# Recitation 5
# Neural Networks

10-301/10-601: Introduction to Machine Learning

2024-10-11

## 1 Matrix Calculus

Consider $\mathbf{x} \in \mathbb{R}^p$, $\mathbf{y} \in \mathbb{R}^r$, $\mathbf{z} \in \mathbb{R}^n$ where $\mathbf{z} = g(\mathbf{y})$, and $\mathbf{y} = f(\mathbf{x})$. We want to derive $d\mathbf{z}/d\mathbf{x}$ (a vector form of the scalar chain rule).

1. If $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ were all scalars, what would $dz/dx$ be?

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}$$

**Shape matching:**

2. Fill in the following shapes:

$$\frac{d\mathbf{y}}{d\mathbf{x}} : \qquad \frac{d\mathbf{z}}{d\mathbf{y}} : \qquad \frac{d\mathbf{z}}{d\mathbf{x}} :$$

$$\frac{d\mathbf{y}}{d\mathbf{x}} : p \times r \quad \frac{d\mathbf{z}}{d\mathbf{y}} : r \times n \quad \frac{d\mathbf{z}}{d\mathbf{x}} : p \times n$$

3. Therefore, the correct derivative is

$$\frac{d\mathbf{z}}{d\mathbf{x}} =$$

$$\frac{d\mathbf{z}}{d\mathbf{x}} = \frac{d\mathbf{y}}{d\mathbf{x}}\frac{d\mathbf{z}}{d\mathbf{y}}$$

**Generalizing a single element:** In order to ensure your derivatives are correct, we recommend you use matrix calculus rules whenever possible. When you're not sure how to apply a rule or if one applies, use the method of generalizing a single element.

4. Example: Suppose $\mathbf{A} \in \mathbb{R}^{n\times n}, \mathbf{x} \in \mathbb{R}^n$, and we want to compute $\frac{d\mathbf{Ax}}{d\mathbf{x}}$. Note that our numerator and denominator are both length-$n$ vectors, so our derivative has shape $\mathbb{R}^{n\times n}$.

5. Is $\left(\frac{d\mathbf{Ax}}{d\mathbf{x}}\right)_{ij}$ equal to $\frac{d\mathbf{Ax}_i}{d\mathbf{x}_j}$ or $\frac{d\mathbf{Ax}_j}{d\mathbf{x}_i}$? Why?

   The latter. To see why, use a non-square matrix $\mathbf{A}$ and observe where the indices must fall in order for the dimensions to work.

6. Compute $\left(\frac{d\mathbf{Ax}}{d\mathbf{x}}\right)_{ij}$:

$$\begin{aligned}
\left(\frac{d\mathbf{Ax}}{d\mathbf{x}}\right)_{ij} &= \frac{d(\mathbf{Ax})_j}{d\mathbf{x}_i} \\
&= \frac{d\mathbf{A}_{j,:}^\top \mathbf{x}}{d\mathbf{x}_i} \\
&= \frac{d}{d\mathbf{x}_i}\sum_{k=1}^{n} \mathbf{A}_{j,k}\mathbf{x}_k \\
&= \mathbf{A}_{j,i}
\end{aligned}$$

   What matrix has $\mathbf{A}_{ji}$ as its $ij$th element? $\mathbf{A}^\top$.

**Applying Matrix Calculus** For example, suppose we are finding the closed-form solution to linear regression: given $\mathbf{X} \in \mathbb{R}^{n\times d}, \mathbf{y} \in \mathbb{R}^n$, we wish to find $\boldsymbol{\theta} \in \mathbb{R}^d$ that minimizes $\|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2$.

7. What is the shape of $\mathbf{X}^\top y$? Is this our solution?

   $\mathbf{X}^\top y \in \mathbb{R}^d$, the same shape as $\boldsymbol{\theta}$. This is not the solution.

