Due Date : February 23rd (11pm), 2022

Instructions

- *For all questions, show your work!*
- *Use LaTeX when writing your answers. We recommend using this assignment latex code as a template. You may reuse most of the notation shorthands, equations and/or tables. See the assignment policy on the course website for more details.*
- *Submit your answers electronically via Gradescope.*
- ***TAs for this assignment are Matthew Scicluna and Akram Erraqabi.***

Question 1 (3-3-3-4-3-4-3)**.** Consider training a standard feed-forward neural network. For the purposes of this question we are interested in a single iteration of SGD on a single training example : $(\boldsymbol{x}, y)$. We denote $f(\boldsymbol{x}, \boldsymbol{\theta})$ as the output of the neural network with model parameters $\boldsymbol{\theta}$. Now let's say g is the output activation function and $a(\boldsymbol{x}, \boldsymbol{\theta})$ is the pre-activation network output such that $f(\boldsymbol{x}, \boldsymbol{\theta}) = g(a(\boldsymbol{x}, \boldsymbol{\theta}))$.

1.1 Assuming the network's goal is to do binary classification (with the detailed structure above), what would be an appropriate activation function for the output layer, i.e. what would be an appropriate function g ?

Answer. A good output activation function would be the **sigmoid** :

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

1.2 What does the output represent under this activation function ?

Answer. The output of the neural network using the sigmoid $\sigma(z)$ as the output activation function gives us an output between 0 and 1 that represents the probability that the example $\boldsymbol{x}$ belongs to class 1 meaning $p(y = 1|\boldsymbol{x})$.

1.3 Let $L_{CE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y)$ be cross-entropy loss, express it as a function of $f(\boldsymbol{x}, \boldsymbol{\theta})$ and y.

Answer. The general formula for the cross entropy loss is defined as :

$$L_{CE}(f(\boldsymbol{x}; \boldsymbol{\theta}), y)) = -\sum_{i}^{C} y_i \cdot log(f(\boldsymbol{x}; \boldsymbol{\theta})_i) \qquad \text{where } C \text{ is the number of class.}$$

However, the question asked to define it in function of $f(\boldsymbol{x}; \boldsymbol{\theta})$ and y. Here $y \in \{0, 1\}$ so, we can modify the equation of the general cross entropy to a version for 2 classes called binary cross entropy loss. The **binary cross entropy loss** is defined as :

$$L_{CE}(f(\boldsymbol{x}; \boldsymbol{\theta}), y)) = -y \cdot log(f(\boldsymbol{x}; \boldsymbol{\theta})) - (1 - y) \cdot log(1 - f(\boldsymbol{x}; \boldsymbol{\theta}))$$

1.4 Compute the partial derivative $\frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})}$.

Answer.

$$\begin{aligned}\frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} &= \frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial f(\boldsymbol{x},\boldsymbol{\theta})} \cdot \frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} \\ &= \frac{\partial}{\partial f(\boldsymbol{x},\boldsymbol{\theta})} - y \cdot log(f(\boldsymbol{x};\boldsymbol{\theta})) - (1-y) \cdot log(1-f(\boldsymbol{x};\boldsymbol{\theta})) \cdot \frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} \\ &= (\frac{-y}{f(\boldsymbol{x},\boldsymbol{\theta})} + \frac{1-y}{1-f(\boldsymbol{x},\boldsymbol{\theta})}) \cdot \frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})}\end{aligned}$$

$$\begin{aligned}\frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} &= \frac{\partial}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} \frac{1}{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}} \\ &= \frac{\partial}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} (1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})})^{-1} \\ &= \frac{-1}{(1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})})^2} \cdot e^{-a(\boldsymbol{x},\boldsymbol{\theta})} \cdot -1 \\ &= \frac{e^{-a(\boldsymbol{x},\boldsymbol{\theta})}}{(1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})})^2} \\ &= \frac{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}-1}{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}} \cdot \frac{1}{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}} \\ &= (\frac{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}}{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}} - \frac{1}{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}}) \cdot \frac{1}{1+e^{-a(\boldsymbol{x},\boldsymbol{\theta})}} \\ &= \sigma(a(\boldsymbol{x},\boldsymbol{\theta})) \cdot (1-\sigma(a(\boldsymbol{x},\boldsymbol{\theta})))\end{aligned}$$

