Worksheet 4: Convolutional Neural Networks by Hand (Matched to Slides)

Goal. This worksheet mirrors the in-class CNN slides. It contains small, concrete forward-pass calculations you can reproduce on the whiteboard: 1D convolution (with stride/padding), 2D convolution, and max/avg pooling. It closes with a mini end-to-end shape-tracking example (conv $\to$ ReLU $\to$ pool $\to$ dense).

1D Convolution (no padding, stride 1)

Setup. Input signal $x = [2,\ 0,\ 3,\ 1,\ -1]$, filter (kernel) $w = [1,\ 0,\ -1]$, stride $s = 1$, no padding. The valid positions are the length-3 windows:

$$x_{1:3} = [2, 0, 3], \qquad x_{2:4} = [0, 3, 1], \qquad x_{3:5} = [3, 1, -1].$$

Dot products (filter reused at each position):

$$\begin{aligned} y_1 &= \langle [2, 0, 3], [1, 0, -1] \rangle = 2 + 0 - 3 = -1, \\ y_2 &= \langle [0, 3, 1], [1, 0, -1] \rangle = 0 + 0 - 1 = -1, \\ y_3 &= \langle [3, 1, -1], [1, 0, -1] \rangle = 3 + 0 - (-1) = 4. \end{aligned}$$

Output: $y = [-1,\ -1,\ 4]$ (length $= 5 - 3 + 1 = 3$).

Key ideas to call out. (i) *Locality*—each output only looks at a short window of x. (ii) *Parameter sharing*—the same w is reused at all positions. (iii) *Output length* (1D, valid conv): $L_{\text{out}} = \left\lfloor \frac{L_{\text{in}} - K}{s} \right\rfloor + 1$.

1D Convolution with Padding and Stride

Setup. Same x and w as above. Now use zero-padding of $p = 1$ on both ends and stride $s = 2$.
Padded input: $\tilde{x} = [0,\ 2,\ 0,\ 3,\ 1,\ -1,\ 0]$ (length 7).
Number of outputs: $L_{\text{out}} = \left\lfloor \frac{7-3}{2} \right\rfloor + 1 = 3$.

Windows and dots:

$$\begin{aligned} \tilde{x}_{1:3} = [0, 2, 0] &\Rightarrow y_1 = 0 \cdot 1 + 2 \cdot 0 + 0 \cdot (-1) = 0, \\ \tilde{x}_{3:5} = [0, 3, 1] &\Rightarrow y_2 = 0 \cdot 1 + 3 \cdot 0 + 1 \cdot (-1) = -1, \\ \tilde{x}_{5:7} = [1, -1, 0] &\Rightarrow y_3 = 1 \cdot 1 + (-1) \cdot 0 + 0 \cdot (-1) = 1. \end{aligned}$$

Output: $y = [0,\ -1,\ 1]$.

Formula to remember. With padding p on each side and stride s, $L_{\text{out}} = \left\lfloor \frac{L_{\text{in}} + 2p - K}{s} \right\rfloor + 1$.

2D Convolution (single-channel image)

Setup. Grayscale input (height=width= 3), valid convolution ($p = 0$), stride $s = 1$:

$$\boldsymbol{X} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & -1 & 0 \\ 2 & 1 & 1 \end{bmatrix}, \qquad \boldsymbol{K} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Output has size $(3 - 2 + 1) \times (3 - 2 + 1) = 2 \times 2$.

$$Y_{1,1} = \langle \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix}, \boldsymbol{K} \rangle = 1 \cdot 1 + 2 \cdot 0 + 0 \cdot 0 + (-1) \cdot (-1) = 2,$$

$$Y_{1,2} = \langle \begin{bmatrix} 2 & 1 \\ -1 & 0 \end{bmatrix}, \boldsymbol{K} \rangle = 2 \cdot 1 + 1 \cdot 0 + (-1) \cdot 0 + 0 \cdot (-1) = 2,$$

$$Y_{2,1} = \langle \begin{bmatrix} 0 & -1 \\ 2 & 1 \end{bmatrix}, \boldsymbol{K} \rangle = 0 \cdot 1 + (-1) \cdot 0 + 2 \cdot 0 + 1 \cdot (-1) = -1,$$

$$Y_{2,2} = \langle \begin{bmatrix} -1 & 0 \\ 1 & 1 \end{bmatrix}, \boldsymbol{K} \rangle = (-1) \cdot 1 + 0 \cdot 0 + 1 \cdot 0 + 1 \cdot (-1) = -2.$$

Output: $\boldsymbol{Y} = \begin{bmatrix} 2 & 2 \\ -1 & -2 \end{bmatrix}$.

