IFT 3395/6390 (6390: GRAD) Theoretical Homework 3

Harmanpreet Singh, Himanshu Arora, Akshay Singh Rana

TOTAL POINTS

61.5 / 70

QUESTION 1

Derivatives and relationships between basic functions 13 pts

1.1 1a 1 / 1

✓ **- 0 pts Correct**

- 1 pts incorrect

1.2 1b 1 / 1

✓ **- 0 pts Correct**

1.3 1c 1 / 1

✓ **- 0 pts Correct**

- 1 pts incorrect

1.4 1d 0 / 1

- 0 pts Correct

- 1 pts Did not express in function of tanh only

✓ **- 1 pts Incorrect**

1.5 1e 0 / 1

- 0 pts Correct

- 1 pts Did not express using indicator functions only

✓ **- 1 pts Incorrect**

When x = 0, we encounter 0/0 and when x<0, your function will return 1 and not -1.

1.6 1f 1 / 1

✓ **- 0 pts Correct**

- 0.5 pts Not the simplest answer

- 1 pts Incorrect

1.7 1g 1 / 1

✓ **- 0 pts Correct**

- 1 pts Incorrect: derivative rect'(0) = 1 and not 0

- 1 pts Did not use the indicator function

- 0.5 pts Not the simplest answer

1.8 1h 1 / 1

✓ **- 0 pts Correct**

- 1 pts Incorrect

1.9 1i 1 / 1

✓ **- 0 pts Correct**

- 1 pts Incorrect

- 0.5 pts Not the simplest answer

1.10 1j 1 / 1

✓ **- 0 pts Correct**

- 0.5 pts One of the two cases is wrong

- 0.5 pts Did not show for both cases

1.11 1k 1 / 1

✓ **- 0 pts Correct**

- 1 pts Incorrect

- 0.5 pts Indexing starts at 0

- 1 pts Didn't use diag function

1.12 1l 1 / 2

✓ **- 0 pts Correct**

- 2 pts No answer

✓ **- 1 pts One of the two activations is wrong or not shown**

- 1.5 pts Did not show operations

- 2 pts Incorrect

- 0.5 pts Did not use element wise multiplication

Need to prove for any loss function

QUESTION 2

Gradient computation for parameter

optimization in a neural net for multiclass classification 37 pts

2.1 2a 2 / 2

✓ **- 0 pts Correct**

- 1 pts one of the 3 formulas is wrong or missing

- 2 pts 2 or 3 of the 3 formulas are wrong

- 0.5 pts h^a as a function of x should be in matrix form !

- 0.5 pts dimension of b^(1) is d_h

2.2 2b 2 / 2

✓ **- 0 pts Correct**

- 2 pts wrong

- 0.5 pts use matrix form !

2.3 2c 4 / 2

✓ **+ 2 pts 2c Correct**

✓ **+ 2 pts 2d Correct**

+ 1 pts 2c partially correct

+ 1 pts 2d partially correct

+ 0 pts both 2c and 2d wrong

2.4 2d 0 / 2

✓ **+ 0 pts This question has been graded along with 2c**

2.5 2e 2 / 2

✓ **- 0 pts Correct**

- 1 pts wrong formulation of optimization (or absent)

- 1 pts wrong expression of Risk (or absent)

- 1 pts wrong expression for n_theta (or absent)

- 0.5 pts lack of clarity

2.6 2f 2 / 2

✓ **- 0 pts Correct**

- 2 pts incorrect

- 1 pts confusing

- 1 pts sign error

- 1 pts missing 1/n

- 2 pts missing learning rate

2.7 2g 3 / 3

✓ **- 0 pts Correct**

- 3 pts incorrect (too many critical mistakes)

- 1 pts notation confusion

- 3 pts incorrect (too confusing / false)

- 1 pts mathematical error

2.8 2h 3 / 3

✓ **- 0 pts Correct**

- 3 pts serious mathematical error when derivating

2.9 2i 2 / 2

✓ **- 0 pts Correct**

- 1 pts Dimension of gradient with respect to W(2) not explicitly given.

- 1 pts Gradients are not expressed in terms of matrix/vector multiplications, but rather as a collection of scalars to form a matrix.

- 1 pts Transpose missing

- 1 pts Dimensions correct, but gradient incorrect

- 0.5 pts Small error

- 1 pts Dimension of gradient wrt to W incorrect. Should be m x d_h.

- 0 pts Blank, no response

2.10 2j 2 / 2

✓ **- 0 pts Correct**

- 2 pts No/incomplete/wrong answer

- 0 pts dL/do^a_k should not be substituted with its expression

- 1 pts Error in indices (e.g. missing transpose)

- 1 pts The sum (given in the question...) is missing.

- 0.5 pts Error in the substitution of dL/do^a_k

- 0 pts Click here to replace this description.

2.11 2k 2 / 2

✓ **- 0 pts Correct**

- 1 pts Incorrect/missing dimensions

- 0.5 pts A dimension is missing/incorrect

- 1 pts Incorrect matrix form/did not use last question

- 2 pts Incorrect

2.12 2l 2 / 2

✓ - 0 pts Correct

- 1 pts Incorrect derivative or did show LR derivative
- 1 pts Incorrect expression for the gradient
- 2 pts Incorrect

2.13 2m 2 / 2

✓ - 0 pts Correct

- 1 pts Incorrect/missing dimensions
- 0.5 pts A dimension is missing
- 1 pts Not a correct matrix form or did not use previous result
- 2 pts Incorrect

2.14 2n 0 / 2

- 0 pts Both equations are correct
- 1 pts One of the equations is incorrect/missing

✓ - 2 pts Incorrect

2.15 2o 2 / 2

✓ - 0 pts Correct

- 2 pts wrong
- 1 pts one of the gradients is wrong

2.16 2p 0 / 2

- 0 pts Correct

✓ - 2 pts wrong

2.17 2q 3 / 3

✓ - 0 pts Correct

- 1.5 pts not entirely right
- 1.5 pts Right, but what about the values of the gradients ?
- 3 pts wrong

QUESTION 3

Practical homework report 20 pts

3.1 Question 1 5 / 5

✓ - 0 pts Correct

- 1.5 pts your loss curves are not right
- 0.5 pts the question specifically asks for 2 figures !

3.2 Question 2 5 / 5

✓ - 0 pts Correct

- 1 pts one of the propositions in 2b is wrong
- 2 pts wrong number of learnable parameters
- 2 pts both propositions in 2b are wrong
- 1 pts 2a answer not clear

3.3 Question 3 5 / 5

✓ - 0 pts Correct

- 2 pts n is n_hidden - 1, not n_hidden
- 2.5 pts wrong number of TOTAL hidden layers
- 1.5 pts one of the figures is wrong
- 2.5 pts both figures are wrong
- 0.5 pts the question specifically asks for 2 figures

3.4 Question 4 3.5 / 5

- 0 pts Correct
- 2 pts your conclusion is not coherent with your plots

✓ - 0.5 pts you should put all curves in one figure !

✓ - 1 pts no way to decide which model is better

- 2.5 pts you did not mention that the reason is the random initialization / usage of one seed only
- 2.5 pts wrong figure
- 1.5 pts no error bars

(a) [1 points] Show that $\sigma(x) = \frac{1}{2}\left(\tanh(\frac{1}{2}x) + 1\right)$

Answer 1.a.

$$\begin{aligned}
\tanh(x) &= \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \\
\tanh(\frac{x}{2}) &= \frac{e^{\frac{x}{2}} - e^{-\frac{x}{2}}}{e^{\frac{x}{2}} + e^{-\frac{x}{2}}} \\
\tanh(\frac{x}{2}) + 1 &= \frac{e^{\frac{x}{2}} - e^{-\frac{x}{2}}}{e^{\frac{x}{2}} + e^{-\frac{x}{2}}} + 1 \\
&= \frac{2e^{\frac{x}{2}}}{e^{\frac{x}{2}} + e^{-\frac{x}{2}}} \\
\frac{1}{2}(\tanh(\frac{x}{2}) + 1) &= \frac{e^{\frac{x}{2}}}{e^{\frac{x}{2}} + e^{-\frac{x}{2}}} \\
&= \frac{1}{1 + \frac{e^{-\frac{x}{2}}}{e^{\frac{x}{2}}}} \\
&= \frac{1}{1 + e^{-x}} = \sigma(x)
\end{aligned}$$