Thefore, as $f(\boldsymbol{x},\boldsymbol{\theta}) = \sigma(a(\boldsymbol{x},\boldsymbol{\theta}))$

$$\begin{aligned}\frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} &= \frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial f(\boldsymbol{x},\boldsymbol{\theta})} \cdot \frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} \\ &= (\frac{-y}{f(\boldsymbol{x},\boldsymbol{\theta})} + \frac{1-y}{1-f(\boldsymbol{x},\boldsymbol{\theta})}) \cdot (\sigma(a(\boldsymbol{x},\boldsymbol{\theta})) \cdot (1-\sigma(a(\boldsymbol{x},\boldsymbol{\theta})))) \\ &= \frac{-y \cdot \sigma(a(\boldsymbol{x},\boldsymbol{\theta})) \cdot (1-\sigma(a(\boldsymbol{x},\boldsymbol{\theta}))}{\sigma(a(\boldsymbol{x},\boldsymbol{\theta}))} + \frac{(1-y) \cdot \sigma(a(\boldsymbol{x},\boldsymbol{\theta})) \cdot (1-\sigma(a(\boldsymbol{x},\boldsymbol{\theta}))}{1-\sigma(a(\boldsymbol{x},\boldsymbol{\theta}))} \\ &= (-y \cdot (1-\sigma(a(\boldsymbol{x},\boldsymbol{\theta}))) + ((1-y) \cdot \sigma(a(\boldsymbol{x},\boldsymbol{\theta}))) \\ &= \boxed{\sigma(a(\boldsymbol{x},\boldsymbol{\theta})) - y}\end{aligned}$$

1.5 Let $L_{MSE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y)$ be the mean-squared error, express it as a function of $f(\boldsymbol{x}, \boldsymbol{\theta})$ and y (multiplicative factors can be ignored).

Answer. The general formula for the mean squared error loss is defined as :

$$L_{MSE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y) = \frac{1}{N} \cdot \sum_{i=1}^{N} (y^{(i)} - f(\boldsymbol{x}, \boldsymbol{\theta})^{(i)})^2$$

However, the question asked to define it in function of $f(\boldsymbol{x}; \boldsymbol{\theta})$ and y. Here we have only one example,so we can modify the equation of the general MSE loss to a simpler version defined as :

$$L_{MSE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y) = (y - f(\boldsymbol{x}, \boldsymbol{\theta}))^2$$

1.6 Compute the partial derivative $\frac{\partial L_{MSE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y)}{\partial a(\boldsymbol{x}, \boldsymbol{\theta})}$.

Answer.

$$\begin{aligned}
\frac{\partial L_{MSE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y)}{\partial a(\boldsymbol{x}, \boldsymbol{\theta})} &= \frac{\partial L_{MSE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y)}{\partial f(\boldsymbol{x}, \boldsymbol{\theta})} \cdot \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial a(\boldsymbol{x}, \boldsymbol{\theta})} \\
&= \frac{\partial}{\partial f(\boldsymbol{x}, \boldsymbol{\theta})} (y - f(\boldsymbol{x}, \boldsymbol{\theta}))^2 \cdot \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial a(\boldsymbol{x}, \boldsymbol{\theta})} \\
&= -2.(y - f(\boldsymbol{x}, \boldsymbol{\theta})) \cdot \frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial a(\boldsymbol{x}, \boldsymbol{\theta})} \\
&= -2.(y - f(\boldsymbol{x}, \boldsymbol{\theta})) \cdot \sigma(a(\boldsymbol{x}, \boldsymbol{\theta})).(1 - \sigma(a(\boldsymbol{x}, \boldsymbol{\theta}))) \qquad \text{using 1.4}
\end{aligned}$$

1.7 Based on your answers to the above questions, what would be the more appropriate loss function for binary classification and why ?

Anwser.

The more appropriate loss function will be the **cross-entropy** (binary cross-entropy) because the main problem with MSE loss is that it's not punishing enough misclassifications as output of the neural network are value between 0 and 1 and the target is 0 or 1 which will lead to small distances between the prediction and the target. Cross entropy, punish well misclassifications, as show in the result of exercise **1.4**. Furthermore, because binary-cross entropy loss expect to have $p(y = 1|\boldsymbol{x})$ as input which is what we are doing with the sigmoid but the MSE loss does not make that assumption.

