COMP433 Assignment 3 Theory,

Omar Elmasaoudi 40255123

Question 1

$$X = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$F = \begin{bmatrix} 1 & 4 & 1 \\ 0 & 2 & 1 \\ 2 & 4 & 2 \end{bmatrix}.$$

(a)

No padding, stride = 1.

Output size:

$$H_{\text{out}} = 5 - 3 + 1 = 3, \qquad W_{\text{out}} = 5 - 3 + 1 = 3.$$

Example for the top-left element:

$$Y_{0,0} = 1 \cdot 1 + 1 \cdot 4 + 1 \cdot 1 + 0 \cdot 0 + 0 \cdot 2 + 1 \cdot 1 + 0 \cdot 2 + 0 \cdot 4 + 0 \cdot 2 = 7.$$

Doing this for all valid positions gives

$$Y^{(a)} = \begin{bmatrix} 7 & 9 & 8 \\ 7 & 9 & 10 \\ 8 & 5 & 8 \end{bmatrix}.$$

(b)

No padding, stride = 2.

Now $S = 2$, still no padding. Output size:

$$H_{\text{out}} = \left\lfloor \frac{5-3}{2} + 1 \right\rfloor = 2, \qquad W_{\text{out}} = 2.$$

The resulting feature map is

$$Y^{(b)} = \begin{bmatrix} 7 & 8 \\ 8 & 8 \end{bmatrix}.$$

(c)

1-pixel padding, stride $= 1$.

We pad X with one layer of zeros around it ($P = 1$, $S = 1$). New effective input is 7×7, so output size is 5×5:

$$H_{\text{out}} = 5, \qquad W_{\text{out}} = 5.$$

The convolution output is

$$Y^{(c)} = \begin{bmatrix} 3 & 5 & 9 & 10 & 6 \\ 5 & 7 & 9 & 8 & 3 \\ 6 & 7 & 9 & 10 & 7 \\ 9 & 8 & 5 & 8 & 6 \\ 8 & 7 & 3 & 7 & 3 \end{bmatrix}.$$

(d)

MaxPooling, kernel size 2, stride 2.

Max pooling is applied directly to the input image X (no filter). With kernel size 2 and stride 2:

$$H_{\text{out}} = \left\lfloor \frac{5-2}{2} + 1 \right\rfloor = 2, \qquad W_{\text{out}} = 2.$$

We take the maximum in each 2×2 window:

$$X = \left[\begin{array}{cc} \boxed{\begin{matrix} 1 & 1 \\ 0 & 0 \end{matrix}} & \boxed{\begin{matrix} 1 & 1 \\ 1 & 1 \end{matrix}} \\ \boxed{\begin{matrix} 0 & 0 \\ 1 & 1 \end{matrix}} & \boxed{\begin{matrix} 0 & 0 \\ 1 & 1 \end{matrix}} \end{array}\right] \Rightarrow Y^{(d)} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

(e)

No padding, stride $= 1$, dilation $= 2$.

Dilation $D = 2$ (on a 3×3 filter) gives an effective kernel size

$$K_{\text{eff}} = 1 + (K-1)D = 1 + 2 \cdot 2 = 5,$$

so with no padding and stride 1 on a 5×5 image, the output is just a single value (1×1).

We sample every other pixel:

$$\text{positions used in } X : \begin{bmatrix} X_{0,0} & X_{0,2} & X_{0,4} \\ X_{2,0} & X_{2,2} & X_{2,4} \\ X_{4,0} & X_{4,2} & X_{4,4} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

Then

$$Y^{(e)}_{0,0} = 1 \cdot 1 + 1 \cdot 4 + 0 \cdot 1 + 0 \cdot 0 + 0 \cdot 2 + 0 \cdot 1 + 1 \cdot 2 + 0 \cdot 4 + 1 \cdot 2 = 9.$$

So

$$Y^{(e)} = [\,9\,].$$

Question 2

Input image size:

$$64 \times 64 \times 3.$$

Architecture:

