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

Instructions

- *For all questions, show your work!*
- *Use LaTeX and the template we provide when writing your answers. 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.

1.2 What does the output represent under this activation function ?

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.

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

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).

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

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

Answer 1.

1. For a binary classification, *sigmoid* would be an appropriate choice of the function g.

$$g(x) = \frac{1}{1 + e^{-x}}$$

2. In the standard case, the sigmoid output represents the probability that the input x belongs to the class $y = 1$, i.e. $f(\boldsymbol{x}, \boldsymbol{\theta}) = \mathbb{P}(y = 1|\boldsymbol{x})$. *Note that it can model the probability of the other class, $y = 0$, but this will change the derivations of the subsequent questions.*

3. In the standard case described above, the cross-entropy loss can be written

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

4. First, we can re-write the derivative as : $\frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} = \frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial f(\boldsymbol{x},\boldsymbol{\theta})} \frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})}$.
Moreover, using the derivative of the sigmoid function :

$$\frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial a(\boldsymbol{x}, \boldsymbol{\theta})} = \frac{\partial g(a(\boldsymbol{x}, \boldsymbol{\theta}))}{\partial a(\boldsymbol{x}, \boldsymbol{\theta})} = g(a(\boldsymbol{x}, \boldsymbol{\theta}))(1 - g(a(\boldsymbol{x}, \boldsymbol{\theta}))) = f(\boldsymbol{x}, \boldsymbol{\theta})(1 - f(\boldsymbol{x}, \boldsymbol{\theta}))$$

On the other hand we have : $\frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial f(\boldsymbol{x},\boldsymbol{\theta})} = -\frac{y}{f(\boldsymbol{x},\boldsymbol{\theta})} + \frac{1-y}{1-f(\boldsymbol{x},\boldsymbol{\theta})}$
Putting both terms together, we get :

$$\frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} = -y(1-f(\boldsymbol{x},\boldsymbol{\theta})) + (1-y)f(\boldsymbol{x},\boldsymbol{\theta}) = f(\boldsymbol{x},\boldsymbol{\theta}) - y$$

5. $L_{MSE}(f(\boldsymbol{x},\boldsymbol{\theta}),y) = 0.5 \times (f(\boldsymbol{x},\boldsymbol{\theta}) - y)^2$

6. We can write

$$\frac{\partial L_{MSE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} = \frac{\partial L_{MSE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial f(\boldsymbol{x},\boldsymbol{\theta})}\frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})}$$

We have that

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

Now, reusing the term $\frac{\partial f(\boldsymbol{x},\boldsymbol{\theta})}{\partial a(\boldsymbol{x},\boldsymbol{\theta})}$ from question 1.4, we find that :

$$\frac{\partial L_{MSE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})} = (f(\boldsymbol{x},\boldsymbol{\theta}) - y)f(\boldsymbol{x},\boldsymbol{\theta})(1-f(\boldsymbol{x},\boldsymbol{\theta}))$$

7. The derivative of the MSE is equivalent to the derivative of CE times $f(\boldsymbol{x},\boldsymbol{\theta})(1-f(\boldsymbol{x},\boldsymbol{\theta}))$. This term does not depend on the target, and vanishes when the output $f(\boldsymbol{x},\boldsymbol{\theta})$ is near 0 or 1. In other words, when the predicted probability is around the extreme values (0 and 1), and especially when it does not correspond to the desired target, the MSE gradient will be very small which will hinder and considerably limit any useful correction towards the right target, corrections reflected in the term $(f(\boldsymbol{x},\boldsymbol{\theta}) - y)$. Hence, the CE is a more appropriate choice for binary classification since it does not exhibit the same issue.

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.

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 $\boldsymbol{x}$.

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 $i \in \{1, ..., K-1\}$.

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 2.

1. Let K be the dimensionality of $\boldsymbol{x}$. For $i \in \{1, ..., K\}$,

$$\begin{aligned} S(\boldsymbol{x}+c)_i &= \frac{\exp(x_i+c)}{\sum_{j=1}^K \exp(x_k+c)} \\ &= \frac{\exp(c)\exp(x_i)}{\sum_{j=1}^K \exp(c)\exp(x_k)} \\ &= \frac{\exp(c)\exp(x_i)}{\exp(c)\sum_{j=1}^K \exp(x_k)} \\ &= S(\boldsymbol{x})_i \end{aligned}$$

2. Let $z = x_1 - x_2$.

$$\sigma(z) = \frac{1}{1+\exp(-x_1+x_2)} = \frac{\exp(x_1)}{\exp(x_1)+\exp(x_2)} = S(\boldsymbol{x})_1$$

   Similarly, $1 - \sigma(z) = S(\boldsymbol{x})_2$.

3. For $i \in \{1, ..., K-1\}$, let $y_i = x_{i+1} - x_1$. By the translation-invariance,

$$S(\boldsymbol{x}) = S(\boldsymbol{x} - x_1) = S([0, x_2 - x_1, x_3 - x_1 ..., x_K - x_1]^\top) = S([0, y_1, ..., y_{K-1}]^\top)$$

4. Each entry of the jacobian can be computed as follows :

$$J_{\text{softmax}}(\boldsymbol{x})_{ij} = \frac{\partial S(\boldsymbol{x})_i}{\partial x_j} = \frac{\delta_{ij}\exp(x_i)\sum_k \exp(x_k) - \exp(x_i)\exp(x_j)}{(\sum_k \exp(x_k))^2} = S(\boldsymbol{x})_i(\delta_{ij} - S(\boldsymbol{x})_j)$$

   Thus, it can be written in compact notation as $\mathbf{Diag}(\boldsymbol{p}) - \boldsymbol{p}\boldsymbol{p}^\top$, where $\boldsymbol{p} = S(\boldsymbol{x})$.