Question 2 (4-4-5-6)**.** Recall the definition of the softmax function : $S(\boldsymbol{x})_i = e^{\boldsymbol{x}_i}/\sum_j e^{\boldsymbol{x}_j}$.

2.1 Show that softmax is translation-invariant, that is : $S(\boldsymbol{x} + c) = S(\boldsymbol{x})$, where c is a scalar constant.

Answer.

$$\begin{aligned} S(\boldsymbol{x}+c) &= \frac{e^{\boldsymbol{x}_i+c}}{\sum_j e^{\boldsymbol{x}_j+c}} = \frac{e^{\boldsymbol{x}_i}\cdot e^{c}}{\sum_j e^{\boldsymbol{x}_j}\cdot e^{c}} \\ &= \frac{e^{\boldsymbol{x}_i}\cdot e^{c}}{e^{c}\cdot\sum_j e^{\boldsymbol{x}_j}} = \frac{e^{\boldsymbol{x}_i}}{\sum_j e^{\boldsymbol{x}_j}} = S(\boldsymbol{x}) \end{aligned}$$

■.

2.2 Let $\boldsymbol{x}$ be a 2-dimensional vector. One can represent a 2-class categorical probability using softmax $S(\boldsymbol{x})$. Show that $S(\boldsymbol{x})$ can be reparameterized using sigmoid function, i.e. $S(\boldsymbol{x}) = [\sigma(z), 1-\sigma(z)]^\top$ where z is a scalar function of $\boldsymbol{x}$.

Answer.

$$S(\boldsymbol{x})_1 = \sigma(z) = \frac{e^{\boldsymbol{x}_1}}{e^{\boldsymbol{x}_1}+e^{\boldsymbol{x}_2}}$$
$$S(\boldsymbol{x})_2 = 1-\sigma(z) = \frac{e^{\boldsymbol{x}_2}}{e^{\boldsymbol{x}_1}+e^{\boldsymbol{x}_2}}$$

$$\begin{aligned} \sigma(z) &= \frac{1}{1+e^{-z}} = \frac{e^{\boldsymbol{x}_1}}{e^{\boldsymbol{x}_1}\cdot(1+e^{-z})} \\ &= \frac{e^{\boldsymbol{x}_1}}{e^{\boldsymbol{x}_1}+e^{\boldsymbol{x}_1}\cdot e^{-z}} = \frac{e^{\boldsymbol{x}_1}}{e^{\boldsymbol{x}_1}+e^{\boldsymbol{x}_1-z}} \\ &= \frac{e^{\boldsymbol{x}_1}}{e^{\boldsymbol{x}_1}+e^{\boldsymbol{x}_1-\boldsymbol{x}_1+\boldsymbol{x}_2}} = \frac{e^{\boldsymbol{x}_1}}{e^{\boldsymbol{x}_1}+e^{\boldsymbol{x}_2}} \end{aligned}$$

So,

$$\boldsymbol{x}_2 = \boldsymbol{x}_1 - z \Leftrightarrow z = \boldsymbol{x}_1 - \boldsymbol{x}_2$$

Therefore, using such a z :

$$\begin{aligned} 1-\sigma(z) &= 1-\frac{1}{1+e^{-z}} = \frac{1+e^{-z}}{1+e^{-z}} - \frac{1}{1+e^{-z}} \\ &= \frac{1+e^{-z}-1}{1+e^{-z}} = \frac{e^{-z}}{1+e^{-z}} = \frac{e^{-\boldsymbol{x}_1+\boldsymbol{x}_2}}{1+e^{-\boldsymbol{x}_1+\boldsymbol{x}_2}} \\ &= \frac{e^{-\boldsymbol{x}_1+\boldsymbol{x}_2}}{1+e^{-\boldsymbol{x}_1+\boldsymbol{x}_2}}\cdot\frac{e^{\boldsymbol{x}_1}}{e^{\boldsymbol{x}_1}} = \frac{e^{\boldsymbol{x}_2}}{e^{\boldsymbol{x}_1}+e^{\boldsymbol{x}_2}} = S(\boldsymbol{x})_2 \end{aligned}$$

We have been showed that $S(\boldsymbol{x})$ can be reparameterized using sigmoid function, i.e. $S(\boldsymbol{x}) = [\sigma(z), 1-\sigma(z)]^\top$ where $z = \boldsymbol{x}_1 - \boldsymbol{x}_2$.