8. What is the closed-form solution? Use matrix calculus to derive the solution.

$$\begin{aligned}
\frac{d}{d\theta}\|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2 &= \frac{d}{d\boldsymbol{\theta}}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) \\
&= \frac{d}{d\boldsymbol{\theta}}\left(\mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top(\mathbf{X}\boldsymbol{\theta}) - (\mathbf{X}\boldsymbol{\theta})^\top\mathbf{y} + (\mathbf{X}\boldsymbol{\theta})^\top(\mathbf{X}\boldsymbol{\theta})\right) \\
&= \frac{d}{d\boldsymbol{\theta}}\left(\mathbf{y}^\top\mathbf{y} - 2\mathbf{y}^\top(\mathbf{X}\boldsymbol{\theta}) + (\mathbf{X}\boldsymbol{\theta})^\top(\mathbf{X}\boldsymbol{\theta})\right) \\
&= \frac{d}{d\boldsymbol{\theta}}\left(\mathbf{y}^\top\mathbf{y} - 2(\mathbf{X}^\top\mathbf{y})^\top\boldsymbol{\theta} + (\mathbf{X}\boldsymbol{\theta})^\top(\mathbf{X}\boldsymbol{\theta})\right) \\
&= \frac{d}{d\boldsymbol{\theta}}\left(\mathbf{y}^\top\mathbf{y} - 2(\mathbf{X}^\top\mathbf{y})^\top\boldsymbol{\theta} + (\mathbf{X}\boldsymbol{\theta})^\top(\mathbf{X}\boldsymbol{\theta})\right) \\
&= -2\frac{d}{d\boldsymbol{\theta}}(\mathbf{X}^\top\mathbf{y})^\top\boldsymbol{\theta} + \frac{d}{d\boldsymbol{\theta}}(\mathbf{X}\boldsymbol{\theta})^\top(\mathbf{X}\boldsymbol{\theta}) \\
&= -2(\mathbf{X}^\top\mathbf{y}) + \frac{d}{d\boldsymbol{\theta}}(\mathbf{X}\boldsymbol{\theta})^\top(\mathbf{X}\boldsymbol{\theta}) \\
&= -2(\mathbf{X}^\top\mathbf{y}) + \left(\frac{d}{d\boldsymbol{\theta}}(\mathbf{X}\boldsymbol{\theta})\right)2(\mathbf{X}\boldsymbol{\theta}) \\
&= -2(\mathbf{X}^\top\mathbf{y}) + \mathbf{X}^\top 2(\mathbf{X}\boldsymbol{\theta}) \\
&= 2\left(\mathbf{X}^\top\mathbf{y} - \mathbf{X}^\top\mathbf{X}\boldsymbol{\theta}\right)
\end{aligned}$$

Set equal to 0 and solve:

$$\begin{aligned}
0 &= 2\left(\mathbf{X}^\top\mathbf{y} - \mathbf{X}^\top\mathbf{X}\boldsymbol{\theta}\right) \\
&= \mathbf{X}^\top\mathbf{y} - \mathbf{X}^\top\mathbf{X}\boldsymbol{\theta} \\
\mathbf{X}^\top\mathbf{X}\boldsymbol{\theta} &= \mathbf{X}^\top\mathbf{y} \\
\boldsymbol{\theta} &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}
\end{aligned}$$

## 2 Forward Propagation

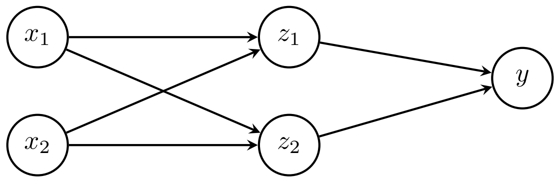


Figure 1: Neural Network For Example Questions

**Forward Propagation** is the process of calculating the value of your loss function, given data, weights and activation functions. Given the input data $\mathbf{x}$, we can transform it by the given weights, $\boldsymbol{\alpha}$, then apply the corresponding activation function to it and finally pass the result to the next layer. Forward propagation does not involve taking derivatives and proceeds from the input layer to the output layer.

**Network Overview** Consider the neural network with one hidden layer shown in Figure 2. The input layer consists of 2 features $\mathbf{x} = [x_1, x_2]^T$, the hidden layer has 2 nodes with output $\mathbf{z} = [z_1, z_2]^T$, and the output layer is a scalar $\hat{y}$. We also add an intercept to the input, $x_0 = 1$ and the output of the hidden layer $z_0 = 1$, both of which are fixed to 1.