1.1 1a **1 / 1**

✓ **- 0 pts** **Correct**

- 1 pts incorrect

(b) [1 points] Show that $\ln \sigma(x) = -\text{softplus}(-x)$

Answer 1.b.

$$\begin{aligned}
\sigma(x) &= \frac{1}{1+e^{-x}} \\
\ln\sigma(x) &= \ln(\frac{1}{1+e^{-x}}) \\
&= \ln 1 - \ln(1+e^{-x}) \\
&= -\ln(1+e^{-x}) = -\text{softplus}(-x)
\end{aligned}$$

1.2 1b 1 / 1

✓ - 0 pts Correct

(c) [1 points] Write the derivative of the sigmoid function, σ', using the σ function only

Answer 1.c.

$$
\begin{aligned}
\sigma(x) &= \frac{1}{1+e^{-x}} \\
&= (1+e^{-x})^{-1} \\
\text{and also, } 1+e^{-x} &= \frac{1}{\sigma(x)} \\
e^{-x} &= \frac{1}{\sigma(x)} - 1 \\
&= \frac{1-\sigma(x)}{\sigma(x)} \\
\sigma'(x) &= -1 \times (1+e^{-x})^{-2} \times \frac{d(1+e^{-x})}{dx} \\
&= -(1+e^{-x})^{-2} \times (0 + e^{-x}\frac{d(-x)}{dx}) \\
&= e^{-x} \times \frac{1}{(1+e^{-x})^2} \\
&= \frac{1-\sigma(x)}{\sigma(x)} \times \sigma^2(x) \\
&= \sigma(x)(1-\sigma(x))
\end{aligned}
$$

1.3 1c **1 / 1**

✓ **- 0 pts** **Correct**

- 1 pts incorrect

(d) [1 points] Write the derivative of the hyperbolic tangent function, $\tanh'$, using the tanh function only

Answer 1.d.

$$
\begin{aligned}
\tanh(x) &= \frac{e^x - e^{-x}}{e^x + e^{-x}} \\
&= (e^x - e^{-x})(e^x + e^{-x})^{-1} \\
\tanh'(x) &= (e^x - e^{-x})\frac{d(e^x + e^{-x})^{-1}}{dx} + \frac{d(e^x - e^{-x})}{dx}(e^x + e^{-x})^{-1} \\
&= (e^x - e^{-x})(-1 \times (e^x + e^{-x})^{-2} \times (e^x - e^{-x})) + (e^x + e^{-x})(e^x + e^{-x})^{-1} \\
&= \frac{e^x - e^{-x}}{e^x + e^{-x}} + 1 = \tanh(x) + 1
\end{aligned}
$$

1.4 1d **0 / 1**

- **0 pts** Correct
- **1 pts** Did not express in function of tanh only

✓ **- 1 pts** **Incorrect**

(e) [1 points] Write the sign function using only using indicator functions: $\text{sign}(x) = \ldots$

Answer 1.e.

$$\begin{aligned}\text{sign}(x) &= \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x = 0 \\ -1, & \text{if } x < 0 \end{cases} \\ &= \mathbf{1}_{|x|>0}(\frac{x}{|x|})\end{aligned}$$

1.5 1e **0 / 1**

- **0 pts** Correct
- **1 pts** Did not express using indicator functions only

✓ **- 1 pts Incorrect**

When x = 0, we encounter 0/0 and when x<0, your function will return 1 and not -1.

(f) [1 points] Write the derivative of the absolute value function ($x \mapsto \text{abs}(x) = |x|$), abs'.

Note: its derivative at 0 is not defined, but your function abs' can return 0 at 0.

Note 2: You have to use the sign function.

Answer 1.f.

$$\text{abs}(x) = |x| = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x = 0 \\ -x, & \text{if } x < 0 \end{cases}$$

$$\begin{aligned} \text{abs}'(x) &= \begin{cases} \frac{dx}{dx}, & \text{if } x > 0 \\ \frac{d0}{dx}, & \text{if } x = 0 \\ \frac{-dx}{dx}, & \text{if } x < 0 \end{cases} \\ &= \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x = 0 \\ -1, & \text{if } x < 0 \end{cases} \\ &= \text{sign}(x) \end{aligned}$$

1.6 1f 1 / 1

✓ **- 0 pts** **Correct**

- 0.5 pts Not the simplest answer

- 1 pts Incorrect

(g) [1 points] Write the derivative of the rectifier function, rect$'$.
Note: its derivative at 0 is undefined, but your function rect$'$ can return 0 at 0.
Note2: You have to use the indicator function in your answer.
Answer 1.g.

$$\begin{aligned}
\text{rect}(x) &= \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases} \\
\text{rect}'(x) &= \begin{cases} \frac{dx}{dx}, & \text{if } x > 0 \\ 0, & \text{if } x = 0 \\ \frac{d0}{dx}, & \text{if } x < 0 \end{cases} \\
&= \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases} \\
&= \mathbf{1}_{x>0}
\end{aligned}$$

1.7 1g **1 / 1**

✓ **- 0 pts Correct**

- 1 pts Incorrect: derivative rect'(0) = 1 and not 0

- 1 pts Did not use the indicator function

- 0.5 pts Not the simplest answer

(h) [1 points] Let the squared L_2 norm of a vector be: $\|\mathbf{x}\|_2^2 = \sum_i \mathbf{x}_i^2$. Write the the gradient of the square of the L_2 norm function, $\frac{\partial \|\mathbf{x}\|_2^2}{\partial \mathbf{x}}$, in vector form.

Answer 1.h.

$$\begin{aligned}
\frac{\partial \|\mathbf{x}\|_2^2}{\partial \mathbf{x}} &= \frac{\partial \sum_i \mathbf{x}_i^2}{\partial \mathbf{x}} \\
&= 2\begin{bmatrix} x_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + 2\begin{bmatrix} 0 \\ x_2 \\ \vdots \\ 0 \end{bmatrix} + \ldots + 2\begin{bmatrix} 0 \\ 0 \\ \vdots \\ x_n \end{bmatrix} \\
&= 2\mathbf{x}
\end{aligned}$$

1.8 1h **1 / 1**

✓ **- 0 pts** **Correct**

- 1 pts Incorrect

(i) [1 points] Let the norm L_1 of a vector be: $\|\mathbf{x}\|_1 = \sum_i |\mathbf{x}_i|$. Write the gradient of the L_1 norm function, $\frac{\partial \|\mathbf{x}\|_1}{\partial \mathbf{x}}$, in vector form.

Answer 1.i.

$$
\begin{aligned}
\frac{\partial \|\mathbf{x}\|_1}{\partial \mathbf{x}} &= \frac{\partial \sum_i |\mathbf{x}_i|}{\partial \mathbf{x}} \\
&= \begin{bmatrix} \frac{\partial |x_1|}{\partial x_1} \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{\partial |x_2|}{\partial x_2} \\ \vdots \\ 0 \end{bmatrix} + \dots \\
&= \begin{bmatrix} \frac{x_1}{|x_1|} \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{x_2}{|x_2|} \\ \vdots \\ 0 \end{bmatrix} + \dots \\
&= \begin{bmatrix} \frac{x_1}{|x_1|} \\ \frac{x_2}{|x_2|} \\ \vdots \end{bmatrix} = \text{sign}(\mathbf{x})
\end{aligned}
$$

1.9 1i **1 / 1**

✓ **- 0 pts** **Correct**

- 1 pts Incorrect

- 0.5 pts Not the simplest answer

(j) [1 points] Show that the partial derivatives of the softmax function are given by: $\frac{\partial S(\boldsymbol{x})_i}{\partial \boldsymbol{x}_j} = S(\boldsymbol{x})_i \mathbf{1}_{i=j} - S(\boldsymbol{x})_i S(\boldsymbol{x})_j$.

Answer 1.j.

Consider 2 cases, $i = j$ and $i \neq j$ while calculating derivatives.

