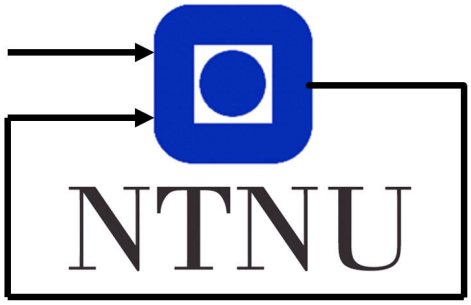
Image Processing - Assignment 2

Group 3
Martin Eek Gerhardsen

October 25



Department of Engineering Cybernetics

Contents

1 Convolutional Neural Networks

1.1 Task 1: Theory

a)

Listing the equations for calculating the resulting height and width:

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

Now defining:

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

Then we get for the width:

$$\begin{aligned} W_2 &= (W_1 - F_W + 2P_W)/S_W + 1 \\ W - 1 &= W - 5 + 2P_W \\ 2P_W &= 4 \\ P_W &= 2 \end{aligned}$$

And for height:

$$\begin{aligned} H_2 &= (H_1 - F_H + 2P_H)/S_H + 1 \\ H - 1 &= H - 5 + 2P_H \\ 2P_H &= 4 \\ P_H &= 2 \end{aligned}$$

So we should therefore be padding with $P_H = 2$ and $P_W = 2$.

c)

Now defining:

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

Then using the same equations as above, we calculate:

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

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

We also knew from the task that the kernel was supposed to be square, and its dimensions odd, which we can confirm from the result. The kernel must therefore be of the size $9x9$.

d)

Starting with subsampling with neighbourhoods, defining:

$$
\begin{aligned}
H_1 &= W_1 = 512 \\
S_H &= S_W = 2 \\
P_H &= P_W = 0 \\
F_H &= F_W = 2 \\
H_2 &= ? \\
W_2 &= ?
\end{aligned}
$$

Then we get:

$$
\begin{aligned}
H_2 &= (H_1 - F_H + 2 * P_H)/S_H + 1 \\
&= (512 - 2 + 2 * 0)/2 + 1 = 255 \\
W_2 &= (W_1 - F_W + 2 * P_W)/S_W + 1 \\
&= (512 - 2 + 2 * 0)/2 + 1 = 255
\end{aligned}
$$

Then, using the same information utilized above, we define:

$$
\begin{aligned}
H_1 &= W_1 = 255 \\
S_H &= S_W = 1 \\
P_H &= P_W = 0 \\
F_H &= F_W = 9 \\
H_2 &= ? \\
W_2 &= ?
\end{aligned}
$$

Then we calculate:

$$\begin{aligned} H_2 &= (H_1 - F_H + 2 * P_H)/S_H + 1 \\ &= (255 - 9 + 2 * 0)/1 + 1 = 247 \\ W_2 &= (W_1 - F_W + 2 * P_W)/S_W + 1 \\ &= (255 - 9 + 2 * 0)/1 + 1 = 247 \end{aligned}$$

Then the spatial dimensions of the pooled feature maps in the first layer would be $247x247$.

e)

Uncertain about this question. Wouldn't it just be the same as the number of feature maps? In that case, 12...

f)

Defining:

$$\begin{aligned} H_1 &= W_1 = 247 \\ S_H &= S_W = 1 \\ P_H &= P_W = 0 \\ F_H &= F_W = 3 \\ H_2 &= ? \\ W_2 &= ? \end{aligned}$$

Then:

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

Therefore, the sizes of the feature map of the second layer would be $245x245$.

g)

Defining the 0-layer as the input layer, so that the other layers can be defined incrementally as well as being the given layer in the table. Furthermore, superscript will be used to specify the layer type. I will use C for Conv2D, M for MaxPool2D

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

The flattening-layer will then convert the $4x4x128$ result from layer 3 to a vector of size $4 * 4 * 128 = 2048$. Then this is mapped with to 64 hidden units, and finally to the 10, representing the ten different classes (digits).

$$
n_{i+1} = F_H^C * F_W^C * C_i * C_{i+1} + C_{i+1} = 25 * C_i * C_{i+1} + C_{i+1} \tag{3}
$$

