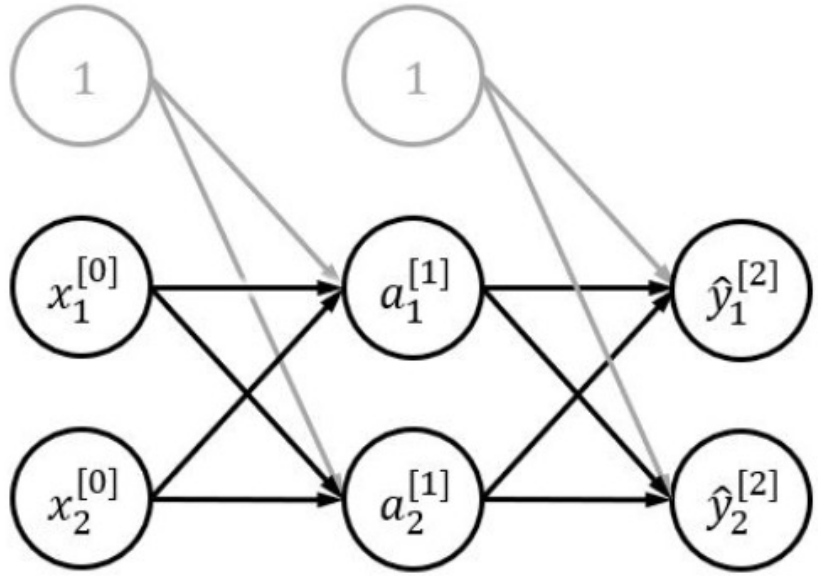
IT5005 Artificial Intelligence

Sirigina Rajendra Prasad
AY2025/2026: Semester 1

Deep Neural Networks

1. **Dot Product-Activate, Forward Propagation**

In this question, we are going to use a neural network with **a 2-D input, one hidden layer with two neurons, and two output neurons**. Additionally, the hidden neurons and the input will **include a bias**. We use **ReLU function** as the non-linear activation function. FYI: $\text{ReLU}(x) = \max(0, x)$.