Case 1: $i = j$,

$$\begin{aligned}
\frac{\partial S(\boldsymbol{x})_i}{\partial x_i} &= \frac{\partial}{\partial x_i} \frac{e^{x_i}}{\sum_j e^{x_j}} \\
&= \frac{1}{\sum_j e^{x_j}} \frac{\partial e^{x_i}}{\partial x_i} + e^{x_i} \frac{\partial \frac{1}{\sum_j e^{x_j}}}{\partial x_i} \\
&= \frac{e^{x_i}}{\sum_j e^{x_j}} - \frac{e^{x_i} e^{x_i}}{(\sum_j e^{x_j})^2} \\
&= S(\boldsymbol{x})_i - S(\boldsymbol{x})_i^2 \\
&= S(\boldsymbol{x})_i - S(\boldsymbol{x})_i S(\boldsymbol{x})_i
\end{aligned}$$

Case 2: $i \neq j$,

$$\begin{aligned}
\frac{\partial S(\boldsymbol{x})_i}{\partial x_j} &= \frac{\partial \frac{e^{x_j}}{\sum_j e^{x_j}}}{\partial x_j} \\
&= \frac{1}{\sum_j e^{x_j}} \frac{\partial e^{x_i}}{\partial x_j} + e^{x_i} \frac{\partial \frac{1}{\sum_j e^{x_j}}}{\partial x_j} \\
&= \frac{0}{\sum_j e^{x_j}} - \frac{e^{x_i} e^{x_j}}{(\sum_j e^{x_j})^2} \\
&= 0 - S(\boldsymbol{x})_i S(\boldsymbol{x})_j
\end{aligned}$$

Hence, combining case 1 and 2, we get $\frac{\partial S(\boldsymbol{x})_i}{\partial \boldsymbol{x}_j} = S(\boldsymbol{x})_i \mathbf{1}_{i=j} - S(\boldsymbol{x})_i S(\boldsymbol{x})_j$

1.10 1j **1 / 1**

✓ **- 0 pts** **Correct**

- 0.5 pts One of the two cases is wrong

- 0.5 pts Did not show for both cases

(k) [1 points] Express the Jacobian matrix of the softmax function $\frac{\partial S(\boldsymbol{x})}{\partial \boldsymbol{x}}$ using matrix-vector notation. Use the *diag* function. Remember that $\frac{\partial S(\boldsymbol{x})}{\partial \boldsymbol{x}}$ is a $n \times n$ matrix, and for all $i, j \in \{1, \ldots, n\}$, the (i, j) entry of the matrix is $\left(\frac{\partial S(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)_{i,j} = \frac{\partial S(\boldsymbol{x})_i}{\partial \boldsymbol{x}_j}$.

Answer 1.k.

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

$$= \text{diag}(S(\boldsymbol{x})) - S(\boldsymbol{x}) S(\boldsymbol{x})^T$$

1.11 1k **1 / 1**

✓ **- 0 pts** **Correct**

- 1 pts Incorrect

- 0.5 pts Indexing starts at 0

- 1 pts Didn't use diag function

(l) [2 points] Let $\boldsymbol{y}$ and $\boldsymbol{x}$ be $n-$dimensional vectors related by $\boldsymbol{y} = f(\boldsymbol{x})$, L be a differentiable loss function. According to the chain rule of calculus, $\nabla_{\boldsymbol{x}} L = (\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}})^\top \nabla_{\boldsymbol{y}} L$, which takes up $O(n^2)$ computational time in general (as it requires a matrix-vector multiplication).

Show that if $f(\boldsymbol{x}) = \sigma(\boldsymbol{x})$ or $f(\boldsymbol{x}) = S(\boldsymbol{x})$, the above matrix-vector multiplication can be simplified to a $O(n)$ operation.

Note that here, we used the sigmoid function for a vector input $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \in \mathbb{R}^n \mapsto \sigma(\boldsymbol{x}) = (\sigma(\boldsymbol{x}_1), \ldots, \sigma(\boldsymbol{x}_n))$.

Answer 1.l.

As proven in Answer 1.c., for $f(\boldsymbol{x}) = \sigma(\boldsymbol{x})$, $(\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}) = \text{diag}(\sigma(x) - \sigma^2(x))$

$$\begin{aligned}
\nabla_{\boldsymbol{x}} L &= (\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}})^\top \nabla_{\boldsymbol{y}} L \\
&= (\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}})^\top \nabla_{\boldsymbol{y}} L \\
&= \text{diag}(\sigma(x) - \sigma^2(x))^\top \nabla_{\boldsymbol{y}} L \\
&= \text{diag}(\sigma(x))^\top \nabla_{\boldsymbol{y}} L - \text{diag}(\sigma^2(x))^\top \nabla_{\boldsymbol{y}} L
\end{aligned}$$

Since $\nabla_{\boldsymbol{y}} L$, $\text{diag}(\sigma(x))^\top$, and $\text{diag}(\sigma^2(x))^\top$ are $n \times 1$ vectors, the above computation takes $O(n)$.

Now for $f(\boldsymbol{x}) = S(\boldsymbol{x})$,

$$\begin{aligned}
\nabla_{\boldsymbol{y}} L &= -\nabla_{\boldsymbol{y}} \boldsymbol{y} \log(S(\boldsymbol{x})) \\
&= -\boldsymbol{y} S(\boldsymbol{x})^{-1}
\end{aligned}$$

$$\begin{aligned}
\nabla_{\boldsymbol{x}} L &= (\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}})^\top \nabla_{\boldsymbol{y}} L \\
&= (\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}})^\top \nabla_{\boldsymbol{y}} L \\
&= (diag(\mathbf{S}(\boldsymbol{x})) - S(\boldsymbol{x}) S(\boldsymbol{x})^T)(-\boldsymbol{y} S(\boldsymbol{x})^{-1}) \\
&= -\boldsymbol{y} + S(\boldsymbol{x})
\end{aligned}$$

Clearly, $\nabla_{\boldsymbol{x}} L$ can be calculated in $O(n)$ operations.

1.12 1l **1 / 2**

✓ **- 0 pts** **Correct**

- 2 pts No answer

✓ **- 1 pts** **One of the two activations is wrong or not shown**

- 1.5 pts Did not show operations

- 2 pts Incorrect

- 0.5 pts Did not use element wise multiplication

Need to prove for any loss function

(a) [2 points] Let $\mathbf{W}^{(1)}$ be a $d_h \times d$ matrix of weights and $\mathbf{b}^{(1)}$ the bias vector be the connections between the input layer and the hidden layer. What is the dimension of $\mathbf{b}^{(1)}$? Give the formula of the pre-activation vector (before the non linearity) of the neurons of the hidden layer $\mathbf{h}^a$ given $\mathbf{x}$ as input, first in a matrix form ($\mathbf{h}^a = \ldots$), and then details on how to compute one element $\mathbf{h}^a_j = \ldots$. Write the output vector of the hidden layer $\mathbf{h}^s$ with respect to $\mathbf{h}^a$.

Answer 2.a.
Since $\mathbf{b}^{(1)}$ contains one bias term for every neuron in the hidden layer containing d_h neurons, its dimension is $d_h \times 1$.

$$\mathbf{h}^a = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}$$

Each row of $\mathbf{W^{(1)}}$ and $\mathbf{b}^{(1)}$ represents the weights and the bias for a neuron in the hidden layer. The output of the j^{th} neuron can then be computed as: $\mathbf{h}^a_j = \mathbf{W}^{(1)}_j\mathbf{x} + \mathbf{b}^{(1)}_j$

$$\mathbf{h}^s = LeakyRELU_\alpha(\mathbf{h^a}) = \max(\mathbf{h^a}, \alpha\mathbf{h^a})$$

2.1 2a **2 / 2**

✓ **- 0 pts** **Correct**

- 1 pts one of the 3 formulas is wrong or missing

- 2 pts 2 or 3 of the 3 formulas are wrong

- 0.5 pts h^a as a function of x should be in matrix form !

- 0.5 pts dimension of b^(1) is d_h

(b) [2 points] Let $\mathbf{W}^{(2)}$ be a weight matrix and $\mathbf{b}^{(2)}$a bias vector be the connections between the hidden layer and the output layer. What are the dimensions of $\mathbf{W}^{(2)}$ and $\mathbf{b}^{(2)}$? Give the formula of the activation function of the neurons of the output layer $\mathbf{o}^a$ with respect to their input $\mathbf{h}^s$ in a matrix form and then write in a detailed form for $\mathbf{o}_k^a$.