Channels and parameter count. For RGB input (3 channels), a 3×3 filter has $3 \cdot 3 \cdot 3 = 27$ weights (plus an optional bias). Each filter produces one feature map; F filters $\Rightarrow$ F output channels.

Pooling Layers (downsampling features)

Setup. Input feature map (4×4). Apply 2×2 pooling with stride 2.

$$\boldsymbol{A} = \begin{bmatrix} 2 & 1 & 3 & 2 \\ 0 & 4 & 5 & 1 \\ 1 & 2 & 0 & 3 \\ 2 & 1 & 2 & 0 \end{bmatrix}.$$

Max pooling takes the maximum in each non-overlapping 2×2 block: $\begin{bmatrix} \max\{2,1,0,4\} & \max\{3,2,5,1\} \\ \max\{1,2,2,1\} & \max\{0,3,2,0\} \end{bmatrix} = \begin{bmatrix} 4 & 5 \\ 2 & 3 \end{bmatrix}$.

Average pooling takes the mean of each block: $\begin{bmatrix} (2+1+0+4)/4 & (3+2+5+1)/4 \\ (1+2+2+1)/4 & (0+3+2+0)/4 \end{bmatrix} = \begin{bmatrix} 1.75 & 2.75 \\ 1.50 & 1.25 \end{bmatrix}$.

When to emphasise. Pooling reduces spatial size (here $4 \times 4 \to 2 \times 2$) and introduces small translation invariance by summarising patches.

Mini CNN: Shape & Parameter Tracking

CNNs as matrix multiplications

$$\textbf{Input } X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}, \qquad \textbf{Filter } K = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}, \qquad \text{(stride 1, valid)}$$

$$\underbrace{\begin{bmatrix} y_{11} \\ y_{12} \\ y_{21} \\ y_{22} \end{bmatrix}}_{\text{vec}(Y)} = \underbrace{\begin{bmatrix} k_{11} & k_{12} & 0 & k_{21} & k_{22} & 0 & 0 & 0 & 0 \\ 0 & k_{11} & k_{12} & 0 & k_{21} & k_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{11} & k_{12} & 0 & k_{21} & k_{22} & 0 \\ 0 & 0 & 0 & 0 & k_{11} & k_{12} & 0 & k_{21} & k_{22} \end{bmatrix}}_{\text{Toeplitz } (4\times 9)} \underbrace{\begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{31} \\ x_{32} \\ x_{33} \end{bmatrix}}_{\text{vec}(X)}$$

$$\underbrace{\begin{bmatrix} y_{11} \\ y_{12} \\ y_{21} \\ y_{22} \end{bmatrix}}_{\text{vec}(Y)} = \underbrace{\begin{bmatrix} x_{11} & x_{12} & x_{21} & x_{22} \\ x_{12} & x_{13} & x_{22} & x_{23} \\ x_{21} & x_{22} & x_{31} & x_{32} \\ x_{22} & x_{23} & x_{32} & x_{33} \end{bmatrix}}_{\text{im2col}(4\times 4)} \underbrace{\begin{bmatrix} k_{11} \\ k_{12} \\ k_{21} \\ k_{22} \end{bmatrix}}_{\text{vec}(K)}$$

$$\text{where} \quad \begin{aligned} y_{11} &= k_{11}x_{11} + k_{12}x_{12} + k_{21}x_{21} + k_{22}x_{22}, \\ y_{12} &= k_{11}x_{12} + k_{12}x_{13} + k_{21}x_{22} + k_{22}x_{23}, \\ y_{21} &= k_{11}x_{21} + k_{12}x_{22} + k_{21}x_{31} + k_{22}x_{32}, \\ y_{22} &= k_{11}x_{22} + k_{12}x_{23} + k_{21}x_{32} + k_{22}x_{33}. \end{aligned}$$