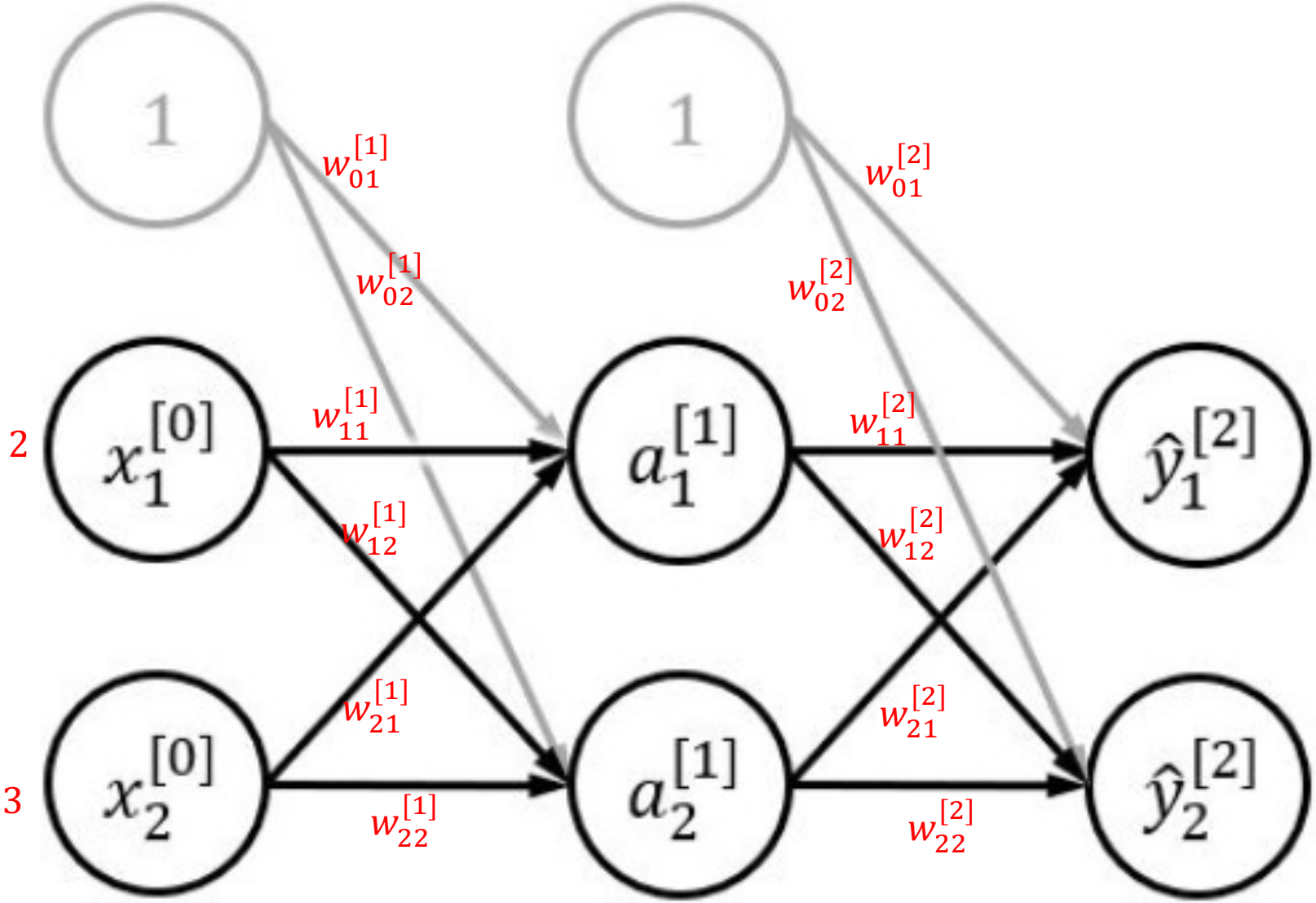


Disclaimer: You may use `NumPy` to compute the following!

Suppose there is a data input $\mathbf{x} = (2, 3)^\top$ and the actual output label is $\mathbf{y} = (0.1, 0.9)^\top$. The weights for the network are

$$\boldsymbol{W}^{[1]} = \begin{bmatrix} 0.1 & 0.1 \\ -0.1 & 0.2 \\ 0.3 & -0.4 \end{bmatrix}, \boldsymbol{W}^{[2]} = \begin{bmatrix} 0.1 & 0.1 \\ 0.5 & -0.6 \\ 0.7 & -0.8 \end{bmatrix},$$

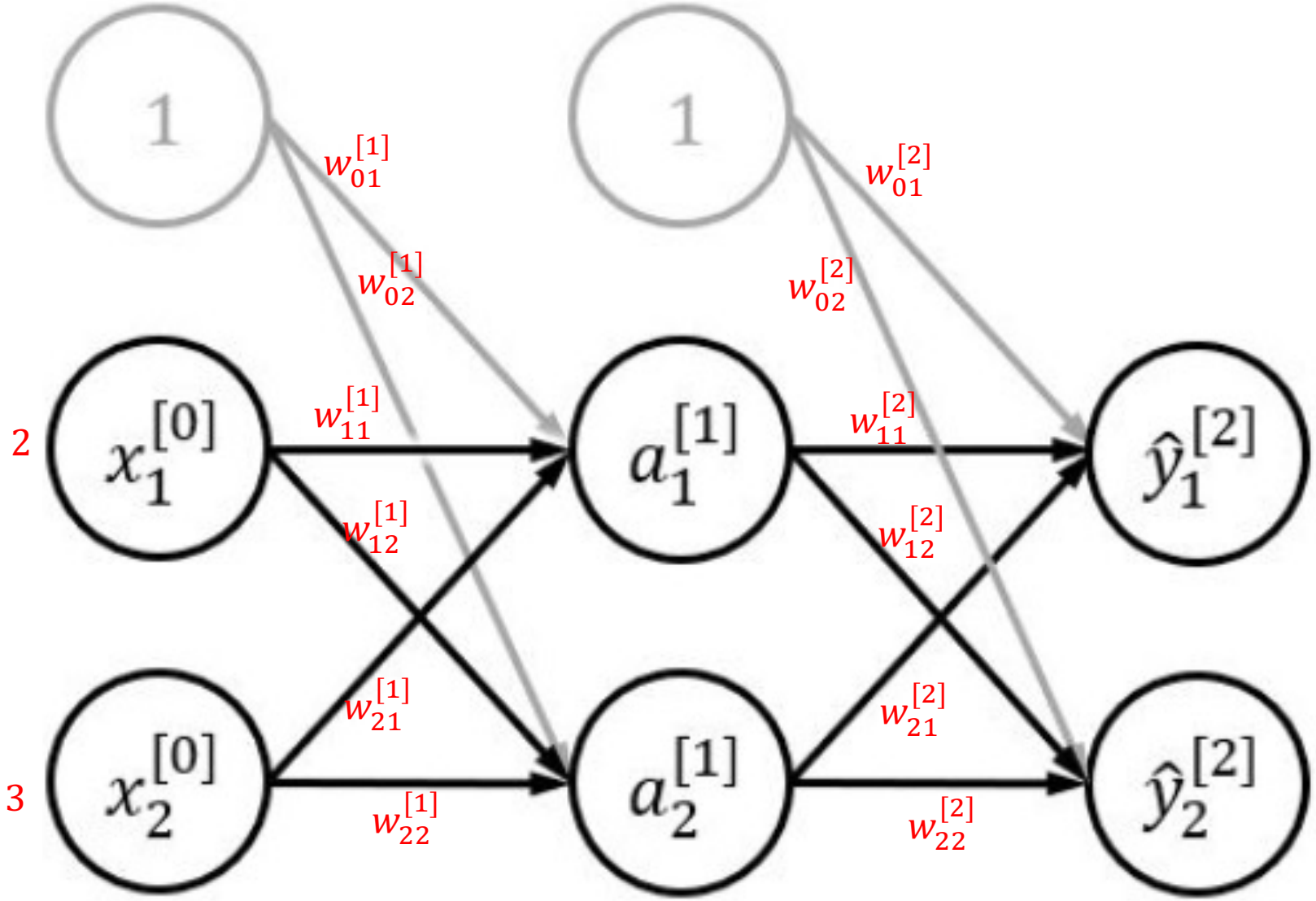
Of course, we also include biases of value $b = 1$ for both hidden and output layers. Calculate the following values after the forward propagation:
$\mathbf{a}^{[1]}$, $\hat{\mathbf{y}}^{[2]}$ and $L(\hat{\mathbf{y}}^{[2]}, \mathbf{y})$. Here, we use MSE (mean squared error) loss function.