Answer 2.b.
Each row of matrix $\mathbf{W}^{(2)}$ represents the weights for every neuron in the output layer and each column represents the weights for every neuron in the hidden layer. Each row of the bias vector $\mathbf{b}^{(2)}$ represents the bias term for every neuron in the output layer. The dimensions of $\mathbf{W}^{(2)}$ and $\mathbf{b}^{(2)}$ will thus be $m \times d_h$ and $m \times 1$ respectively.

$$\mathbf{o}^a = \mathbf{W}^{(2)}\mathbf{h}^s + \mathbf{b}^{(2)}$$
$$\mathbf{o}_k^a = \mathbf{W}_k^{(2)}\mathbf{h}^s + \mathbf{b}_k^{(2)}$$

2.2 2b **2 / 2**

✓ **- 0 pts** **Correct**

- 2 pts wrong

- 0.5 pts use matrix form !

(c) [2 points] The output of the neurons at the output layer is given by:

$$\mathbf{o}^s \quad = \quad \text{softmax}(\mathbf{o}^a)$$

Give the precise equation for $\mathbf{o}_k^s$ as a function of $\mathbf{o}_j^a$. **Show** that the $\mathbf{o}_k^s$ are positive and sum to 1. Why is this important?

Answer 2.c.

$$\mathbf{o}_k^s = \text{softmax}(\mathbf{o}_k^a) = \frac{e^{\mathbf{o}_k^a}}{\sum_{j=1}^m e^{\mathbf{o}_j^a}}$$

Since exponential function is always positive and all the $\mathbf{o}_k^s$ only contain division of exponential functions and their sum, hence the $\mathbf{o}_k^s$ are always positive.

$$\begin{aligned}\sum_{k=1}^m \mathbf{o}_k^s &= \sum_{k=1}^m \text{softmax}(\mathbf{o}_k^a) = \sum_{k=1}^m \frac{e^{\mathbf{o}_k^a}}{\sum_{j=1}^m e^{\mathbf{o}_j^a}} \\ &= \frac{e^{\mathbf{o}_1^a}}{e^{\mathbf{o}_1^a} + e^{\mathbf{o}_2^a} + \ldots + e^{\mathbf{o}_m^a}} + \ldots + \frac{e^{\mathbf{o}_m^a}}{e^{\mathbf{o}_1^a} + e^{\mathbf{o}_2^a} + \ldots + e^{\mathbf{o}_m^a}} \\ &= \frac{e^{\mathbf{o}_1^a} + e^{\mathbf{o}_2^a} + \ldots + e^{\mathbf{o}_m^a}}{e^{\mathbf{o}_1^a} + e^{\mathbf{o}_2^a} + \ldots + e^{\mathbf{o}_m^a}} = 1\end{aligned}$$

If the above property did not hold true, the output of our neural network would not have been a valid probability distribution. Training a neural network using log-loss or cross-entropy loss requires the output to be a valid probability distribution since it essentially tries to minimize the KL divergence between the true distribution and the model's learned conditional distribution.

2.3 2c **4 / 2**

- ✓ **+ 2 pts** **2c Correct**
- ✓ **+ 2 pts** **2d Correct**
- **+ 1 pts** 2c partially correct
- **+ 1 pts** 2d partially correct
- **+ 0 pts** both 2c and 2d wrong

(d) [2 points] The neural net computes, for an input vector $\mathbf{x}$, a vector of probability scores $\mathbf{o}^s(\mathbf{x})$. The probability, computed by a neural net, that an observation $\mathbf{x}$ belong to class y is given by the y^{th} output $\mathbf{o}^s_y(\mathbf{x})$. This suggests a loss function such as:

$$L(\mathbf{x}, y) = \text{-log } \mathbf{o}^s_y(\mathbf{x})$$

Find the equation of L as a function of the vector $\mathbf{o}^a$. It is easily achievable with the correct substitution using the equation of the previous question.

Answer 2.d.

$$\begin{aligned}
L(\mathbf{x}, y) &= \text{-log } \mathbf{o}^s_y(\mathbf{x}) \\
&= \text{-log softmax}(\mathbf{o}^a_y) \\
&= \text{-log } \frac{e^{\mathbf{o}^a_y}}{\sum_{j=1}^{m} e^{\mathbf{o}^a_j}} \\
&= \text{-log } e^{\mathbf{o}^a_y} \text{ -log } \sum_{j=1}^{m} e^{\mathbf{o}^a_j} \\
&= \text{-}\mathbf{o}^a_y \text{ -log } \sum_{j=1}^{m} e^{\mathbf{o}^a_j}
\end{aligned}$$

2.4 2d 0 / 2

✓ + 0 pts This question has been graded along with 2c

(e) [2 points] The training of the neural net will consist of finding parameters that minimize the empirical risk $\hat{R}$ associated with this loss function. What is $\hat{R}$? What is precisely the set θ of parameters of the network? How many scalar parameters n_θ are there? Write down the optimization problem of training the network in order to find the optimal values for these parameters.

Answer 2.e.
The empirical risk $\hat{R}$ is an estimate of the true risk and is defined as the average loss over the dataset,

$$\hat{R} = \frac{1}{n} \sum_{i=1}^{n} L(\mathbf{x}_i, y_i; \theta)$$

θ are the parameters that the neural network learns during training. It includes all the weights and the biases for every neuron in the hidden and the output layer. $\theta = \{\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(2)}\}$

Using the dimensions of the matrices and vectors in θ from answers 2.a. and 2.b., $n_\theta = d_h \times d + d_h + m \times d_h + m$

The optimization problem to find the optimal parameters is given by:

$$\arg\min_{\theta} \hat{R}$$

$$\text{or, } \arg\min_{\theta} \sum_{i=1}^{n} L(\mathbf{x}_i, y_i; \theta)$$

2.5 2e 2 / 2

✓ **- 0 pts** **Correct**

- 1 pts wrong formulation of optimization (or absent)

- 1 pts wrong expression of Risk (or absent)

- 1 pts wrong expression for n_theta (or absent)

- 0.5 pts lack of clarity

(f) [2 points] To find a solution to this optimization problem, we will use gradient descent. What is the (batch) gradient descent equation for this problem?

Answer 2.f.

$$\theta := \theta - \eta \frac{d\hat{R}}{d\theta}$$

where η is the learning rate and $\theta = \{\mathbf{W}^{(1])}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(2)}\}$

Specifically, for each iteration, update rule is as follow:

$$\mathbf{W}^{(1)} := \mathbf{W}^{(1)} - \eta \frac{d\hat{R}}{d\mathbf{W}^{(1)}}$$

$$\mathbf{b}^{(1)} := \mathbf{b}^{(1)} - \eta \frac{d\hat{R}}{d\mathbf{b}^{(1)}}$$

$$\mathbf{W}^{(2)} := \mathbf{W}^{(2)} - \eta \frac{d\hat{R}}{d\mathbf{W}^{(2)}}$$

$$\mathbf{b}^{(2)} := \mathbf{b}^{(2)} - \eta \frac{d\hat{R}}{d\mathbf{b}^{(2)}}$$

2.6 2f **2 / 2**

✓ **- 0 pts** **Correct**

- 2 pts incorrect

- 1 pts confusing

- 1 pts sign error

- 1 pts missing 1/n

- 2 pts missing learning rate

Answer 2.g.

$$\frac{\partial L}{\partial \mathbf{o}^s} = -\frac{\partial \log \mathbf{o}^s_y}{\partial \mathbf{o}^s} = -\mathbf{o}^{-s} \circ \text{onehot}_m(y)$$
$$= \begin{bmatrix} -\frac{1}{\mathbf{o}^s_1} \\ \vdots \\ -\frac{1}{\mathbf{o}^s_y} \\ \vdots \\ -\frac{1}{\mathbf{o}^s_m} \end{bmatrix} \circ \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ -\frac{1}{\mathbf{o}^s_y} \\ \vdots \\ 0 \end{bmatrix}$$

$$\frac{\partial \mathbf{o}^s}{\partial \mathbf{o}^a} = \frac{\partial \text{softmax}(\mathbf{o}^a)}{\partial \mathbf{o}^a}$$