$\boldsymbol{\alpha}$ is the matrix of weights from the inputs to the hidden layer and $\boldsymbol{\beta}$ is the matrix of weights from the hidden layer to the output layer. $\alpha_{j,i}$ represents the weight going *to* the node $z_j$ in the hidden layer *from* the node $x_i$ in the input layer (e.g. $\alpha_{1,2}$ is the weight from $x_2$ to $z_1$), and $\boldsymbol{\beta}$ is defined similarly. We will use a **ReLU** activation function for the hidden layer and no activation for the output layer.

**Network Details** Equivalently, we define each of the following.

The input:

$$\mathbf{x} = [x_0, x_1, x_2]^T \tag{1}$$

Linear combination at the first (hidden) layer:

$$a_j = \sum_{i=0}^{2} \alpha_{j,i} \cdot x_i, \ \forall j \in \{1, \dots, 2\} \tag{2}$$

Activation at the first (hidden) layer:

$$z_j = \text{ReLU}(a_j) = \max(0, a_j), \forall j \in \{1, \dots, 2\} \tag{3}$$

$$\mathbf{z} = [z_0, z_1, z_2]^T \tag{4}$$

Linear combination at the second (output) layer:

$$\hat{y} = \sum_{j=0}^{2} \beta_j \cdot z_j, \tag{5}$$

Here we fold in the intercept term $\alpha_{j,0}$ by thinking of $x_0 = 1$, and fold in $\beta_0$ by thinking of $z_0 = 1$.

**Loss** We will use Squared error loss, $\ell(\hat{y}, y)$:

$$\ell(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2 \tag{6}$$

1. Why and how do we include an intercept term in the input and in the hidden-layer?

   Similar to how an intercept term in linear regression allows it to better fit data, the intercept term helps the neural network better fit its data as well.

   We simply fold in the intercept term into our input vector as the 0th term and make the value equal 1.

2. Why do we need to use nonlinear activation functions in our neural net?

   A neural network with only linear activation functions would be no different than a linear regression. (Try forward propagating with only linear functions on the given example)

We initialize the network weights as:

$$\boldsymbol{\alpha} = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} 0 & 1 & 2 \end{bmatrix}$$

For the following questions, we use $y = 3$.

1. **Scalar Form:**

   - Given $x_1 = 1$, $x_2 = 2$, What are the values of $a$?

$$a_2 = \sum_{i=0}^{2} \alpha_{2,i} x_i =$$

$$a_2 = \sum_{i=0}^{2} \alpha_{2,i} x_i = 2$$

   - Given $z_1 = 0, z_2 = 1$ calculate $\hat{y}, l$

$$\hat{y} = \sum_{i=0}^{2} \beta_i \cdot z_i =$$

$$\hat{y} = \sum_{i=0}^{2} \beta_i \cdot z_i = 2$$

$$l = \frac{1}{2}(2-3)^2 = \frac{1}{2}$$

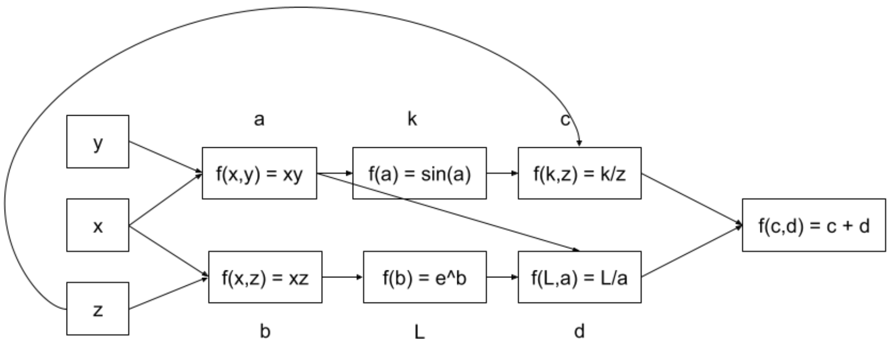
2. **Vector Form:** Find the vector form of forward computation, given **x** is a column vector.

