 2 + 2 \cdot 0)/2 + 1 = 256 \\
W_2 &= (W_1 - F_W + 2 \cdot P_W)/S_W + 1 \\
&= (512 - 2 + 2 \cdot 0)/2 + 1 = 256
\end{aligned}
$$

With all the new parameters:

$$
\begin{aligned}
H_1 &= W_1 = 256 \\
P_H &= P_W = 0 \\
F_H &= F_W = 9 \\
S_H &= S_W = 1
\end{aligned}
$$

We then calculate the spatial dimensions of the pooled feature maps in the first layer:

$$\begin{aligned} H_2 &= (H_1 - F_H + 2 \cdot P_H)/S_H + 1 \\ &= (256 - 9 + 2 \cdot 0)/1 + 1 = 247 \\ W_2 &= (W_1 - F_W + 2 \cdot P_W)/S_W + 1 \\ &= (256 - 9 + 2 \cdot 0)/1 + 1 = 247 \end{aligned}$$

Which gives us spatial dimensions of 247×247

d)

Each feature map has a depth of 1. This gives an output with a depth of 12 if we have 12 feature maps in the first layer.

e)

The given parameters are:

$$\begin{aligned} H_1 = W_1 &= 247 \\ P_H = P_W &= 0 \\ F_H = F_W &= 3 \\ S_H = S_W &= 1 \end{aligned}$$

We get:

$$\begin{aligned} H_2 &= (H_1 - F_H + 2 \cdot P_H)/S_H + 1 \\ &= (247 - 3 + 2 \cdot 0)/1 + 1 = 245 \\ W_2 &= (W_1 - F_W + 2 \cdot P_W)/S_W + 1 \\ &= (247 - 3 + 2 \cdot 0)/1 + 1 = 245 \end{aligned}$$

The sizes of the feature maps in the second layer are 245×245

f)

For the input layer 0 to layer 3, the table gives us:

$$
\begin{aligned}
H_0 &= W_0 = 32 \\
P_H^{Max} &= P_W^{Max} = 0 \\
F_H^{Max} &= F_W^{Max} = 2 \\
S_H^{Max} &= S_W^{Max} = 2 \\
P_H^{Conv} &= P_W^{Conv} = 2 \\
F_H^{Conv} &= F_W^{Conv} = 5 \\
S_H^{Conv} &= S_W^{Conv} = 1 \\
C_0 &= 3 \\
C_1 &= 32 \\
C_2 &= 64 \\
C_3 &= 128 \\
C_4 &= 64
\end{aligned}
$$

Furthermore, the flatten layer outputs a vector of size $4 \cdot 4 \cdot 128$ from the $4 \times 4 \times 128$ image from layer 3. This is then mapped with 64 hidden units in layer 4, and 10 in layer 5.

The amount of parameters in each layer can be calculated with:

$$
n_{i+1} = F_H \cdot F_W \cdot C_i \cdot C_{i+1} + C_{i+1} \tag{3}
$$

For the first 3 layers we get:

$$
\begin{aligned}
n_1 &= 5 \cdot 5 \cdot 3 \cdot 32 + 32 = 2432 \\
n_2 &= 5 \cdot 5 \cdot 32 \cdot 64 + 64 = 51264 \\
n_3 &= 5 \cdot 5 \cdot 64 \cdot 128 + 128 = 204928
\end{aligned}
$$

After flattening, and then summing all the parameters we get:

$$
\begin{aligned}
n_4 &= 4 \cdot 4 \cdot 128 \cdot 64 + 64 = 131136131073 \\
n_5 &= 64 \cdot 10 \cdot 1 + 1 = 641 \\
n_{network} &= \sum_{i=1}^{5} n_i = 390275
\end{aligned}
$$

Task 2: Programming

a)

The original and max-pooled images are shown in fig. 1 to fig. 4. For the checkerboard, we can only see the white parts of the checkerboard after max-pooling. This is because the intensity changes too quickly for it to show up in the max-pooled image. There is always at least one white part in every 2x2 sub-group of the original image, which means that the white value will always be chosen.

Figure 1: The original image of the cat Chelsea.

Figure 2: The image of the cat Chelsea after max-pooling

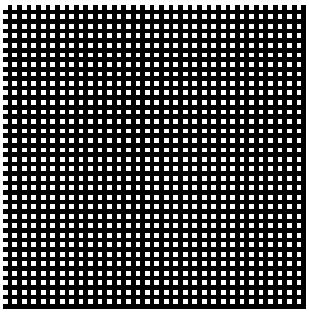

Figure 3: The original image of the checkerboard.