Using answer 1.j. and 1.k.,

$$\frac{\partial \mathbf{o}^s}{\partial \mathbf{o}^a} = \begin{bmatrix} o^s_1 - o^s_1 o^s_1 & \cdots\cdots & -o^s_1 o^s_m \\ \vdots & & \vdots \\ \vdots & & \vdots \\ -o^s_m o^s_1 & \cdots\cdots & o^s_m - o^s_m o^s_m \end{bmatrix}$$

$$\frac{\partial L}{\partial \mathbf{o}^a} = \frac{\partial L}{\partial \mathbf{o}^s} \frac{\partial \mathbf{o}^s}{\partial \mathbf{o}^a}$$
$$= \begin{bmatrix} o^s_1 - o^s_1 o^s_1 & \cdots\cdots & -o^s_1 o^s_m \\ \vdots & & \vdots \\ \vdots & & \vdots \\ -o^s_m o^s_1 & \cdots\cdots & o^s_m - o^s_m o^s_m \end{bmatrix}_{m \times m} \begin{bmatrix} 0 \\ \vdots \\ -\frac{1}{\mathbf{o}^s_y} \\ \vdots \\ 0 \end{bmatrix}_{m \times 1} = \begin{bmatrix} \mathbf{o}^s_1 \\ \vdots \\ \mathbf{o}^s_y - 1 \\ \vdots \\ \mathbf{o}^s_m \end{bmatrix}_{m \times 1}$$
$$= \mathbf{o}^s - \text{onehot}_m(y)$$

2.7 2g 3 / 3

✓ **- 0 pts** **Correct**

- 3 pts incorrect (too many critical mistakes)

- 1 pts notation confusion

- 3 pts incorrect (too confusing / false)

- 1 pts mathematical error

(h) [3 points] Compute the gradients with respect to parameters $\mathbf{W}^{(2)}$ and $\mathbf{b}^{(2)}$ of the output layer. Since L depends on $\mathbf{W}^{(2)}_{kj}$ and $\mathbf{b}^{(2)}_k$ only through $\mathbf{o}^a_k$ the result of the chain rule is:

$$\frac{\partial L}{\partial \mathbf{W}^{(2)}_{kj}} = \frac{\partial L}{\partial \mathbf{o}^a_k} \frac{\partial \mathbf{o}^a_k}{\partial \mathbf{W}^{(2)}_{kj}}$$

and

$$\frac{\partial L}{\partial \mathbf{b}^{(2)}_k} = \frac{\partial L}{\partial \mathbf{o}^a_k} \frac{\partial \mathbf{o}^a_k}{\partial \mathbf{b}^{(2)}_k}$$

Answer 2.h.

$$\begin{aligned}
\mathbf{o}^a_k &= \mathbf{W}^{(2)}_k \mathbf{h}^s + \mathbf{b}^{(2)}_k \\
\frac{\partial \mathbf{o}^a_k}{\partial \mathbf{W}^{(2)}_{kj}} &= \frac{\partial \mathbf{W}^{(2)}_k \mathbf{h}^s}{\partial \mathbf{W}^{(2)}_{kj}} + \frac{\partial \mathbf{b}^{(2)}_k}{\partial \mathbf{W}^{(2)}_j} = \mathbf{h}^s_j \\
\frac{\partial L}{\partial \mathbf{W}^{(2)}_{kj}} &= \frac{\partial L}{\partial \mathbf{o}^a_k} \frac{\partial \mathbf{o}^a_k}{\partial \mathbf{W}^{(2)}_{kj}} \\
&= \frac{\partial L}{\partial \mathbf{o}^a_k} \mathbf{h}^s_j
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \mathbf{o}^a_k}{\partial \mathbf{b}^{(2)}_k} &= \frac{\partial \mathbf{W}^{(2)}_k \mathbf{h}^s}{\partial \mathbf{b}^{(2)}_k} + \frac{\partial \mathbf{b}^{(2)}_k}{\partial \mathbf{b}^{(2)}_k} = 1 \\
\frac{\partial L}{\partial \mathbf{b}^{(2)}_k} &= \frac{\partial L}{\partial \mathbf{o}^a_k} \frac{\partial \mathbf{o}^a_k}{\partial \mathbf{b}^{(2)}_k} \\
&= \frac{\partial L}{\partial \mathbf{o}^a_k}
\end{aligned}$$

2.8 2h **3 / 3**

✓ **- 0 pts** **Correct**

- 3 pts serious mathematical error when derivating

(i) [2 points] Write down the gradient of the last question in matrix form and define the dimensions of all matrix or vectors involved. (For the gradient with respect to $\mathbf{W}^{(2)}$, we want to arrange the partial derivatives in a matrix that has the same shape as $\mathbf{W}^{(2)}$, and that's what we call the gradient)

What are the dimensions?

Take time to understand why the above equalities are the same as the equations of the last question.

Answer 2.i.

$$\frac{\partial L}{\partial \mathbf{W}^{(2)}} = \frac{\partial L}{\partial \mathbf{o}^a} \mathbf{h}^{s^T}$$
$$\frac{\partial L}{\partial \mathbf{b}^{(2)}} = \frac{\partial L}{\partial \mathbf{o}^a}$$

Dimensions of h^s: $d_h \times 1$
Dimensions of $\frac{\partial L}{\partial \mathbf{W}^{(2)}}$: $m \times d_h$
Dimensions of $\frac{\partial L}{\partial \mathbf{b}^{(2)}}$: $m \times 1$

2.9 2i 2 / 2

✓ **- 0 pts Correct**

- 1 pts Dimension of gradient with respect to W(2) not explicitly given.

- 1 pts Gradients are not expressed in terms of matrix/vector multiplications, but rather as a collection of scalars to form a matrix.

- 1 pts Transpose missing

- 1 pts Dimensions correct, but gradient incorrect

- 0.5 pts Small error

- 1 pts Dimension of gradient wrt to W incorrect. Should be m x d_h.

- 0 pts Blank, no response

(j) [2 points] What is the partial derivative of the loss L with respect to the output of the neurons at the hidden layer? Since L depends on $\mathbf{h}_j^s$ only through the activations of the output neurons $\mathbf{o}^a$ the chain rule yields:

$$\frac{\partial L}{\partial \mathbf{h}_j^s} = \sum_{k=1}^{m} \frac{\partial L}{\partial \mathbf{o}_k^a} \frac{\partial \mathbf{o}_k^a}{\partial \mathbf{h}_j^s}$$

Answer 2.j.

$$\begin{aligned}
\mathbf{o}_k^a &= \mathbf{W}_k^{(2)} \mathbf{h}^s + \mathbf{b}_k^{(2)} \\
\frac{\partial \mathbf{o}_k^a}{\partial \mathbf{h}_j^s} &= \frac{\partial \mathbf{W}_k^{(2)} \mathbf{h}^s}{\partial \mathbf{h}_j^s} + \frac{\partial \mathbf{b}_k^{(2)}}{\partial \mathbf{h}_j^s} = \mathbf{W}_{kj}^{(2)} \\
&= \sum_{k=1}^{m} \frac{\partial L}{\partial \mathbf{o}_k^a} \frac{\partial \mathbf{o}_k^a}{\partial \mathbf{h}_j^s} = \sum_{k=1}^{m} \frac{\partial L}{\partial \mathbf{o}_k^a} \mathbf{W}_{kj}^{(2)}
\end{aligned}$$

2.10 2j 2 / 2

✓ **- 0 pts** **Correct**

- 2 pts No/incomplete/wrong answer

- 0 pts dL/do^a_k should not be substituted with its expression

- 1 pts Error in indices (e.g. missing transpose)

- 1 pts The sum (given in the question...) is missing.

- 0.5 pts Error in the substitution of dL/do^a_k

- 0 pts Click here to replace this description.

(k) [2 points] Write down the gradient of the last question in matrix form and define the dimensions of all matrix or vectors involved.

What are the dimensions?

Take time to understand why the above equalities are the same as the equations of the last question.

Answer 2.k.

$$\begin{aligned}\frac{\partial L}{\partial \mathbf{h}^s} &= \begin{bmatrix} \sum_{k=1}^{m}(\mathbf{o}_k^s - \mathbf{1}_{k=y})\mathbf{W}_{k1}^{(2)} \\ \sum_{k=1}^{m}(\mathbf{o}_k^s - \mathbf{1}_{k=y})\mathbf{W}_{k2}^{(2)} \\ \vdots \\ \sum_{k=1}^{m}(\mathbf{o}_k^s - \mathbf{1}_{k=y})\mathbf{W}_{kd_h}^{(2)} \end{bmatrix} \\ &= W^{(2)^T}\mathbf{o}^s - W^{(2)^T}\text{onehot}_m(y) \\ &= W^{(2)^T}(\mathbf{o}^s - \text{onehot}_m(y))\end{aligned}$$

Dimensions of $W^{(2)}$: $m \times d_h$
Dimensions of $\mathbf{o}^{(s)}$: $m \times 1$
Dimensions of $\text{onehot}_m(y)$: $m \times 1$
Therefore, dimensions of the above vector are $d_h \times 1$ as it contains derivative of the loss L for every neuron in the hidden layer.