To calculate the number of parameters, we look at each filters individually. Using eq. (3), we can calculate the number of parameters for each layer:

$$
\begin{aligned}
n_1 &= 25 * 3 * 32 + 32 = 2432 \\
n_2 &= 25 * 32 * 64 + 64 = 51264 \\
n_3 &= 25 * 64 * 128 + 128 = 204928 \\
n_4 &= 4 * 4 * 128 * 64 + 1 = 131073 \\
n_5 &= 64 * 10 + 1 = 641 \\
n &= \sum_{i=1}^{5} n_i = 390338
\end{aligned}
$$

I also calculate the spatial dimensions of the layers. This wasn't strictly

needed to calculate the number of parameters, but was useful for task 2:

$$H_1^C = (H_0 - F_H^C + 2 * P_H^C)/S_H^C + 1 = (32 - 5 + 2 * 2)/1 + 1 = 32$$
$$H_1^M = (H_1^C - F_H^M + 2 * P_H^M)/S_H^M + 1 = (32 - 2 + 2 * 0)/2 + 1 = 16$$
$$H_2^C = (H_1^M - F_H^C + 2 * P_H^C)/S_H^C + 1 = (16 - 5 + 2 * 2)/1 + 1 = 16$$
$$H_2^M = (H_2^C - F_H^M + 2 * P_H^M)/S_H^M + 1 = (16 - 2 + 2 * 0)/2 + 1 = 8$$
$$H_3^C = (H_2^M - F_H^C + 2 * P_H^C)/S_H^C + 1 = (8 - 5 + 2 * 2)/1 + 1 = 8$$
$$H_3^M = (H_3^C - F_H^M + 2 * P_H^M)/S_H^M + 1 = (8 - 2 + 2 * 0)/2 + 1 = 4$$
$$W_1^C = (W_0 - F_W^C + 2 * P_W^C)/S_W^C + 1 = (32 - 5 + 2 * 2)/1 + 1 = 32$$
$$W_1^M = (W_1^C - F_W^M + 2 * P_W^M)/S_W^M + 1 = (32 - 2 + 2 * 0)/2 + 1 = 16$$
$$W_2^C = (W_1^M - F_W^C + 2 * P_W^C)/S_W^C + 1 = (16 - 5 + 2 * 2)/1 + 1 = 16$$
$$W_2^M = (W_2^C - F_W^M + 2 * P_W^M)/S_W^M + 1 = (16 - 2 + 2 * 0)/2 + 1 = 8$$
$$W_3^C = (W_2^M - F_W^C + 2 * P_W^C)/S_W^C + 1 = (8 - 5 + 2 * 2)/1 + 1 = 8$$
$$W_3^M = (W_3^C - F_W^M + 2 * P_W^M)/S_W^M + 1 = (8 - 2 + 2 * 0)/2 + 1 = 4$$

1.2 Task 2: Programming

a)

b)

c)

d)

e)

2 Filtering in the Frequency Domain

2.1 Task 3: Theory

a)

From the convolution theorem, we can see that the Fourier transform of a convolution of two signals is multiplication of the Fourier transforms of those individual signals. So a stepwise description would be:

- Find the Fourier transform of the individual signals (i.e. using FFT)
- Pointwise multiply the transformed signals
- Use inverse Fourier transform to find the convolution of the original signals

b)

Low- and high-pass filters tell us which frequencies should be kept, low-pass allows low frequencies to pass, and high-pass allows high frequencies to pass.

This means that low-pass filters should be close to the filter gain (i.e. 1) frequencies closer to the origin than the cut-off frequency (low frequencies), and should be close to zero for higher frequencies than the cut-off frequency. Depending on how we want the filter to behave, the filter can either be simililar to a cylinder or a cone, where cylinder gives a much *harsher* cut-off frequency (no frequencies after cut-off) and cone gives a *smoother* cut-off frequency (allowing, but suppressing some frequencies after cut-off).

The high-pass filter is simply the $K - LPF$, where K is the filter gain (i.e. 1 again) and LPF is the low-pass filter. This implies that the high-pass filter follows the same pattern as the low-pass, but where the low-pass filter includes frequencies, the high-pass filter removes them, and vice versa where the high-pass includes the frequencies.

c)

As the images are shifted, we find the low frequencies in the centre and high around the image. Also using the white parts of the images as high amplitude, and black as low amplitude.

a)

b) This is a high-pass filter, the lower frequencies in the centre are largely suppressed, and the higher frequencies are multiplied with values closer to 1.

c) This is a low-pass filter, the lower frequencies in the centre are largely included, while the higher frequencies further out are

2.2 Task 4: Programming

a)

b)