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 $1 \le k \le K$, 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 3. First since $\tanh(x) = 2\sigma(2x) - 1$, we have $\sigma(x) = \frac{1}{2}\left(\tanh(\frac{x}{2}) + 1\right)$. Thus,

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

where $\tilde{\omega_{ji}}^{(1)} = \frac{\omega_{ji}^{(1)}}{2}$, $\tilde{\omega_{kj}}^{(2)} = \frac{\omega_{kj}^{(2)}}{2}$ for $j \geq 1$, and $\tilde{\omega_{k0}}^{(2)} = \left(\omega_{k0}^{(2)} + \sum_{j=1}^{M} \frac{\omega_{kj}^{(2)}}{2}\right)$.

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 ?

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

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

Answer 4. 1. The output shape of a convolutional layer is

$$o = \left\lfloor \frac{i + 2p - k}{s} \right\rfloor + 1$$

where i, p, k, s are the input size, padding size, kernel size, and stride size, respectively. Initially, the input is of shape $(3, 128, 128)$. After the first layer, the representation is of shape $(32, 64, 64)$. After the second layer, the representation is of shape $(32, 32, 32)$. After the last layer, the output is of shape $(64, 32, 32)$. Thus the output has $64 \times 32 \times 32 = 65536$ dimensions.

2. $64 \times 32 \times 3 \times 3 = 18432$.

3. Full : $\big[1, 2, 5, 8, 6, 8\big]$; Valid : $\big[5, 8\big]$; Same : $\big[2, 5, 8, 6\big]$.

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 $|\boldsymbol{a}^2|$.

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.

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**.

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

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.

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

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

Answer 5. 1. $|\boldsymbol{a}^2| = d - k_1 - k_2 + 2$.

2.

$$a_i^2 = \sum_{j=1}^{k_2} h_{i+j-1}^1 w_j^2 \tag{9}$$

By setting $n = i + j - 1$, we get that $j = n - i + 1$ and thus

$$\frac{\partial a_i^2}{\partial h_n^1} = \begin{cases} w_{n-i+1}^2, & \text{if } \max(n - k_2 + 1, 1) \leq i \leq \min(n, |\boldsymbol{a}^2|) \\ 0, & \text{otherwise} \end{cases} . \tag{10}$$

3.

$$(\nabla_{\boldsymbol{h}^1} L)_n = \sum_{i=1}^{|\boldsymbol{a}^2|} (\nabla_{\boldsymbol{a}^2} L)_i \frac{\partial a_i^2}{\partial h_n^1} \tag{11}$$

$$= \sum_{i=n-k_2+1}^{n} (\nabla_{\boldsymbol{a}^2} L)_i w_{n-i+1}^2 \quad \text{recall } (\nabla_{\boldsymbol{a}^2} L)_i = 0 \text{ when } i < 1 \text{ or } i > |\boldsymbol{a}^2| \tag{12}$$

Notice the reverse ordering of w^2 compared to $\nabla_{\boldsymbol{a}^2} L$

Replace $n - i + 1$ by $k_2 - j + 1$ to bring in flip operator, hence $i = n + j - k_2$

$$= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-k_2} w_{k_2-j+1}^2 \tag{13}$$

$$= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-k_2} \text{flip}(\boldsymbol{w}^2)_j \tag{14}$$

$$= (\nabla_{\boldsymbol{a}^2} L \;\tilde{*}\; \text{flip}(\boldsymbol{w}^2))_n \tag{15}$$

4. $|\boldsymbol{a}^2| = d + k_1 + k_2 - 2$.

5.

$$a_i^2 = \sum_{j=1}^{k_2} h_{i+j-k_2}^1 w_j^2 \tag{16}$$

By setting $n = i + j - k_2$, we get that $j = n + k_2 - i$ and thus

$$\frac{\partial a_i^2}{\partial h_n^1} = \begin{cases} w_{n+k_2-i}^2, & \text{if } n \le i \le n + k_2 - 1 \\ 0, & \text{otherwise} \end{cases}. \tag{17}$$

6.

$$(\nabla_{\boldsymbol{h}^1} L)_n = \sum_{i=1}^{|\boldsymbol{a}^2|} (\nabla_{\boldsymbol{a}^2} L)_i \frac{\partial a_i^2}{\partial h_n^1} \tag{18}$$

$$= \sum_{i=n}^{n+k_2-1} (\nabla_{\boldsymbol{a}^2} L)_i w_{n+k_2-i}^2 \tag{19}$$

Notice the reverse ordering of w^2 compared to $\nabla_{\boldsymbol{a}^2} L$

Replace $n + k_2 - i$ by $k_2 - j + 1$ to bring in flip operator, hence $i = n + j - 1$

$$= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-1} w_{k_2-j+1}^2 \tag{20}$$

$$= \sum_{j=1}^{k_2} (\nabla_{\boldsymbol{a}^2} L)_{n+j-1} \text{flip}(\boldsymbol{w}^2)_j \tag{21}$$

$$= (\nabla_{\boldsymbol{a}^2} L * \text{flip}(\boldsymbol{w}^2))_n \tag{22}$$