■.

2.3 Let $\boldsymbol{x}$ be a K-dimensional vector ($K \geq 2$). Show that $S(\boldsymbol{x})$ can be represented using $K-1$ parameters, i.e. $S(\boldsymbol{x}) = S([0, y_1, y_2, ..., y_{K-1}]^\top)$, where y_i is a scalar function of $\boldsymbol{x}$ for $i \in \{1, ..., K-1\}$.

Answer.

First, I should recall that :

$$\boldsymbol{x} \in \mathbb{R}^K \qquad \text{s.t} \quad K \geq 2$$
$$S(\boldsymbol{x})_i = \frac{e^{\boldsymbol{x}_i}}{\sum_j e^{\boldsymbol{x}_j}}$$

Then, as $S(\boldsymbol{x})$ is translation invariant :

$$\begin{aligned} S(\boldsymbol{x}) &= S(\boldsymbol{x} - \boldsymbol{x}_1) = S([0, \boldsymbol{x}_2 - \boldsymbol{x}_1, \boldsymbol{x}_3 - \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k - \boldsymbol{x}_1]^\top) \\ &= S([0, y_1, y_2, \ldots, y_{K-1}]^\top) \qquad\qquad \text{s.t } y_i = \boldsymbol{x}_i - \boldsymbol{x}_1 \text{ with } 1 \leq i \leq K-1 \end{aligned}$$

■.

2.4 Show that the Jacobian of the softmax function $J_{\text{softmax}}(\boldsymbol{x})$ can be expressed as : $\mathbf{Diag}(\boldsymbol{p}) - \boldsymbol{p}\boldsymbol{p}^\top$, where $\boldsymbol{p} = S(\boldsymbol{x})$.

Answer.

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \frac{\partial S(\boldsymbol{x})}{\partial \boldsymbol{x}} \in \mathbb{R}^{n \times n}$$

$$\frac{\partial S(\boldsymbol{x})}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial S(\boldsymbol{x})_1}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial S(\boldsymbol{x})_1}{\partial \boldsymbol{x}_n} \\ \vdots & & \vdots \\ \frac{\partial S(\boldsymbol{x})_n}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial S(\boldsymbol{x})_n}{\partial \boldsymbol{x}_n} \end{bmatrix}$$

$$\begin{aligned} \frac{\partial S(\boldsymbol{x})_i}{\partial \boldsymbol{x}_j} &= \frac{\partial}{\partial \boldsymbol{x}_j} \frac{e^{\boldsymbol{x}_i}}{\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}}} \\ &= \frac{\partial}{\partial \boldsymbol{x}_j} e^{\boldsymbol{x}_i} \left(\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}} \right)^{-1} \\ &= (\frac{\partial}{\partial \boldsymbol{x}_j} e^{\boldsymbol{x}_i}) \cdot \left(\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}} \right)^{-1} + e^{\boldsymbol{x}_i} \cdot \frac{\partial}{\partial \boldsymbol{x}_j} \left(\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}} \right)^{-1} \\ &= \frac{\mathbb{1}_{i=j} e^{\boldsymbol{x}_i}}{\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}}} - \frac{e^{\boldsymbol{x}_i}}{(\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}})^2} \cdot \frac{\partial}{\partial \boldsymbol{x}_j} \sum_{j\prime} e^{\boldsymbol{x}_{j\prime}} \\ &= \frac{\mathbb{1}_{i=j} e^{\boldsymbol{x}_i}}{\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}}} - \frac{e^{\boldsymbol{x}_i}}{(\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}})^2} \cdot e^{\boldsymbol{x}_j} \\ &= \frac{\mathbb{1}_{i=j} e^{\boldsymbol{x}_i}}{\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}}} - \frac{e^{\boldsymbol{x}_i}}{\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}}} \cdot \frac{e^{\boldsymbol{x}_j}}{\sum_{j\prime} e^{\boldsymbol{x}_{j\prime}}} = \mathbb{1}_{i=j} S(\boldsymbol{x})_i - S(\boldsymbol{x})_i S(\boldsymbol{x})_j \end{aligned}$$