- Conv1: $k = 3 \times 3 \times 32,\ p = 1,\ s = 1$
- Conv2: $k = 7 \times 7 \times 64,\ p = 0,\ s = 1$
- Pooling: MaxPool 2×2, no overlap ($s = 2$)
- Conv3: $k = 3 \times 3 \times 128,\ p = 1,\ s = 1$
- Conv4: $k = 7 \times 7 \times 256,\ p = 0,\ s = 1$
- Pooling: MaxPool 2×2, no overlap ($s = 2$)
- Flatten
- Dense: size $= 10$

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2p - k}{s} + 1 \right\rfloor, \qquad W_{\text{out}} = \left\lfloor \frac{W_{\text{in}} + 2p - k}{s} + 1 \right\rfloor.$$

Layer-by-layer dimensions (in $H \times W \times N$):

$$\begin{aligned}
\text{Input} &: 64 \times 64 \times 3 \\
\text{Conv1} &: \underbrace{64 \times 64}_{(64+2\cdot 1-3)/1+1} \times 32 \\
\text{Conv2} &: \underbrace{58 \times 58}_{(64+0-7)/1+1} \times 64 \\
\text{Pooling1} &: \underbrace{29 \times 29}_{\left\lfloor \frac{58-2}{2}+1 \right\rfloor} \times 64 \\
\text{Conv3} &: \underbrace{29 \times 29}_{(29+2\cdot 1-3)/1+1} \times 128 \\
\text{Conv4} &: \underbrace{23 \times 23}_{(29+0-7)/1+1} \times 256 \\
\text{Pooling2} &: \underbrace{11 \times 11}_{\left\lfloor \frac{23-2}{2}+1 \right\rfloor} \times 256 \\
\text{Flatten} &: 1 \times 1 \times (11 \cdot 11 \cdot 256) = 1 \times 1 \times 30\,976 \\
\text{Dense} &: 1 \times 1 \times 10.
\end{aligned}$$

So the output volumes for each layer are:

Conv1:	$64 \times 64 \times 32$
Conv2:	$58 \times 58 \times 64$
Pooling1:	$29 \times 29 \times 64$
Conv3:	$29 \times 29 \times 128$
Conv4:	$23 \times 23 \times 256$
Pooling2:	$11 \times 11 \times 256$
Flatten:	$1 \times 1 \times 30{,}976$
Dense:	$1 \times 1 \times 10$

Question 3

- Input images: $32 \times 32 \times 3$ (RGB).
- Conv1: 3×3 filters, 16 filters, stride 1, padding 1.
- Conv2: 3×3 filters, 16 filters, stride 1, no padding.
- Average pooling: 2×2, stride 2.
- Fully-connected (FC) layer with ReLU.

(a) Conv1 keeps the spatial resolution because of padding $P = 1$:

$$H_{\text{out}} = \frac{32 + 2 \cdot 1 - 3}{1} + 1 = 32, \qquad W_{\text{out}} = 32.$$

Depth is 16. Therefore, the number of neurons is

$$\#\text{neurons in Conv1} = 32 \times 32 \times 16 = 16{,}384.$$

Each neuron corresponds to one spatial location and one filter.

(b) Each Conv1 filter has size 3×3 and spans all 3 input channels, so:

$$\#\text{weights per filter} = 3 \times 3 \times 3 = 27.$$

There are 16 filters, so

$$\#\text{weights in Conv1} = 27 \times 16 = 432.$$

Each filter has one bias term:

$$\#\text{biases in Conv1} = 16.$$

(c) The input to Conv2 is $32 \times 32 \times 16$. With $K = 3$, $S = 1$, $P = 0$:

$$H_{\text{out}} = \frac{32 - 3}{1} + 1 = 30, \qquad W_{\text{out}} = 30.$$

Depth is again 16, so

$$\#\text{neurons in Conv2} = 30 \times 30 \times 16 = 14{,}400.$$

(d) Each Conv2 filter:

$$\#\text{weights per filter} = 3 \times 3 \times 16 = 144.$$

With 16 filters:

$$\#\text{weights in Conv2} = 144 \times 16 = 2{,}304.$$

Again, one bias per filter:

$$\#\text{biases in Conv2} = 16.$$

(e) After Conv2, the feature maps are $30 \times 30 \times 16$. Average pooling with 2×2 and stride 2:

$$H_{\text{out}} = \left\lfloor \frac{30-2}{2} + 1 \right\rfloor = \left\lfloor \frac{28}{2} + 1 \right\rfloor = 15,$$

$$W_{\text{out}} = \left\lfloor \frac{30-2}{2} + 1 \right\rfloor = 15.$$

Depth remains 16. So the encoder output volume is

$$15 \times 15 \times 16.$$

Flattening this gives

$$\#\text{features to FC} = 15 \times 15 \times 16 = 3{,}600.$$

(f) Using multiple convolutional layers with small filters is better because it reduces the number of parameters compared to a single large filter with the same receptive field, which helps prevent overfitting. Additionally stacking layers introduces multiple non-linearities, allowing the network to learn hierarchical features.

Question 4

- The input to logistic regression is the flattened encoder output, which has dimensionality 3,600.
- For binary classification, logistic regression applies a linear transformation followed by a sigmoid:

  $$z = \mathbf{w}^\top \mathbf{h} + b, \qquad \hat{y} = \sigma(z),$$

  where $\mathbf{h} \in R^{3600}$ is the flattened CNN output.

So the logistic regression model takes a 3,600-dimensional input.

Question 5

$$X = \begin{bmatrix} -1 & 0 & 1 \\ 2 & 3 & 4 \\ 5 & 6 & 7 \\ 8 & 9 & 10 \end{bmatrix}, \qquad F = \begin{bmatrix} 2 & 1 \\ 0 & -1 \end{bmatrix}.$$

(a) Output size:

$$H_{\text{out}} = 4 - 2 + 1 = 3, \qquad W_{\text{out}} = 3 - 2 + 1 = 2.$$

Using cross-correlation:

$$Y_{i,j} = \sum_{u=0}^{1} \sum_{v=0}^{1} X_{i+u,\, j+v} F_{u,v}.$$

Computations:

$$Y = \begin{bmatrix} -5 & -3 \\ 1 & 3 \\ 7 & 9 \end{bmatrix}.$$

(b) Apply $\text{ReLU}(z) = \max(0, z)$ element-wise:

$$Y^{(\text{ReLU})} = \begin{bmatrix} 0 & 0 \\ 1 & 3 \\ 7 & 9 \end{bmatrix}.$$

(c) After 2×2 average pooling, stride 2.
With $H = 3$, $W = 2$, $K = 2$, $S = 2$, the pooled output is 1×1: we take the top-left 2×2 block

$$\begin{bmatrix} 0 & 0 \\ 1 & 3 \end{bmatrix},$$

whose average is

$$\frac{0 + 0 + 1 + 3}{4} = 1.$$

So the final pooled activation map is

$$Y^{(\text{pool})} = [\,1\,].$$

Question 6

We have a simple 1D RNN with:

$$h_t = w_{\text{in}} x_t + w_{\text{hh}} h_{t-1}, \qquad y = w_{\text{hy}} h_L,$$

and loss

$$\ell = \frac{1}{2}(y - y_r)^2.$$

(a) By definition (no activation function):

$$h_t = w_{\text{in}} x_t + w_{\text{hh}} h_{t-1}, \qquad t = 1, \dots, L.$$

(b) Forward pass for $x = [0.3, 0.4, 0.2]$.
Given

$$x = [x_1, x_2, x_3] = [0.3, 0.4, 0.2], \quad h_0 = 0.5, \quad w_{\text{in}} = w_{\text{hh}} = w_{\text{hy}} = 0.2,$$