2.11 2k **2 / 2**

✓ **- 0 pts** **Correct**

- 1 pts Incorrect/missing dimensions

- 0.5 pts A dimension is missing/incorrect

- 1 pts Incorrect matrix form/did not use last question

- 2 pts Incorrect

(l) [2 points] What is the partial derivative of the loss L with respect to the activation of the neurons at the hidden layer? Since L depends on the activation $\mathbf{h}_j^a$ only through $\mathbf{h}_j^s$ of this neuron, the chain rule gives:

$$\frac{\partial L}{\partial \mathbf{h}_j^a} = \frac{\partial L}{\partial \mathbf{h}_j^s} \frac{\partial \mathbf{h}_j^s}{\partial \mathbf{h}_j^a}$$

Note $\mathbf{h}_j^s = \text{LeakyRELU}_\alpha(\mathbf{h}_j^a)$: the leaky rectifier function is applied element-wise. Start by writing the derivative of the rectifier function $\frac{\partial \, \text{LeakyRELU}_\alpha(z)}{\partial z} = \text{LeakyRELU}_\alpha{}'(z) = \ldots$.

Answer 2.l.

$$\begin{aligned}
\text{LeakyRELU}_\alpha(z) &= \max(\alpha z, z) \\
\text{LeakyRELU}_\alpha{}'(z) &= \frac{\partial \, \text{LeakyRELU}_\alpha(z)}{\partial z} = \frac{\partial \max(\alpha z, z)}{\partial z} \\
&= \begin{cases} \frac{dz}{dz}, & \text{if } z > 0 \\ 0, & \text{if } z = 0 \\ \frac{d\alpha z}{dz}, & \text{if } z < 0 \end{cases} \\
&= \begin{cases} 1, & \text{if } z > 0 \\ 0, & \text{if } z = 0 \\ \alpha, & \text{if } z < 0 \end{cases} \\
&= \mathbf{1}_{z>0} + \alpha \mathbf{1}_{z<0} \\
\frac{\partial L}{\partial \mathbf{h}_j^a} &= \frac{\partial L}{\partial \mathbf{h}_j^s} \frac{\partial \mathbf{h}_j^s}{\partial \mathbf{h}_j^a} = \frac{\partial L}{\partial \mathbf{h}_j^s} \text{LeakyRELU}_\alpha{}'(\mathbf{h}_j^a) \\
&= \frac{\partial L}{\partial \mathbf{h}_j^s} (\mathbf{1}_{\mathbf{h}_j^a > 0} + \alpha \mathbf{1}_{\mathbf{h}_j^a < 0})
\end{aligned}$$

2.12 2l **2 / 2**

✓ **- 0 pts** **Correct**

- 1 pts Incorrect derivative or did show LR derivative

- 1 pts Incorrect expression for the gradient

- 2 pts Incorrect

(m) [2 points] Write down the gradient of the last question in matrix form and define the dimensions of all matrix or vectors involved.

Annswer 2.m.

$$\frac{\partial L}{\partial \mathbf{h}^a} = \begin{bmatrix} (\sum_{k=1}^{m}(\mathbf{o}_k^s - \mathbf{1}_{k=y})\mathbf{W}_{k1}^{(2)})(\mathbf{1}_{\mathbf{h}_1^a>0} + \alpha\mathbf{1}_{\mathbf{h}_1^a<0}) \\ (\sum_{k=1}^{m}(\mathbf{o}_k^s - \mathbf{1}_{k=y})\mathbf{W}_{k2}^{(2)})(\mathbf{1}_{\mathbf{h}_2^a>0} + \alpha\mathbf{1}_{\mathbf{h}_2^a<0}) \\ \vdots \\ (\sum_{k=1}^{m}(\mathbf{o}_k^s - \mathbf{1}_{k=y})\mathbf{W}_{kd_h}^{(2)})(\mathbf{1}_{\mathbf{h}_{d_h}^a>0} + \alpha\mathbf{1}_{\mathbf{h}_{d_h}^a<0}) \end{bmatrix}$$
$$= (\mathbf{W}^{(2)^T}(\mathbf{o}^s - \text{onehot}_m(y))) \circ \text{LeakyRELU}_\alpha{}'(\mathbf{h}^a)$$

Dimensions of $W^{(2)}$: $m \times d_h$
Dimensions of $\mathbf{o}^{(s)}$: $m \times 1$
Dimensions of $\text{onehot}_m(y)$: $m \times 1$
Dimensions of $\text{LeakyRELU}_\alpha{}'(\mathbf{h}^a)$: $d_h \times 1$
Dimensions of $\frac{\partial L}{\partial \mathbf{h}^a}$: $d_h \times 1$

2.13 2m **2 / 2**

✓ **- 0 pts Correct**

- 1 pts Incorrect/missing dimensions

- 0.5 pts A dimension is missing

- 1 pts Not a correct matrix form or did not use previous result

- 2 pts Incorrect

(n) [2 points] What is the gradient with respect to the parameters $\mathbf{W}^{(1)}$ and $\mathbf{b}^{(1)}$ of the hidden layer?

Hint: same logic as a previous question.

Answer 2.n.

$$\frac{\partial \mathbf{h}_j^a}{\partial \mathbf{W}_{lj}^{(1)}} = \frac{\partial(\mathbf{W}_j^{(1)}\mathbf{x} + \mathbf{b}_j^{(1)})}{\partial \mathbf{W}_{lj}^{(1)}} = \frac{\partial \mathbf{W}_j^{(1)}\mathbf{x}}{\partial \mathbf{W}_{lj}^{(1)}} + \frac{\partial \mathbf{b}_j^{(1)})}{\partial \mathbf{W}_{lj}^{(1)}} = x_j$$

$$\frac{\partial \mathbf{h}_j^a}{\partial \mathbf{b}_l^{(1)}} = \frac{\partial(\mathbf{W}_j^{(1)}\mathbf{x} + \mathbf{b}_j^{(1)})}{\partial \mathbf{b}_l^{(1)}} = \frac{\partial \mathbf{W}_j^{(1)}\mathbf{x}}{\partial \mathbf{b}_l^{(1)}} + \frac{\partial \mathbf{b}_j^{(1)}}{\partial \mathbf{b}_l^{(1)}} = 1$$

$$\frac{\partial L}{\partial \mathbf{W}_{kj}^{(1)}} = \frac{\partial L}{\partial \mathbf{h}_j^a}\frac{\partial \mathbf{h}_j^a}{\partial \mathbf{W}_{lj}^{(1)}}$$

$$= \frac{\partial L}{\partial \mathbf{h}_j^a} \times x_j$$

$$\frac{\partial L}{\partial \mathbf{b}_l^{(1)}} = \frac{\partial L}{\partial \mathbf{h}_j^a}\frac{\partial \mathbf{h}_j^a}{\partial \mathbf{b}_l^{(1)}}$$

$$= \frac{\partial L}{\partial \mathbf{h}_j^a}$$

2.14 2n **0 / 2**

- **0 pts** Both equations are correct
- **1 pts** One of the equations is incorrect/missing

✓ **- 2 pts Incorrect**

(o) [2 points] Write down the gradient of the last question in matrix form and define the dimensions of all matrix or vectors involved.
Hint: same logic as a previous question.
Answer 2.o.

$$\frac{\partial \mathbf{h}^a}{\partial \mathbf{W}^{(1)}} = \frac{\partial(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})}{\partial \mathbf{W}^{(1)}} = \frac{\partial \mathbf{W}^{(1)}\mathbf{x}}{\partial \mathbf{W}^{(1)}} + \frac{\partial \mathbf{b}^{(1)})}{\partial \mathbf{W}^{(1)}} = \mathbf{x}$$

$$\frac{\partial \mathbf{h}^a}{\partial \mathbf{b}^{(1)}} = \frac{\partial(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})}{\partial \mathbf{b}^{(1)}} = \frac{\partial \mathbf{W}^{(1)}\mathbf{x}}{\partial \mathbf{b}^{(1)}} + \frac{\partial \mathbf{b}^{(1)})}{\partial \mathbf{b}^{(1)}} = 1$$

$$\begin{aligned}\frac{\partial L}{\partial \mathbf{W}^{(1)}} &= \frac{\partial L}{\partial \mathbf{h}^a}\frac{\partial \mathbf{h}^a}{\partial \mathbf{W}^{(1)}} \\ &= \frac{\partial L}{\partial \mathbf{h}^a} \times \mathbf{x}^T \\ \frac{\partial L}{\partial \mathbf{b}^{(1)}} &= \frac{\partial L}{\partial \mathbf{h}^a}\frac{\partial \mathbf{h}^a}{\partial \mathbf{b}^{(1)}} \\ &= \frac{\partial L}{\partial \mathbf{h}^a}\end{aligned}$$

Dimensions of $\frac{\partial L}{\partial \mathbf{b}^{(1)}}$: $d_h \times 1$
Dimensions of $\frac{\partial L}{\partial \mathbf{W}^{(1)}}$: $d_h \times d$
Dimensions of x^T: $1 \times d$

2.15 2o **2 / 2**

✓ **- 0 pts** **Correct**

- 2 pts wrong

- 1 pts one of the gradients is wrong

(p) [2 points] What are the partial derivatives of the loss L with respect to $\mathbf{x}$?