Therefore,

$$\frac{\partial S(\boldsymbol{x})_i}{\partial \boldsymbol{x}_j} = \mathbb{1}_{i=j} S(\boldsymbol{x})_i - S(\boldsymbol{x})_i S(\boldsymbol{x})_j$$

$$\frac{\partial S(\boldsymbol{x})}{\partial \boldsymbol{x}} = \text{Diag}(S(\boldsymbol{x})) - S(\boldsymbol{x})S(\boldsymbol{x})^\top = \text{Diag}(\boldsymbol{p}) - \boldsymbol{p}\boldsymbol{p}^\top \in \mathbb{R}^{n \times n}$$

■.

Question 3 (6)**.** Consider a 2-layer neural network $y : \mathbb{R}^D \to \mathbb{R}^K$ of the form :

$$y(x, \Theta, \sigma)_k = \sum_{j=1}^{M} \omega_{kj}^{(2)} \sigma \left(\sum_{i=1}^{D} \omega_{ji}^{(1)} x_i + \omega_{j0}^{(1)} \right) + \omega_{k0}^{(2)}$$

for $1 \leq k \leq K$, with parameters $\Theta = (\omega^{(1)}, \omega^{(2)})$ and logistic sigmoid activation function σ. Show that there exists an equivalent network of the same form, with parameters $\Theta' = (\tilde{\omega}^{(1)}, \tilde{\omega}^{(2)})$ and tanh activation function, such that $y(x, \Theta', \tanh) = y(x, \Theta, \sigma)$ for all $x \in \mathbb{R}^D$, and express Θ' as a function of Θ.

Answer.

Firstly, the relation between $\tanh(x)$ and $\sigma(x)$ is :

$$\begin{aligned}
\text{thanh}(x) &= \frac{e^x - e^{-x}}{e^x + e^{-x}} \\
&= \frac{e^x - e^{-x} - 2e^{-x}}{e^x + e^{-x}} = 1 - \frac{2e^{-x}}{e^x + e^{-x}} \\
&= 1 - \frac{e^{-x}2}{e^{-x} \cdot (e^{2x} + 1)} = 1 - \frac{2}{e^{2x} + 1} \\
&= 1 - 2.\sigma(-2x). = 1 - 2.(1 - \sigma(2x)) \qquad \text{as } \sigma(-x) = 1 - \sigma(x) \\
&= 1 - 2 + 2.\sigma(2x) = 2.\sigma(2x) - 1
\end{aligned}$$

So, $\text{thanh}(x) = 2.\sigma(2x) - 1 \Leftrightarrow \sigma(x) = \frac{1}{2} \cdot (\text{thanh}(\frac{1}{2}x) + 1)$

Therefore,

$$
\begin{aligned}
y(x,\Theta,\sigma)_k &= \sum_{j=1}^{M} \omega_{kj}^{(2)} \sigma\left(\sum_{i=1}^{D} \omega_{ji}^{(1)} x_i + \omega_{j0}^{(1)}\right) + \omega_{k0}^{(2)} \\
&= \sum_{j=1}^{M} \omega_{kj}^{(2)} \frac{1}{2}(\tanh\left(\sum_{i=1}^{D} \frac{\omega_{ji}^{(1)}}{2} x_i + \frac{\omega_{j0}^{(1)}}{2}\right) + 1) + \omega_{k0}^{(2)} \\
&= \sum_{j=1}^{M} \frac{\omega_{kj}^{(2)}}{2} \tanh\left(\sum_{i=1}^{D} \frac{\omega_{ji}^{(1)}}{2} x_i + \frac{\omega_{j0}^{(1)}}{2}\right) + (\sum_{j=1}^{M} \frac{\omega_{kj}^{(2)}}{2} + \omega_{k0}^{(2)}) \\
&= \sum_{j=1}^{M} \tilde{\omega_{kj}}^{(2)} \tanh\left(\sum_{i=1}^{D} \tilde{\omega_{ji}}^{(1)} x_i + \tilde{\omega_{j0}}^{(1)}\right) + \tilde{\omega_{k0}}^{(2)} = y(x,\Theta',\tanh)_k
\end{aligned}
$$

■.