$$\boldsymbol{W}^{[1]} = \begin{bmatrix} w_{01}^{[1]} & w_{02}^{[1]} \\ w_{11}^{[1]} & w_{12}^{[1]} \\ w_{21}^{[1]} & w_{22}^{[1]} \end{bmatrix} = \begin{bmatrix} 0.1 & 0.1 \\ -0.1 & 0.2 \\ 0.3 & -0.4 \end{bmatrix}$$

$$\boldsymbol{W}^{[2]} = \begin{bmatrix} w_{01}^{[2]} & w_{02}^{[2]} \\ w_{11}^{[2]} & w_{12}^{[2]} \\ w_{21}^{[2]} & w_{22}^{[2]} \end{bmatrix} = \begin{bmatrix} 0.1 & 0.1 \\ 0.5 & -0.6 \\ 0.7 & -0.8 \end{bmatrix}$$

$$\boldsymbol{X} = \begin{bmatrix} a_0^{[0]} \\ a_1^{[0]} \\ a_2^{[0]} \end{bmatrix} = \begin{bmatrix} x_0^{[0]} \\ x_1^{[0]} \\ x_2^{[0]} \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$



$$\mathbf{a}^{[1]} = ReLU((\mathbf{W}^{[1]})^\top \mathbf{X})$$

$$\boldsymbol{a}^{[1]} = \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \end{bmatrix} = \begin{bmatrix} 0.8 \\ 0 \end{bmatrix}$$

$$\hat{\mathbf{y}}^{[2]} = ReLU((\mathbf{W}^{[2]})^\top \mathbf{a}^{[1]})$$

$$\hat{\boldsymbol{y}}^{[2]} = \begin{bmatrix} \hat{y}_1^{[2]} \\ \hat{y}_2^{[2]} \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$$

$$\begin{aligned} L(\hat{\mathbf{y}}^{[2]}, \mathbf{y}) &= \frac{1}{2} \times ((\hat{y}_1^{[2]} - y_1)^2 + (\hat{y}_2^{[2]} - y_2)^2) \\ &= \frac{1}{2} \times ((0.5 - 0.1)^2 + (0 - 0.9)^2) \\ &= \frac{1}{2} \times (0.16 + 0.81) \\ &= 0.485 \end{aligned}$$