Hint: same logic as a previous question.
Answer 2.p.

$$\begin{aligned}
\mathbf{h}^a &= \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)} \\
\frac{\partial \mathbf{h}^a}{\partial \mathbf{x}} &= \frac{\partial(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})}{\partial \mathbf{x}} \\
&= \frac{\partial(\mathbf{W}^{(1)}\mathbf{x})}{\partial \mathbf{x}} + \frac{\partial \mathbf{b}^{(1)}}{\partial \mathbf{x}} = \mathbf{W}^{(1)}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial L}{\partial \mathbf{x}} &= \frac{\partial L}{\partial \mathbf{h}^a}\frac{\partial \mathbf{h}^a}{\partial \mathbf{x}} \\
&= \mathbf{W}^{(1)^T} \times \frac{\partial L}{\partial \mathbf{h}^a}
\end{aligned}$$

2.16 2p **0 / 2**

- **0 pts** Correct

✓ **- 2 pts wrong**

(q) [3 points] We will now consider a **regularized** empirical risk : $\tilde{R} = \hat{R} + \mathcal{L}(\theta)$, where θ is the vector of all the parameters in the network and $\mathcal{L}(\theta)$ describes a scalar penalty as a function of the parameters θ. The penalty is given importance according to a prior preferences for the values of θ. The L_2 (quadratic) regularization that penalizes the square norm (norm L_2) of the weights (but not the biases) is more standard, is used in ridge regression and is sometimes called "weight-decay". Here we shall consider a double regularization L_2 and L_1 which is sometimes named "elastic net" and we will use different **hyperparameters** (positive scalars $\lambda_{11}, \lambda_{12}, \lambda_{21}, \lambda_{22}$) to control the effect of the regularization at each layer

$$\begin{aligned}
\mathcal{L}(\theta) &= \mathcal{L}(\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(2)}) \\
&= \lambda_{11}\|\mathbf{W}^{(1)}\|_1 + \lambda_{12}\|\mathbf{W}^{(1)}\|_2^2 + \lambda_{21}\|\mathbf{W}^{(2)}\|_1 + \lambda_{22}\|\mathbf{W}^{(2)}\|_2^2 \\
&= \lambda_{11}\left(\sum_{i,j}|\mathbf{W}_{ij}^{(1)}|\right) + \lambda_{12}\left(\sum_{ij}(\mathbf{W}_{ij}^{(1)})^2\right) + \lambda_{21}\left(\sum_{i,j}|\mathbf{W}_{ij}^{(2)}|\right) \\
&\qquad + \lambda_{22}\left(\sum_{ij}(\mathbf{W}_{ij}^{(2)})^2\right)
\end{aligned}$$

We will in fact minimize the regularized risk $\tilde{R}$ instead of $\hat{R}$. How does this change the gradient with respect to the different parameters?

Answer 2.q.

Since,

$$\begin{aligned}
\tilde{R} &= \hat{R} + \mathcal{L}(\theta) \\
\frac{\partial \tilde{R}}{\partial \theta} &= \frac{\partial(\hat{R} + \mathcal{L}(\theta))}{\partial \theta} \\
&= \frac{\partial \hat{R}}{\partial \theta} + \frac{\partial \mathcal{L}(\theta)}{\partial \theta}
\end{aligned}$$

the gradient of the penalty with respect to the parameters θ will be added to the existing gradients. We have already computed $\frac{\partial \hat{R}}{\partial \theta}$ in the above questions, where $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}\}$

We use the matrix cookbook to write the gradients of matrix norms of $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ w.r.t. $\mathcal{L}(\theta)$. This could also be easily

proved by extending the proofs from question 1.h and 1.i, where we calculate the derivative of L1 ans L2 norms of vectors.

$$\frac{\partial \mathcal{L}(\mathbf{W}^{(1)})}{\partial \mathbf{W}^{(1)}} = \lambda_{11} sign(\mathbf{W}^{(1)}) + 2\lambda_{12}\mathbf{W}^{(1)}$$
$$\frac{\partial \mathcal{L}(\mathbf{W}^{(2)})}{\partial \mathbf{W}^{(2)}} = \lambda_{21} sign(\mathbf{W}^{(2)}) + 2\lambda_{22}\mathbf{W}^{(2)}$$
$$\frac{\partial \mathcal{L}(\mathbf{b}^{(1)})}{\partial \mathbf{b}^{(1)}} = 0$$
$$\frac{\partial \mathcal{L}(\mathbf{b}^{(2)})}{\partial \mathbf{b}^{(2)}} = 0$$

Therefore,

$$\frac{\partial \tilde{R}}{\partial \mathbf{W}^{(1)}} = \frac{\partial \hat{R}}{\partial \mathbf{W}^{(1)}} + \frac{\partial \mathcal{L}(\theta)}{\partial \mathbf{W}^{(1)}}$$
$$= ((\mathbf{W}^{(2)^T}(\mathbf{o}^s - \text{onehot}_m(y))) \circ \text{LeakyRELU}_\alpha{}'(\mathbf{h}^a)) \times \mathbf{x}^T + \lambda_{11} sign(\mathbf{W}^{(1)}) + 2\lambda_{12}\mathbf{W}^{(1)}$$
$$\frac{\partial \tilde{R}}{\partial \mathbf{W}^{(2)}} = \frac{\partial \hat{R}}{\partial \mathbf{W}^{(2)}} + \frac{\partial \mathcal{L}(\theta)}{\partial \mathbf{W}^{(2)}}$$
$$= (\mathbf{o}^s - \text{onehot}_m(y)) \times \mathbf{h}^{s^T} + \lambda_{21} sign(\mathbf{W}^{(2)}) + 2\lambda_{22}\mathbf{W}^{(2)}$$
$$\frac{\partial \tilde{R}}{\partial \mathbf{b}^{(1)}} = ((\mathbf{W}^{(2)^T}(\mathbf{o}^s - \text{onehot}_m(y))) \circ \text{LeakyRELU}_\alpha{}'(\mathbf{h}^a))$$
$$\frac{\partial \tilde{R}}{\partial \mathbf{b}^{(2)}} = (\mathbf{o}^s - \text{onehot}_m(y))$$

2.17 2q **3 / 3**

✓ **- 0 pts** **Correct**

- 1.5 pts not entirely right

- 1.5 pts Right, but what about the values of the gradients ?