With,

$$
\Theta' = (\tilde{\omega}^{(1)}, \tilde{\omega}^{(2)})
$$

$$
\tilde{\omega_{ji}}^{(1)} = \frac{\omega_{ji}^{(1)}}{2}, \quad \tilde{\omega_{j0}}^{(1)} = \frac{\omega_{j0}^{(1)}}{2}, \quad \tilde{\omega_{kj}}^{(2)} = \frac{\omega_{kj}^{(2)}}{2}, \quad \tilde{\omega_{k0}}^{(2)} = (\sum_{j=1}^{M} \frac{\omega_{kj}^{(2)}}{2} + \omega_{k0}^{(2)})
$$

Question 4 (5-5-6)**.** Consider a convolutional neural network. Assume the input is a colorful image of size 128×128 in the RGB representation. The first layer convolves 32 8×8 kernels with the input, using a stride of 2 and a zero-padding of 3 (three zeros on each side). The second layer downsamples the output of the first layer with a 2×2 non-overlapping max pooling. The third layer convolves 64 3×3 kernels with a stride of 1 and a zero-padding of size 1 on each border.

4.1 What is the dimensionality of the output of the last layer, i.e. the number of scalars it contains ?

Answer.

Reminder :

$$
\begin{aligned}
H_{\text{out}} &= \left\lfloor \frac{H_{in} + 2 \times \text{padding}[0] - \text{dilation}[0] \times (\text{kernel\_size}[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor \\
W_{\text{out}} &= \left\lfloor \frac{W_{in} + 2 \times \text{padding}[1] - \text{dilation}[1] \times (\text{kernel\_size}[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor
\end{aligned}
$$

Convolution 1 :

$$
\begin{aligned}
H_{conv1} &= \left\lfloor \frac{128 + 2 \times 3 - 1 \times (8-1) - 1}{2} + 1 \right\rfloor = 64 \\
W_{conv1} &= \left\lfloor \frac{128 + 2 \times 3 - 1 \times (8-1) - 1}{2} + 1 \right\rfloor = 64
\end{aligned}
$$

output of first convolution : $32 \times 64 \times 64$

Maxpooling 1 :
As it is non-overlapping, this is equivalent to having a string of 2.

$$H_{pool1} = \left\lfloor \frac{64 + 2 \times 0 - 1 \times (2-1) - 1}{2} + 1 \right\rfloor = 32$$
$$W_{pool1} = \left\lfloor \frac{64 + 2 \times 0 - 1 \times (2-1) - 1}{2} + 1 \right\rfloor = 32$$

output of first maxpooling : $32 \times 32 \times 32$

Convolution 2 :

$$H_{conv2} = \left\lfloor \frac{32 + 2 \times 1 - 1 \times (3-1) - 1}{1} + 1 \right\rfloor = 32$$
$$W_{conv2} = \left\lfloor \frac{32 + 2 \times 1 - 1 \times (3-1) - 1}{1} + 1 \right\rfloor = 32$$

output of second convolution : $64 \times 32 \times 32$

$\boxed{\text{There is } 64 \times 32 \times 32 = 65536 \text{ scalars in the output of the last layer}}$

4.2 Not including the biases, how many parameters are needed for the last layer ?
Answer.

The number of parameters of the last layer is $32 \times 32 \times 32$ (channel, height, width) and the kernel size is $64 \times 3 \times 3 \times 32$ (num_kernels, height, width, channel). The number of parameters for the last layer without biases is $\boxed{3 \times 3 \times 32 \times 64 = 18432}$ parameters.

4.3 Compute the *full*, *valid*, and *same* convolution (with kernel flipping) for the following 1D matrices : $\big[1, 2, 3, 4\big] * \big[1, 0, 2\big]$
Anwser.

rem : $\tilde{*}$ is equal to the convolution with flipped kernel.

valid convolution result : $\big[1, 2, 3, 4\big] \tilde{*} \big[1, 0, 2\big] = \big[1, 2, 3, 4\big] * \big[2, 0, 1\big] = [5, 8]$
same convolution result : $\big[1, 2, 3, 4\big] \tilde{*} \big[1, 0, 2\big] = \big[1, 2, 3, 4\big] * \big[2, 0, 1\big] = [2, 5, 8, 6]$
full convolution result : $\big[1, 2, 3, 4\big] \tilde{*} \big[1, 0, 2\big] = \big[1, 2, 3, 4\big] * \big[2, 0, 1\big] = [1, 2, 5, 8, 6, 8]$