$$\begin{aligned}
\mathbf{a} &= \boldsymbol{\alpha}\hat{\mathbf{x}} \\
&= \begin{bmatrix} \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} \\
&= \begin{bmatrix} \alpha_{1,0}x_0 + \alpha_{1,1}x_1 + \alpha_{1,2}x_2 \\ \alpha_{2,0}x_0 + \alpha_{2,1}x_1 + \alpha_{2,2}x_2 \end{bmatrix} \\
&= \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \qquad (7) \\
\mathbf{z} &= \mathrm{ReLU}(\mathbf{a}) \\
\hat{y} &= \boldsymbol{\beta}\hat{\mathbf{z}} \\
&= \begin{bmatrix} \beta_0 & \beta_1 & \beta_2 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \end{bmatrix} \\
&= \beta_0 z_0 + \beta_1 z_1 + \beta_2 z_2
\end{aligned}$$

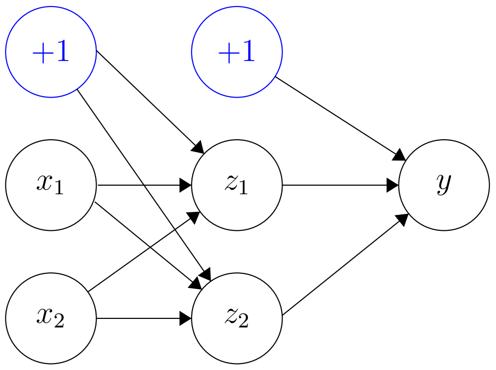
## 3 Computation Diagrams

1. For the following function $f$, create the computation graph using the conventions defined in lecture.

$$f(x, y, z) = \frac{\sin(xy)}{z} + \frac{e^{xz}}{xy}$$



2. For the following neural network, draw the corresponding computation graph. Assume that all hidden units use the ReLU function as the activation function and that the loss is mean squared error. Provide the shape of all parameters defined in the computation graph. Assume the weights for the first layer and second layers are respectively the matrices $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.



$$\begin{tikzcd}
& \boxed{\vec{\mathbf{x}}_{1:2}}\ \vec{\mathbf{x}}' \arrow[d, "\text{augment with bias}"] \\
\boxed{\boldsymbol{\alpha}} \arrow[dr] & \boxed{[1, \vec{\mathbf{x}}']}\ \vec{\mathbf{x}} \arrow[d] \\
& \boxed{\boldsymbol{\alpha}\vec{\mathbf{x}} \iff a_j = \sum_{i=0}^{2} \alpha_{j,i} x_i,\ \forall j \in \{1,2\}}\ \vec{\mathbf{a}} \arrow[d] \\
& \boxed{\text{ReLU}(\vec{\mathbf{a}})}\ \vec{\mathbf{z}}' \arrow[d, "\text{augment with bias}"] \\
\boxed{\boldsymbol{\beta}} \arrow[dr] & \boxed{[1, \vec{\mathbf{z}}']}\ \vec{\mathbf{z}} \arrow[d] \\
& \boxed{\boldsymbol{\beta}\vec{\mathbf{z}} \iff \sum_{j=0}^{2} \beta_j z_j}\ \hat{y} \arrow[d] \\
\boxed{y} \arrow[r] & \boxed{\frac{1}{2}(\hat{y} - y)^2}\ \ell
\end{tikzcd}$$

$\boldsymbol{\alpha} \in \mathbb{R}^{2\times 3} \quad \boldsymbol{\beta} \in \mathbb{R}^{1\times 3}$

## 4 Backward Propagation

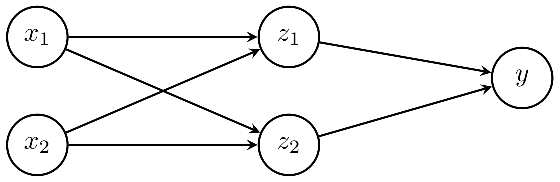


Figure 2: Neural Network For Example Questions

Given a Neural Network and a corresponding loss function $J(\theta)$, backpropagation gives us the gradient of the loss function with respect to the weights of the neural network. The method is called *backward* propagation because we calculate the gradients of the final layer of weights first, then proceed backward to the first layer. In a simple neural network with one hidden layer, the partial derivatives that we need for learning are $\frac{\partial \ell}{\partial \alpha_{ji}}$ and $\frac{\partial \ell}{\partial \beta_{kj}}$, and we need to apply chain rule recursively to obtain these. Note that in implementation, it is easier to use matrix/vector forms to conduct computations.