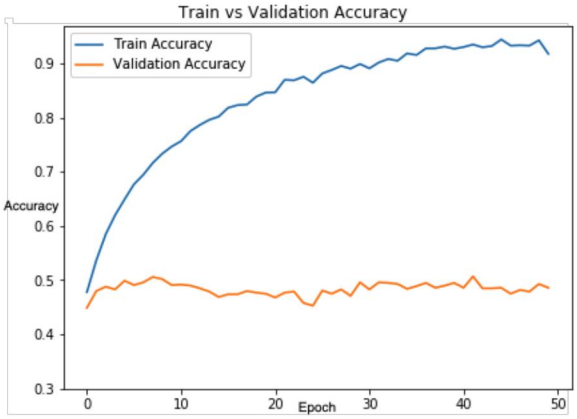
- 3 pts wrong

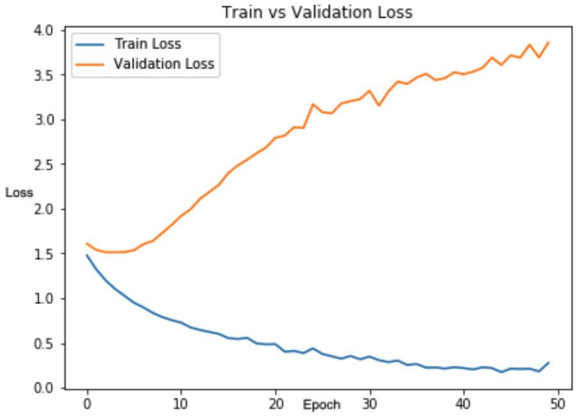
1. Undergraduates 5 bonus pts Graduates 5 pts Train a neural network with 2 hidden layers, of size 512 and 256 respectively on the CIFAR-10 dataset, for 50 epochs. Use a learning rate of 0.003, and a batch size of 100. Use the RELU activation function. For reproducibility purposes, **please set the random seed to 0**.

   Include in your report the two following figures:

   - A figure containing the evolution of both the training and validation accuracies during training
   - A figure containing the evolution of both the training and validation losses during training.

   For verification purposes, you should get a training accuracy larger than 0.8 at the 50th epoch.





3.1 Question 1 **5 / 5**

✓ **- 0 pts Correct**

- 1.5 pts your loss curves are not right

- 0.5 pts the question specifically asks for 2 figures !

2. Undergraduates 5 bonus pts Graduates 5 pts

   (a) Explain in no more than two sentences why the performances on the validation set are not as good as on the training set.
   Answer:
   The model is over-fitting on the training set due to a large number of parameters compared to the number of classes and training data and it does seems to memorize the training data rather than learning the patterns in it.

   (b) How could the performance gap be lowered? Make two propositions in the form of short bullet points.
   Answer:

   - **Regularization:** Apply regularization techniques such as L1 and L2 regularization to force the model to learn the patters in training data rather than memorizing it. We can also add layers like Dropout or Batch Normalization.
   - **Reducing the network's capacity:** Simplifying the model by reducing the number of neurons or layers which results in better generalization.

   (c) How many learnable parameters (scalars) does the previous neural network have ?
   Answer:
   3072*512 + 512 + 512*256 + 256 + 256*10 + 10 = 1707274

3. Undergraduates 5 bonus pts Graduates 5 pts In this question, we are going to compare how the depth and width of the neural network affects the performances of this classification problem. For this purpose, we consider a deep neural network with equal number of neurons amongst the hidden layers starting from the second one. In order to have a fair comparison, we need to ensure that this new neural network has approximately the same number of learnable parameters as the previous one. We consider a neural network with n_hidden hidden layers. The first hidden layer has 512 neurons, and all the remaining ones have 120 neurons each.

3.2 Question 2 **5 / 5**

✓ **- 0 pts Correct**

- 1 pts one of the propositions in 2b is wrong

- 2 pts wrong number of learnable parameters

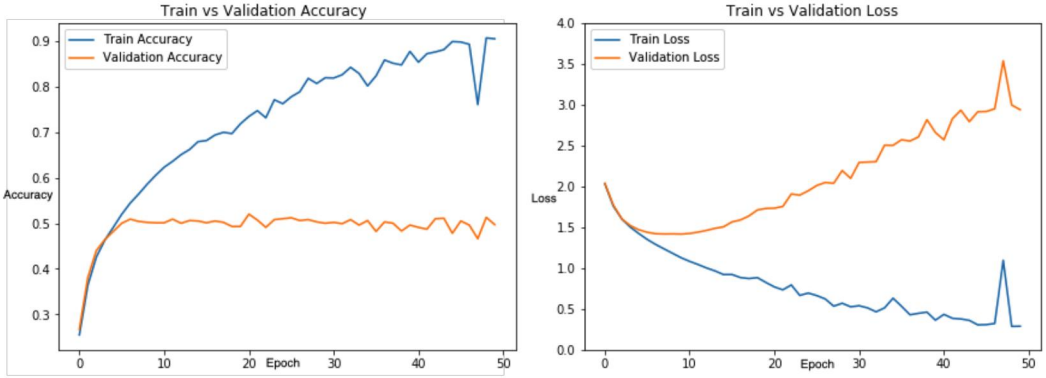
- 2 pts both propositions in 2b are wrong

- 1 pts 2a answer not clear

- Find *n_hidden* such that the new network has a number of parameters that is as close as possible to the number of parameters of the initial neural network. Include this number, along with your reasoning, in the report.
  Answer
  hidden layers = 7
  hidden dimensions=(512,120,120,120,120,120,120)
  W1 = 3072*512 = 1572864
  B1 = 512*1 = 512
  W2 = 512*120 = 61440
  W3 .. W7 = 120*120 = 14400 * 5 = 72000
  B2 .. B7 = 120*1 = 120 * 6 = 720
  W8 = 120*10 = 1200
  B8 = 10*1 = 10

  Total Parameters : 1708746
  Parameters in previous layer : 1707274

- Train the new network with the same parameters as in the first question (same learning rate, batch size, activation function, and same seed). Include in your report two figures containing the training/validation accuracies and losses, similar to the first question.



3.3 Question 3 **5 / 5**

✓ **- 0 pts Correct**

- 2 pts n is n_hidden - 1, not n_hidden

- 2.5 pts wrong number of TOTAL hidden layers

- 1.5 pts one of the figures is wrong

- 2.5 pts both figures are wrong

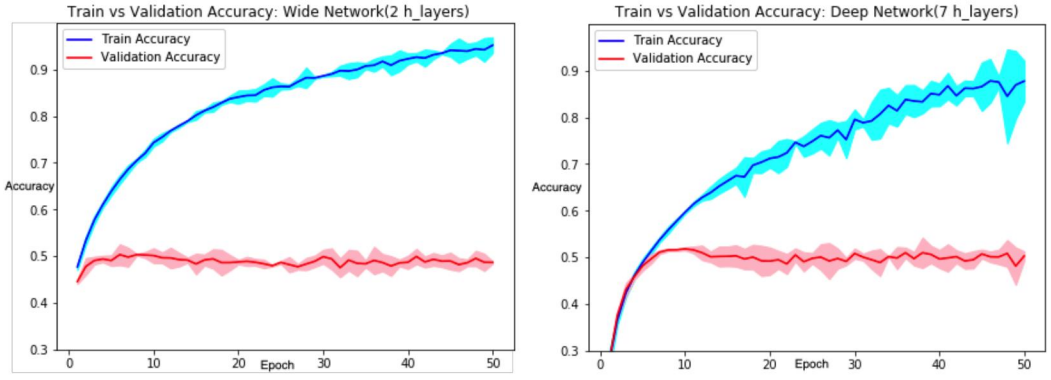
- 0.5 pts the question specifically asks for 2 figures

4. Undergraduates 5 bonus pts Graduates 5 pts

- Why isn't it sufficient to visually compare the figures of the previous question to the figures of the first question to decide which neural network performs best ?
  Answer:
  Since the neural networks are very sensitive to the hyper-parameters, we cant really compare them by running it just once when both the graphs for accuracy and loss looks quite similar.

  Also, the initial parameters for wide and deep network will be quite different and it does make it difficult to compare them directly so we do need to run them multiple times and compare their average performances like in the below figure to see if the difference in meaningful.

- Train both neural networks for 50 epochs, using 3 different seeds of your choice (you need to report them), with the same parameters as in the first question (you thus have to train 6 networks). Include in your report one figure containing the average training and validation accuracies of both neural networks during training, along with error bars corresponding to the standard deviations you obtain in the 3 different runs multiplied by a factor of your choice **that you need to specify in your report**
  Answer
  Both the models were trained with three different seeds each i.e. 42, 123, 7890
  The error plots are plotted with 2*Std deviation.



3.4 Question 4 3.5 / 5

- **0 pts** Correct
- **2 pts** your conclusion is not coherent with your plots
- ✓ **- 0.5 pts** **you should put all curves in one figure !**
- ✓ **- 1 pts** **no way to decide which model is better**
- **2.5 pts** you did not mention that the reason is the random initialization / usage of one seed only
- **2.5 pts** wrong figure
- **1.5 pts** no error bars