Figure 4: The image of the checkerboard after max-pooling.

b)

The plot for the training and validation loss during training with 4 epochs is show in fig. 5. The final validation loss was 0.06920924860981122 and the final validation accuracy was 0.9748. The final training loss was 0.06584434289182371 and the final training accuracy was 0.9777197766666667. Because the final losses are so close to each other, we can't conclude that any over-training has occurred.

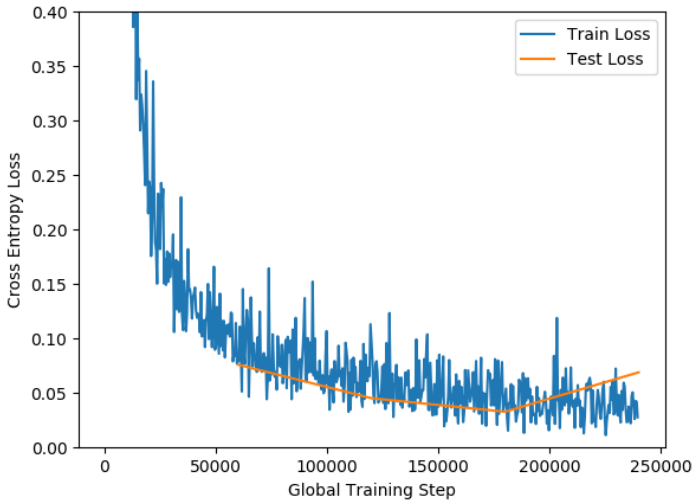

Figure 5: Training and test cross entropy loss with 4 epochs.

c)

The plot for the training and validation loss during training with 10 epochs is show in fig. 6. The final validation loss was 0.0213717243510866962 and the final validation accuracy was 0.9927. The final training loss was 0.011386962217215271 and the final training accuracy was 0.9968776666666669.

In this case, the final validation loss is nearly double that of the training loss. This indicates that some degree of overfitting has occurred.

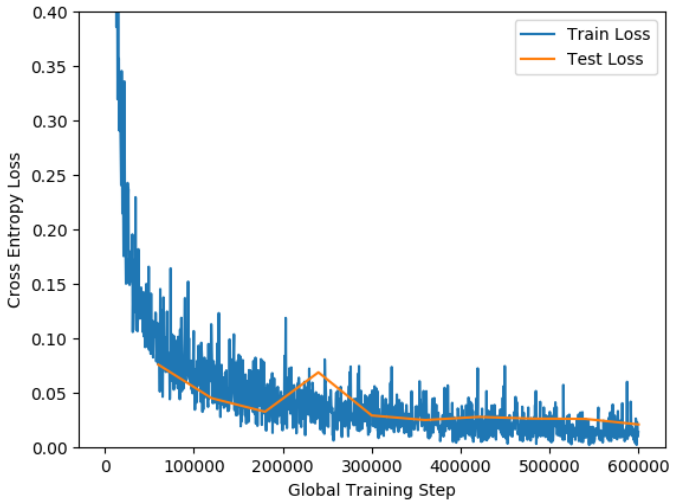

Figure 6: Training and test cross entropy loss with 10 epochs.

d)

The resulting visualization is shown in fig. 7:

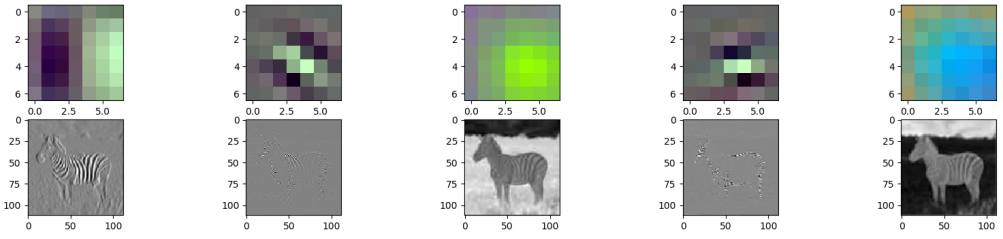

Figure 7: Visualization of filter and grey-scale activation of the filter.

e)

Filter 5: This filter looks similar to the Sobel-x kernel. This kernel is used for edge detection in the x-axis, which is very apparent with how the vertical stripes for the zebra are shown in the grey-scale activation of the filer. We can see that the filter has brighter (green) weights on the right and darker weights (purple) on the left.