1. Many gradients are calculated in back propagation. Which of these gradients are used to update the weights? Do not include intermediate value(s) used to calculate these gradient(s).
   The gradients with respect to $\alpha$ and $\beta$ are used in updating. The rest are intermediate values used to calculate these two gradients

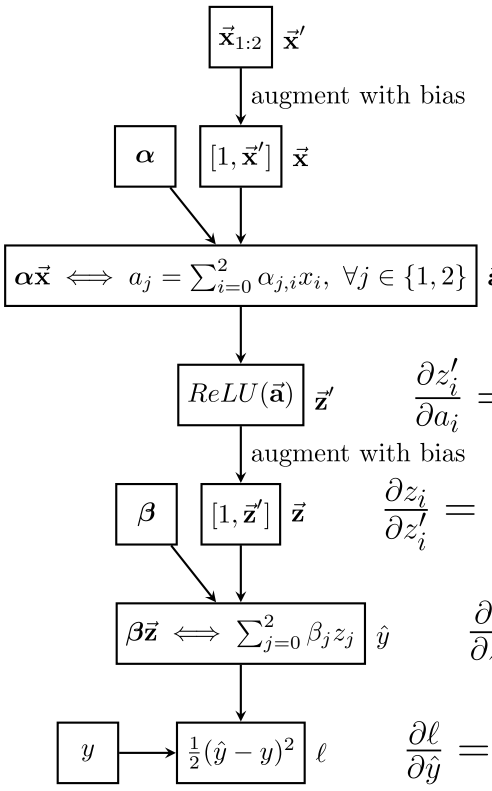
2. **Scalar Form:** Given

   - $x_1 = 1$, $x_2 = 2$
   - $a_1 = 3$, $a_2 = 2$
   - $z_1 = 3$, $z_2 = 2$
   - $\boldsymbol{\alpha} = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \end{bmatrix}$
   - $\boldsymbol{\beta} = \begin{bmatrix} 0 & 1 & 2 \end{bmatrix}$
   - $y = 3$,

   what are the values of $\frac{\partial \ell}{\partial \beta_1}$, $\frac{\partial \ell}{\partial \alpha_{1,1}}$?

   **Hint:** Derive expressions for $\frac{\partial \ell}{\partial \beta_i}$ and $\frac{\partial \ell}{\partial \alpha_{j,i}}$ first, then substitute in values.

   For convenience, the computation graph for the neural network is displayed below:



$$\frac{\partial z_i'}{\partial a_i} =$$

$$\frac{\partial z_i}{\partial z_i'} =$$

$$\frac{\partial \hat{y}}{\partial z_i} =$$

$$\frac{\partial \ell}{\partial \hat{y}} =$$

Figure 3: Computation graph for the neural network in Figure 2

**Hint:** $\frac{\partial ReLU(x)}{\partial x} = 1$ if $x > 0$, $0$ if $x <= 0$

$$\frac{\partial \ell}{\partial \beta_i} = \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \beta_i}$$

$$\frac{\partial \ell}{\partial \beta_1} =$$

As a reminder, we were given that $\ell = \frac{1}{2}(\hat{y} - y)^2$ and $\hat{y} = \sum_{j=0}^{3} \beta_j z_j$

So we can calculate:

$$\begin{aligned}\frac{\partial \ell}{\partial \beta_1} &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \beta_1} \\ &= \frac{\partial}{\partial \hat{y}} \left[ \frac{1}{2} (\hat{y} - y)^2 \right] \frac{\partial}{\partial \beta_1} \left[ \sum_{j=0}^{3} \beta_j z_j \right] \\ &= (\hat{y} - y)\, z_1 = (7 - 3) * 3 = 12\end{aligned}$$

The backprop algorithm caches the value of $\frac{\partial \ell}{\partial \hat{y}}$ for computing further downstream values. This is in contrast to simple symbolic differentiation, which would calculate all the partial derivatives each time.