Question 5 (2-5-6-2-5-6)**.** Let us use the notation $*$ and $\tilde{*}$ to denote the valid and full convolution operator **without kernel flipping**, respectively. The operations are defined as

$$\text{valid} : (\boldsymbol{x} * \boldsymbol{w})_n = \sum_{j=1}^{k} x_{n+j-1} w_j \tag{1}$$

$$\text{full} :(\boldsymbol{x} \,\tilde{*}\, \boldsymbol{w})_n = \sum_{j=1}^{k} x_{n+j-k} w_j \ , \tag{2}$$

where k is the size of the kernel $\boldsymbol{w}$. As a convention, the value of a vector indexed "out-of-bound" is zero, e.g. if $\boldsymbol{x} \in \mathbb{R}^d$, then $x_i = 0$ for $i < 1$ and $i > d$. We define the flip operator which reverse the ordering of the components of a vector, i.e. $\text{flip}(\boldsymbol{w})_j = w_{k-j+1}$.

Consider a convolutional network with 1-D input $\boldsymbol{x} \in \mathbb{R}^d$. Its first and second convolutional layers have kernel $\boldsymbol{w}^1 \in \mathbb{R}^{k_1}$ and $\boldsymbol{w}^2 \in \mathbb{R}^{k_2}$, respectively. Assume $k_1 < d$ and $k_2 < d$. The network is specified as follows :

$$\boldsymbol{a}^1 \leftarrow \boldsymbol{x} * \boldsymbol{w}^1 \text{ (valid convolution)} \tag{3}$$
$$\boldsymbol{h}^1 \leftarrow \text{ReLU}(\boldsymbol{a}^1) \tag{4}$$
$$\boldsymbol{a}^2 \leftarrow \boldsymbol{h}^1 * \boldsymbol{w}^2 \text{ (valid convolution)} \tag{5}$$
$$\boldsymbol{h}^2 \leftarrow \text{ReLU}(\boldsymbol{a}^2) \tag{6}$$
$$... \tag{7}$$
$$L \leftarrow ... \tag{8}$$

where L is the loss.

5.1 What is the dimensionality of $\boldsymbol{a}^2$? Denote it by $|\boldsymbol{a}^2|$.

Answer.

$$|\boldsymbol{a}^1| = \left\lfloor \frac{d + 2 \times 0 - 1 \times (k_1 - 1) - 1}{1} + 1 \right\rfloor = d - k_1 + 1$$

$$|\boldsymbol{h}^1| = |\boldsymbol{a}^1|$$

$$|\boldsymbol{a}^2| = \left\lfloor \frac{(d - k_1 + 1) + 2 \times 0 - 1 \times (k_2 - 1) - 1}{1} + 1 \right\rfloor = d - k_1 + 1 - k_2 + 1 = \boxed{d - k_1 - k_2 + 2}$$

5.2 Derive $\frac{\partial a_i^2}{\partial h_n^1}$. Answer for all $i \in \{1, ..., |\boldsymbol{a}^2|\}$, given a particular n.

Answer.

$$\frac{\partial \boldsymbol{a}_i^2}{\partial \boldsymbol{h}_n^1} = \frac{\partial}{\partial \boldsymbol{h}_n^1} \sum_{j=1}^{k_2} \boldsymbol{h}_{i+j-1}^1 \boldsymbol{w}_j^2 = \begin{cases} \boldsymbol{w}_j^2 & \text{if} \quad i + j - 1 = n \\ 0 & \text{otherwise} \end{cases}$$

where $j = n - i + 1$ the derivative is $\boldsymbol{w}_{n-i+1}^2$

5.3 Show that $\nabla_{\boldsymbol{h}^1} L = \nabla_{\boldsymbol{a}^2} L \,\tilde{*}\, \text{flip}(\boldsymbol{w}^2)$ (full convolution). Start with

$$(\nabla_{\boldsymbol{h}^1} L)_n = \sum_{i=1}^{|\boldsymbol{a}^2|} (\nabla_{\boldsymbol{a}^2} L)_i \frac{\partial a_i^2}{\partial h_n^1}$$

For the following, assume the convolutions in equations (3) and (5) are **full instead of valid**.

Anwser.