we get:

$$h_1 = 0.2 \cdot 0.3 + 0.2 \cdot 0.5 = 0.06 + 0.10 = 0.16,$$

$$h_2 = 0.2 \cdot 0.4 + 0.2 \cdot 0.16 = 0.08 + 0.032 = 0.112,$$

$$h_3 = 0.2 \cdot 0.2 + 0.2 \cdot 0.112 = 0.04 + 0.0224 = 0.0624,$$

$$y = w_{\text{hy}} h_3 = 0.2 \cdot 0.0624 = 0.01248.$$

(c) Gradients
Let $L = 3$. First note:

$$\frac{\partial \ell}{\partial y} = y - y_r.$$

Since $y = w_{\text{hy}} h_3$,

$$\frac{\partial \ell}{\partial w_{\text{hy}}} = (y - y_r)\, h_3,$$

$$\frac{\partial \ell}{\partial h_3} = (y - y_r)\, w_{\text{hy}}.$$

The recurrence is

$$h_1 = w_{\text{in}} x_1 + w_{\text{hh}} h_0, \quad h_2 = w_{\text{in}} x_2 + w_{\text{hh}} h_1, \quad h_3 = w_{\text{in}} x_3 + w_{\text{hh}} h_2.$$

We can write h_3 explicitly in terms of h_0, x_1, x_2, x_3:

$$h_1 = w_{\text{in}} x_1 + w_{\text{hh}} h_0,$$

$$h_2 = w_{\text{in}} x_2 + w_{\text{hh}} h_1 = w_{\text{in}} x_2 + w_{\text{hh}} (w_{\text{in}} x_1 + w_{\text{hh}} h_0),$$

$$h_3 = w_{\text{in}} x_3 + w_{\text{hh}} h_2 = w_{\text{in}} x_3 + w_{\text{hh}} \Big[w_{\text{in}} x_2 + w_{\text{hh}} (w_{\text{in}} x_1 + w_{\text{hh}} h_0) \Big].$$

Hence

$$h_3 = h_0 w_{\text{hh}}^3 + w_{\text{in}} w_{\text{hh}}^2 x_1 + w_{\text{in}} w_{\text{hh}} x_2 + w_{\text{in}} x_3.$$

Therefore

$$y = w_{\text{hy}} h_3 = w_{\text{hy}} \left(h_0 w_{\text{hh}}^3 + w_{\text{in}} w_{\text{hh}}^2 x_1 + w_{\text{in}} w_{\text{hh}} x_2 + w_{\text{in}} x_3 \right).$$

Now:

$$\frac{\partial \ell}{\partial w_{\text{hy}}} = (y - y_r)\, h_3,$$

with h_3 as above.

For w_{in} and w_{hh}, we first noted that

$$\frac{\partial \ell}{\partial h_3} = (y - y_r) w_{\text{hy}}, \quad \frac{\partial \ell}{\partial w_{\text{in}}} = \frac{\partial \ell}{\partial h_3} \frac{\partial h_3}{\partial w_{\text{in}}}, \quad \frac{\partial \ell}{\partial w_{\text{hh}}} = \frac{\partial \ell}{\partial h_3} \frac{\partial h_3}{\partial w_{\text{hh}}}.$$

And when we differentiate h_3:

$$\frac{\partial h_3}{\partial w_{\text{in}}} = w_{\text{hh}}^2 x_1 + w_{\text{hh}} x_2 + x_3,$$

$$\frac{\partial h_3}{\partial w_{\text{hh}}} = 3 h_0 w_{\text{hh}}^2 + 2 w_{\text{hh}} w_{\text{in}} x_1 + w_{\text{in}} x_2.$$