$$\frac{\partial \ell}{\partial \alpha_{j,i}} = \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_j} \frac{\partial z_j}{\partial a_j} \frac{\partial a_j}{\partial \alpha_{j,i}} \qquad\qquad \frac{\partial \ell}{\partial \alpha_{1,1}} =$$

To find $\frac{\partial \ell}{\partial \alpha_{1,1}}$, we would have to step backwards through the network and store the relevant partial derivatives.

$$\begin{aligned}\frac{\partial \ell}{\partial \hat{y}} &= 4 \text{ (calculated above)} \\ \frac{\partial \ell}{\partial z_1} &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_1} = 4 * \beta_1 = 4 * 1 = 4 \\ \frac{\partial \ell}{\partial a_1} &= \frac{\partial \ell}{\partial z_1} \frac{\partial z_1}{\partial a_1} = 4 * 1 = 4 \\ \frac{\partial \ell}{\partial \alpha_{1,1}} &= \frac{\partial \ell}{\partial a_1} \frac{\partial a_1}{\partial \alpha_{1,1}} = 4 * x_1 = 4 * 1 = 4\end{aligned}$$

3. **Vector Form:** What are the values of $\frac{\partial \ell}{\partial \boldsymbol{\beta}}$, $\frac{\partial \ell}{\partial \boldsymbol{\alpha}}$?

$$\frac{\partial \ell}{\partial \boldsymbol{\beta}} = \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \boldsymbol{\beta}} = (\hat{y} - y)\mathbf{z}^T = \begin{bmatrix} 4 & 12 & 8 \end{bmatrix}$$

Denote $\hat{\boldsymbol{\beta}}$ as $\boldsymbol{\beta}$ without the first entry.

$$\frac{\partial \hat{y}}{\partial z_i} = \boldsymbol{\beta}_i \quad \forall i \in \{1, 2\}$$

$$\frac{\partial \hat{y}}{\partial \mathbf{z}} = \hat{\boldsymbol{\beta}}^T$$

$$\frac{\partial z_i}{\partial a_i} = \begin{cases} 1 \text{ if } a_i > 0 \\ 0 \text{ if } a_i <= 0 \end{cases}$$

$$\begin{aligned} \frac{\partial \ell}{\partial \mathbf{a}} &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{a}} \\ &= (\hat{y} - y) \cdot \hat{\boldsymbol{\beta}}^T \cdot \left(\frac{\partial ReLU(\mathbf{a})}{\partial \mathbf{a}}\right) \\ &= (\hat{y} - y) \cdot \hat{\boldsymbol{\beta}}^T \odot \left(\frac{\partial ReLU(\mathbf{a})}{\partial \mathbf{a}}\right) \qquad (\odot \text{ is element-wise multiplication}) \\ &= 4 \begin{bmatrix} 1 \\ 2 \end{bmatrix} \odot \begin{bmatrix} \frac{\partial ReLU(a_1)}{\partial a_1} \\ \frac{\partial ReLU(a_2)}{\partial a_2} \end{bmatrix} \\ &= 4 \begin{bmatrix} 1 \\ 2 \end{bmatrix} \odot \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 8 \end{bmatrix} \end{aligned}$$

$$\begin{aligned} \frac{\partial \ell}{\partial \alpha_{ji}} &= \frac{\partial \ell}{\partial a_j} \frac{\partial a_j}{\partial \alpha_{ji}} \\ &= \frac{\partial \ell}{\partial a_j} x_i \end{aligned}$$

$$\begin{aligned} \frac{\partial \ell}{\partial \boldsymbol{\alpha}} &= \begin{bmatrix} \frac{\partial \ell}{\partial a_1} x_0 & \frac{\partial \ell}{\partial a_1} x_1 & \frac{\partial \ell}{\partial a_1} x_2 \\ \frac{\partial \ell}{\partial a_2} x_0 & \frac{\partial \ell}{\partial a_2} x_1 & \frac{\partial \ell}{\partial a_2} x_2 \end{bmatrix} \\ &= \frac{\partial \ell}{\partial \mathbf{a}} \mathbf{x}^T \\ &= \begin{bmatrix} 4 & 4 & 8 \\ 8 & 8 & 16 \end{bmatrix} \end{aligned}$$