2. **Let's Activate!**

We can define a neural network as follows:

$$\hat{y} = g(\mathbf{W}^{[\mathbf{L}]^\mathbf{T}} \dots g(\mathbf{W}^{[\mathbf{2}]^\mathbf{T}} \cdot g(\mathbf{W}^{[\mathbf{1}]^\mathbf{T}} x)))$$

where $\mathbf{W}^{[\mathbf{l}] \in \{\mathbf{1}, \cdots, \mathbf{L}\}}$ is a weight matrix. You're given the following weight matrices:

$$\mathbf{W}^{[\mathbf{3}]} = \begin{bmatrix} 1.2 & -2.2 \\ 1.2 & 1.3 \end{bmatrix}, \mathbf{W}^{[\mathbf{2}]} = \begin{bmatrix} 2.1 & -0.5 \\ 0.7 & 1.9 \end{bmatrix}, \mathbf{W}^{[\mathbf{1}]} = \begin{bmatrix} 1.4 & 0.6 \\ 0.8 & 0.6 \end{bmatrix}$$

Furthermore, you are given $g(z) = \text{SiLU}(z) = \frac{z}{1+e^{-z}}$ between all layers *except the last layer.*

Disclaimer: Feel free to use `NumPy` to help with the following question!

Is it possible to replace the whole neural network with just one matrix in both cases **with** and **without** non-linear activations $g(z)$? For both cases, either shows that it is possible by providing such a matrix or prove that there exists no such matrix. What does this signify about the importance of the non-linear activation?

Without Activation Function

Solution: Without $\mathrm{SiLU}(z)$, we can show that we can replace the neural network with the following matrix $\mathbf{M^T}$:

$$M^T = \begin{bmatrix} 1.2 & -2.2 \\ 1.2 & 1.3 \end{bmatrix}^T \begin{bmatrix} 2.1 & -0.5 \\ 0.7 & 1.9 \end{bmatrix}^T \begin{bmatrix} 1.4 & 0.6 \\ 0.8 & 0.6 \end{bmatrix}^T$$
$$= \begin{bmatrix} 4.56 & 3.408 \\ -6.82 & -3.658 \end{bmatrix}$$

Without activation function, the entire network collapses to a simple linear model, i.e., the network is as good as network with single layer

$$\hat{y} = \mathbf{W^{[L]T}} \dots \mathbf{W^{[2]T}} \mathbf{W^{[1]T}} x$$
$$= \mathbf{A}x, \quad \text{where } \mathbf{A} = \mathbf{W^{[L]T}} \dots \mathbf{W^{[2]T}} \mathbf{W^{[1]T}} \text{ by matrix multiplication}$$

With SiLU Activation

$$\hat{y}_1 = \begin{bmatrix} 1.2 & -2.2 \\ 1.2 & 1.3 \end{bmatrix}^T g\left(\begin{bmatrix} 2.1 & -0.5 \\ 0.7 & 1.9 \end{bmatrix}^T g\left(\begin{bmatrix} 1.4 & 0.6 \\ 0.8 & 0.6 \end{bmatrix}^T \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \right)$$

$$= \begin{bmatrix} 3.0571 \\ -5.2727 \end{bmatrix}$$

$$\hat{y}_2 = \begin{bmatrix} 1.2 & -2.2 \\ 1.2 & 1.3 \end{bmatrix}^T g\left(\begin{bmatrix} 2.1 & -0.5 \\ 0.7 & 1.9 \end{bmatrix}^T g\left(\begin{bmatrix} 1.4 & 0.6 \\ 0.8 & 0.6 \end{bmatrix}^T \begin{bmatrix} 2 \\ 0 \end{bmatrix} \right) \right)$$

$$= \begin{bmatrix} 7.7257 \\ -13.2458 \end{bmatrix}$$

Since $x_2 = 2x_1$, $\mathbf{M^T}x_2 = 2\mathbf{M^T}x_1 \implies \hat{y}_2 = 2\hat{y}_1$ by linearity of $\mathbf{M^T}$. But by the computation we did above, $\hat{y}_2 \neq 2\hat{y}_1$, thus there exist no such $\mathbf{M^T}$.

3. **Working with Dimensions**

You're building a self-driving car program that takes in grayscale images of size 32×32 where 32 is the image height and width. There are 4 classes your simplified program has to classify: $\{\text{car}, \text{person}, \text{traffic light}, \text{stop sign}\}$. You start off experimenting with a Multi-layer Perceptron composed of three linear layers of the form $y = W^T x$, where $x \in \mathcal{R}^d$ is the input vector, W is the weight matrix, and y is the network output.