Filter 8: This filter extracts white diagonal lines that go from left down to the right in the image. We can see that this corresponds to parts of the outline of the zebra.

Filter 19; This filter highlights the green and yellow parts of the image with focus on the bottom right parts, and extracts the green ground from the image (grass).

Filter 22: This filter looks similar to the Sobel-y kernel, as it primarily extracts edges in the y-axis.This kernel seems to also extract some diagonal features.

Filter 34: This filter highlights blue parts of the image with focus on the bottom right parts of the image. For the zebra picture this seems to primarily be the sky.

Task 3: Filtering in the frequency domain theory

a)

The convolution theorem is given by:

$$\mathcal{F}\{f * g\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{g\} \tag{4}$$

This tells us that we can can multiply the Fourier transform of the individual signals to get the Fourier transform of the convolution of the two. The steps then become:

- Find the Fourier transform for each of the signals
- Use pointwise multiplication on the results
- Find the convolution of the signals by using the inverse Fourier transform

b)

For a low-pass filter, signals with a frequency higher than the cut-off frequency should be close to zero, while we wish to preserve signals with lower frequencies. Depending on how "sharp" we want the cut-off to be, we can choose either a cone or a cylinder shape for the filter.

A high-pass filter works in a similar way, but removes frequencies where a low-pass filter wants to include them, and includes the higher frequencies. In short: High-pass filters allows high frequencies to pass, while low-pass filters allows low frequencies to pass.

c)

a) is a high-pass filter which suppresses low frequencies. It can be seen that the filter allows lower frequencies along the y-axis, and blocks them along the x-axis.

b) is a high-pass filter. We can see that the kernel suppresses low frequencies in the centre, while higher frequencies have a larger value. This filter also suppresses signals without higher frequencies in both directions (x and y).

c) is a low-pass filter, as the high frequencies in the outer part are suppressed, and the low frequencies in the centre are included.

Task4: Programming

a)

The filtered images and the before/after amplitude of the transformation is shown in fig. 8 for the high-pass filter, and fig. 9 for the low-pass filter.

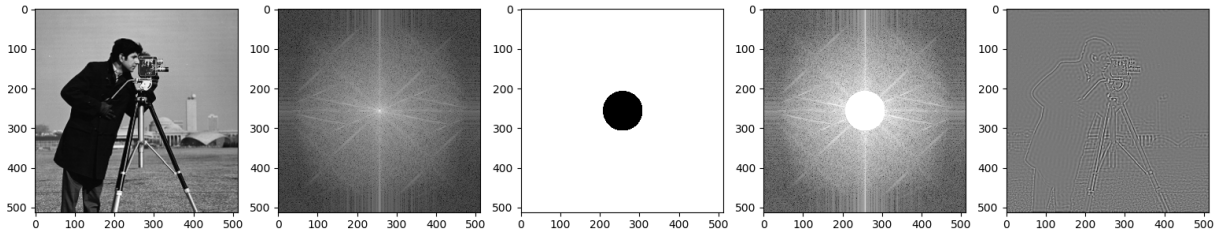

Figure 8: Original image, FFT of image, the FFT kernel, convolution and resulting image for the high-pass filter.

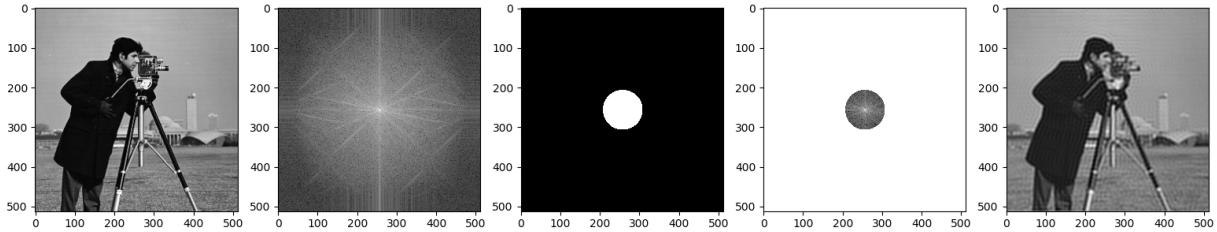

Figure 9: Original image, FFT of image, the FFT kernel, convolution and resulting image for the low-pass filter.

b)

The filtered images and the before/after amplitude of the transformation is shown in fig. 10 for the gaussian filter, and fig. 11 for the sobel-x filter.