Therefore:

$$\boxed{\frac{\partial \ell}{\partial w_{\text{in}}} = (y - y_r)\, w_{\text{hy}} \left(w_{\text{hh}}^2 x_1 + w_{\text{hh}} x_2 + x_3 \right)}$$

$$\boxed{\frac{\partial \ell}{\partial w_{\text{hh}}} = (y - y_r)\, w_{\text{hy}} \left(3 h_0 w_{\text{hh}}^2 + 2 w_{\text{hh}} w_{\text{in}} x_1 + w_{\text{in}} x_2 \right)}$$

$$\boxed{\frac{\partial \ell}{\partial w_{\text{hy}}} = (y - y_r) \left(h_0 w_{\text{hh}}^3 + w_{\text{in}} w_{\text{hh}}^2 x_1 + w_{\text{in}} w_{\text{hh}} x_2 + w_{\text{in}} x_3 \right)}$$

where y itself is

$$y = w_{\text{hy}} \left(h_0 w_{\text{hh}}^3 + w_{\text{in}} w_{\text{hh}}^2 x_1 + w_{\text{in}} w_{\text{hh}} x_2 + w_{\text{in}} x_3 \right).$$

All expressions use only $y_r, h_0, w_{\text{in}}, w_{\text{hh}}, w_{\text{hy}}, x_1, x_2, x_3$.

(d) We already have from part b:

$$y = 0.01248.$$

Substitute $x_1 = 0.3,\ x_2 = 0.4,\ x_3 = 0.2,\ h_0 = 0.5,\ w_{\text{in}} = w_{\text{hh}} = w_{\text{hy}} = 0.2,\ y_r = 0.4$ into the gradient formulas. This gives us

$$\frac{\partial \ell}{\partial w_{\text{in}}} \approx -0.022631168, \qquad \frac{\partial \ell}{\partial w_{\text{hh}}} \approx -0.012710656, \qquad \frac{\partial \ell}{\partial w_{\text{hy}}} \approx -0.024181248.$$

Using gradient descent

$$w^{\text{new}} = w^{\text{old}} - \eta \frac{\partial \ell}{\partial w}, \quad \eta = 0.15,$$

we obtain:

$$\begin{aligned} w_{\text{in}}^{\text{new}} &= 0.2 - 0.15 \cdot (-0.022631168) \approx 0.2033947, \\ w_{\text{hh}}^{\text{new}} &= 0.2 - 0.15 \cdot (-0.012710656) \approx 0.2019066, \\ w_{\text{hy}}^{\text{new}} &= 0.2 - 0.15 \cdot (-0.024181248) \approx 0.2036272. \end{aligned}$$

Question 7

Each x_i is projected into:

$$q_i = W_Q x_i \quad \text{(query)}, \qquad k_i = W_K x_i \quad \text{(key)}, \qquad v_i = W_V x_i \quad \text{(value)},$$

where W_Q, W_K, W_V are the learned matrices.
To compute the output representation for x_1:

1. Compute the query for position 1:

$$q_1 = W_Q x_1.$$

2. Compute keys and values for all positions $j = 1, 2, 3$:

$$k_j = W_K x_j, \qquad v_j = W_V x_j.$$

3. Compute attention scores between q_1 and each k_j:

$$e_{1j} = \frac{q_1^\top k_j}{\sqrt{d_k}},$$

where d_k is the dimensionality of the keys.

4. Normalize scores with softmax to obtain attention weights:

$$\alpha_{1j} = \frac{\exp(e_{1j})}{\sum_{m=1}^{3} \exp(e_{1m})}, \qquad j = 1, 2, 3.$$

5. The output representation for position 1 is the weighted sum of values:

$$z_1 = \sum_{j=1}^{3} \alpha_{1j}\, v_j.$$

Here:

- Queries q_1 ask "what am I looking for?" from position 1.
- Keys k_j encode "what content do I have?" at each position j.
- Values v_j carry the information that will be mixed together.

The self-attention mechanism lets z_1 integrate information from x_1, x_2, x_3 according to the learned attention weights α_{1j}.