We can infer from the previous question that $i \in \{n - k_2 + 1, n\}$ as $j \in \{1, \ldots, k_2\}$. Elsewhere, the derivative of $\frac{\partial a_i^2}{\partial h_n^1}$ will be 0.

$$\begin{aligned}
(\nabla_{h^1} L)_n &= \sum_{i=1}^{|\boldsymbol{a}^2|} (\nabla_{\boldsymbol{a}^2} L)_i \, \frac{\partial \boldsymbol{a}_i^2}{\partial \boldsymbol{h}_n^1} \\
&= \sum_{i=n+1-k_2}^{n} (\nabla_{\boldsymbol{a}^2} L)_i \, \boldsymbol{w}_{n-i+1}^2 \\
&= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-k_2} \, \boldsymbol{w}_{k_2-j+1}^2 \\
&= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-k_2} \, \text{fllp}\left(\boldsymbol{w}^2\right)_j \\
&= \boxed{\left(\nabla_{\boldsymbol{a}^2} L \tilde{*} \, \text{flip}\left(\boldsymbol{w}^2\right)\right)_n}
\end{aligned}$$

■.

5.4 What is the dimensionality of $\boldsymbol{a}^2$? Denote it by $|\boldsymbol{a}^2|$.

Answer.

$$|\boldsymbol{a}^1| = \left\lfloor \frac{d + 2 \times (k_1 - 1) - 1 \times (k_1 - 1) - 1}{1} + 1 \right\rfloor = d - 2k_1 - 2 - k_1 + 1 = d + k_1 - 1$$

$$|\boldsymbol{h}^1| = |\boldsymbol{a}^1|$$

$$|\boldsymbol{a}^2| = \left\lfloor \frac{(d + k_1 - 1) + 2 \times (k_2 - 1) - 1 \times (k_2 - 1) - 1}{1} + 1 \right\rfloor = \boxed{d + k_1 + k_2 - 2}$$

5.5 Derive $\frac{\partial a_i^2}{\partial h_n^1}$. Answer for all $i \in \{1, ..., |\boldsymbol{a}^2|\}$, given a particular n.

Answer.

$$\frac{\partial \boldsymbol{a}_i^2}{\partial \boldsymbol{h}_n^1} = \frac{\partial}{\partial \boldsymbol{h}_n^1} \sum_{j=1}^{k_2} \boldsymbol{h}_{i+j-1}^1 \boldsymbol{w}_j^2 = \begin{cases} \boldsymbol{w}_j^2 & \text{if} \quad i+j-k_2 = n \\ 0 & \text{otherwise} \end{cases}$$

where $j = n + k_2 - i$ the derivative is $\boldsymbol{w}_{n+k_2-i}^2$.

5.6 Show that $\nabla_{\boldsymbol{h}^1} L = \nabla_{\boldsymbol{a}^2} L * \text{flip}(\boldsymbol{w}^2)$ (valid convolution). Start with

$$(\nabla_{\boldsymbol{h}^1} L)_n = \sum_{i=1}^{|\boldsymbol{a}^2|} (\nabla_{\boldsymbol{a}^2} L)_i \frac{\partial a_i^2}{\partial h_n^1}$$

Anwser.

We can infer from the previous question that $i \in \{n, n + k_2 - 1\}$ as $j \in \{1, \ldots, k_2\}$. Elsewhere, the derivative of $\frac{\partial a_i^2}{\partial h_n^1}$ will be 0.

Therefore,

$$\begin{aligned}
(\nabla_{h^1} L)_n &= \sum_{i=1}^{|\boldsymbol{a}^2|} (\nabla_{\boldsymbol{a}^2} L)_i \frac{\partial \boldsymbol{a}_i^2}{\partial \boldsymbol{h}_n^1} \\
&= \sum_{i=n}^{n+k_2-1} (\nabla_{\boldsymbol{a}^2} L)_i \boldsymbol{w}_{n+k_2-i}^2 \\
&= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-1} \boldsymbol{w}_{k_2-j+1}^2 \\
&= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-1} \text{flip}\left(\boldsymbol{w}^2\right)_j \\
&= \boxed{\left(\nabla_{\boldsymbol{a}^2} L * \text{flip}\left(\boldsymbol{w}^2\right)\right)_n}
\end{aligned}$$

■.