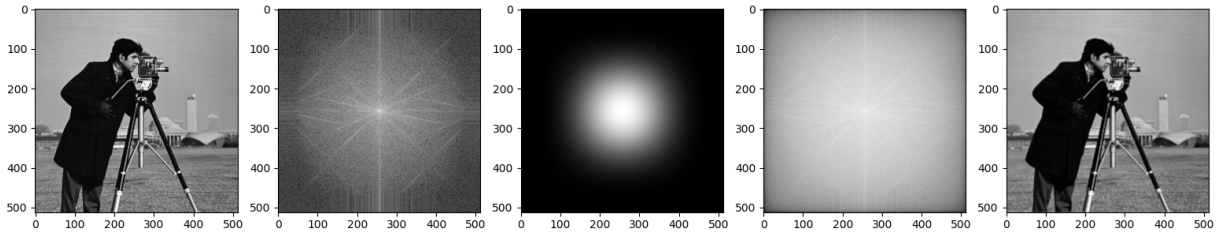

Figure 10: Original image, FFT of image, the FFT kernel, convolution and resulting image for the gaussian filter.

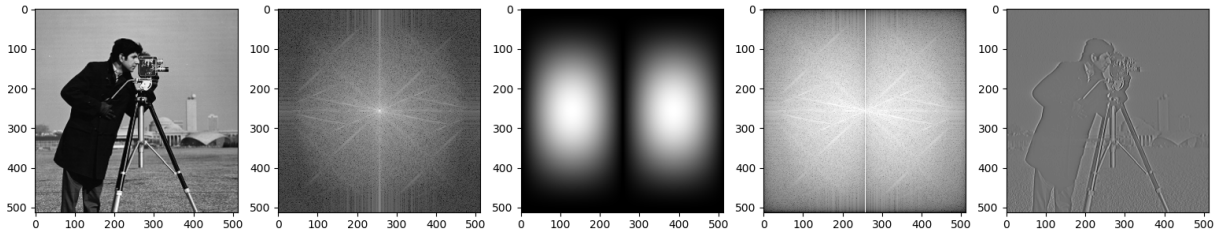

Figure 11: Original image, FFT of image, the FFT kernel, convolution and resulting image for the sobel-x filter.

c)

The original image and the result of the filtering can be seen in fig. 12 and fig. 13. This filter seems like a notch filter, which is a very narrow bandstop filter which removes certain frequencies from the image. In this case the diagonal scanlines have been removed from the image.

Figure 12: Original image of a clown.

Figure 13: Result of the filtering with the unknown kernel on the clown-image.

d)

The original and sharpened inamge of a moon can be seen in fig. 14

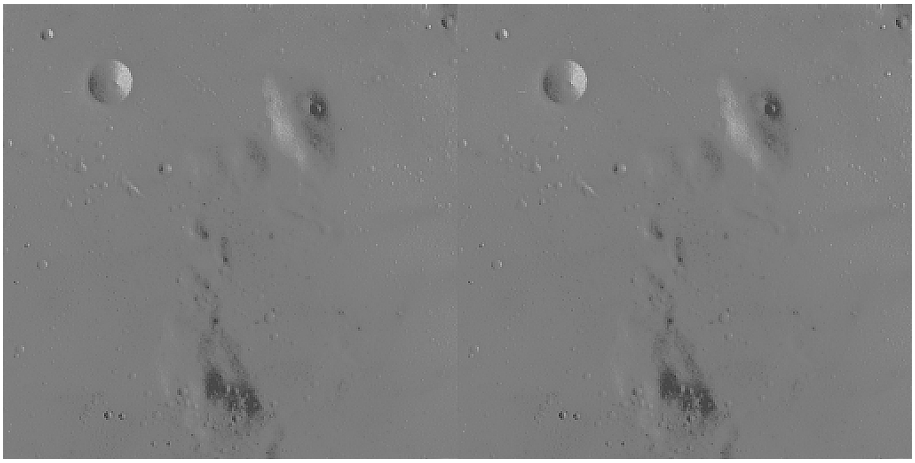

Figure 14: Original image to the left and the result of the sharpening to the right.