What are the dimensions of the input vector, the weight matrix, and the output vector of the three linear layers, given the following details? Assume the batch size is 1.

layer	Input dim	Weight Matrix dim	Output dim
Linear layer 1	_____ $\times 1$	_____ $\times$ _____	512×1
Linear layer 2	512×1	_____ $\times 128$	_____ $\times 1$
Linear layer 3	128×1	_____ $\times 4$	_____ $\times 1$

4. Consider a two-layer neural network with single perceptron in each layer and assume that bias term $b = 0$. The weights for the first and second layer are initialized as 2 and 3, respectively, i.e., $w^{[1]} = 2$ and $w^{[2]} = 3$. Furthermore, an exponential linear unit (ELU) is used as the activation function at each perceptron. The ELU is defined as:

$$g(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha(\exp(x) - 1), & \text{else} \end{cases} \tag{1}$$

The derivative of ELU is defined as:

$$g'(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ \alpha(\exp(x)), & \text{else} \end{cases} \tag{2}$$

Assume that the parameter $\alpha = 1$. The input to the neural network is $x = -1$ and the expected output is $y = -1.4255$.

(a) Calculate $a^{[1]}$ and $a^{[2]}$.

(b) Calculate the mean square error, i.e., $(y - \hat{y})^2$, where $\hat{y}$ is the output of neural network.

(c) Calculate $\frac{\partial L}{\partial w^{[2]}}$ and $\frac{\partial L}{\partial w^{[1]}}$.

(a) Calculate $a^{[1]}$ and $a^{[2]}$

The activation $a^{[1]}$ for the first layer is given by:

$$a^{[1]} = g(w^{[1]} \cdot x) = g(2 \cdot (-1)) = g(-2) = \exp(-2) - 1 = -0.865$$

The activation $a^{[2]}$ for the second layer is:

$$a^{[2]} = g(w^{[2]} \cdot a^{[1]}) = g(3 \cdot (\exp(-2) - 1))$$

$3 \cdot (\exp(-2) - 1) < 0$. Hence,

$$a^{[2]} = \exp(3 \cdot (\exp(-2) - 1)) - 1 = -0.92527$$

(b) Mean Square Error

$$
\begin{aligned}
MSE &= (y - \hat{y})^2 \\
&= (-0.9252 - (-1.4255)))^2 \\
&= 0.25
\end{aligned}
$$

(c) Calculate $\frac{\partial L}{\partial w^{[2]}}$ and $\frac{\partial L}{\partial w^{[1]}}$

The derivative of the loss with respect to $w^{[2]}$ is:

$$\frac{\partial L}{\partial w^{[2]}} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial f^{[2]}} \cdot \frac{\partial f^{[2]}}{\partial w^{[2]}}$$

where

$$\begin{aligned} \frac{\partial L}{\partial \hat{y}} &= -2(y - \hat{y}) \\ \frac{\partial \hat{y}}{\partial f^{[2]}} &= \exp(w^{[2]} a^{[1]}) \\ \frac{\partial f^{[2]}}{\partial w^{[2]}} &= a^{[1]} \end{aligned}$$

Hence,

$$\begin{aligned} \frac{\partial L}{\partial w^{[2]}} &= 2 * (0.5002) * \exp(3 * -0.865) * (-0.865) \\ &= -0.06459 \end{aligned}$$

The derivative of the loss with respect to $w^{[1]}$ is:

$$\frac{\partial L}{\partial w^{[1]}} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial f^{[2]}} \cdot \frac{\partial f^{[2]}}{\partial g^{[1]}} \frac{\partial g^{[1]}}{\partial f^{[1]}} \frac{\partial f^{[1]}}{\partial w^{[1]}}$$

where:

$$\begin{aligned} \frac{\partial g^{[1]}}{\partial f^{[1]}} &= \exp(w^{[1]}x) \\ \frac{\partial f^{[1]}}{\partial w^{[1]}} &= x \end{aligned}$$

Hence,

$$\begin{aligned} \frac{\partial L}{\partial w^{[1]}} &= 2 * (0.5002) * \exp(3 * -0.865) * 3 \exp(2 * (-1)) * (-1) \\ &= -0.0303 \end{aligned}$$