Vectorized Backpropagation

Deep Learning

Brad Quinton, Scott Chin

Learning Objectives

- Extend our understanding of backpropagation to vectorized operations

Overview

- Review of vectorized operations and notations on neural networks
- Introduce vectorized backprop for vector-in vector-out operations
    - Recap: Example with tanh activation function
    - Example with relu activation function
- Introduce vectorized backprop for matrix-in matrix-out operations
    - Example with matrix multiplication
- backprop through cost function
- Backprop through broadcasted operations

Review of Vectorized Operations on Neural Networks

Recall why we want vectorized code

- The faster you can compute all your gradients, the faster you can do one iteration of parameter updates during gradient decent
- The faster you can do one iteration of gradient decent, the faster it is to complete the training of your model
- Finding a good model for your application is an iterative process (Train, Assess, Adjust, repeat). The faster you can train, the faster you can go iterate.

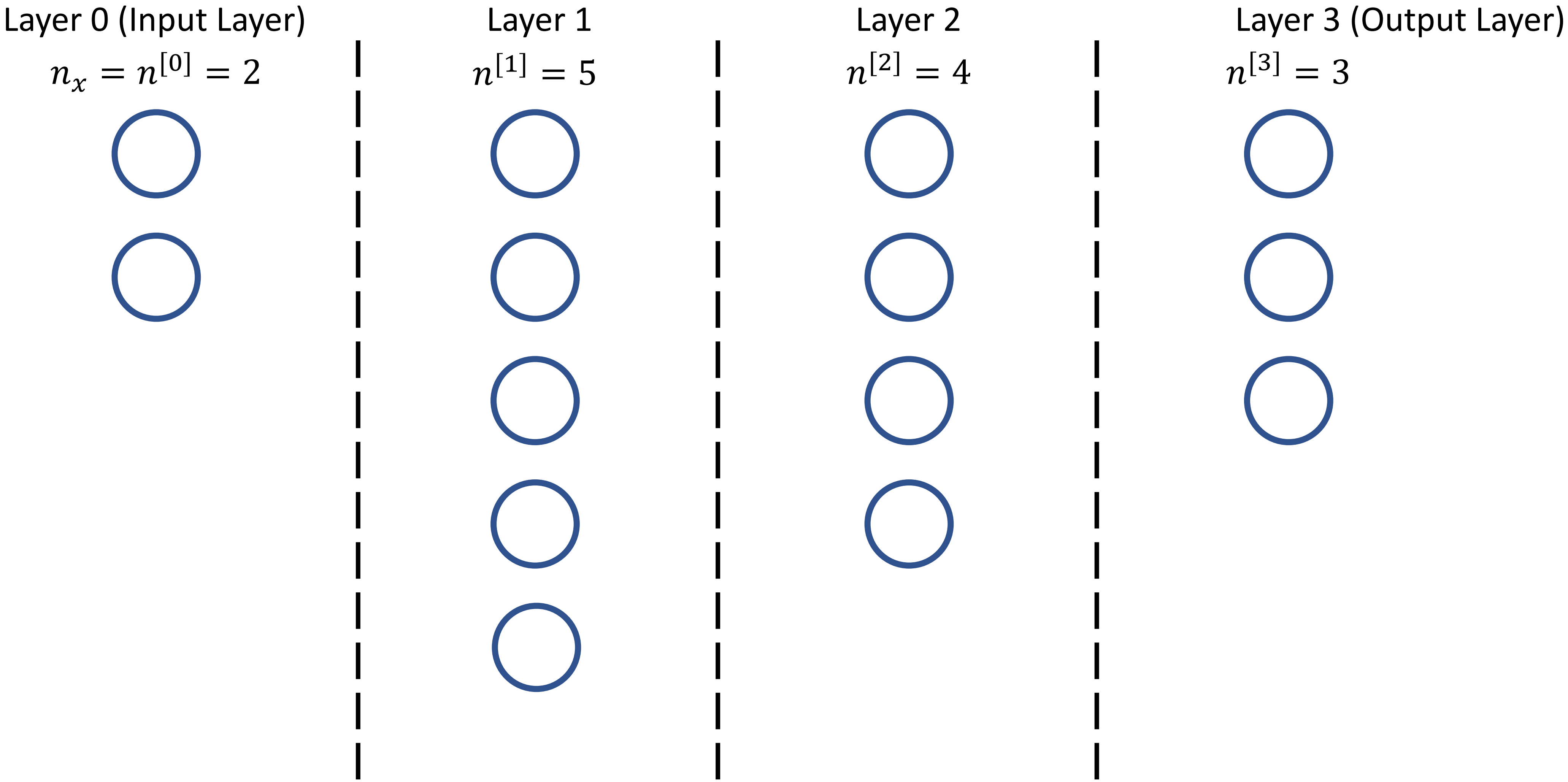
Recall why we want vectorized code

- Modern CPUs, GPUs, and special purpose AI silicon all gain computational efficiency by grouping key operations together so they can be executed in parallel
- This is called vectorization

iPhone 12 A14 Processor Announced October 12, 2020

70%
faster ML accelerators
First 5 nm chip
A14
in a smartphone
16-core
Neural Engine
6-core CPU
50% faster
GPU
Faster than any other smartphone chip
New image signal processor
A14
11 trillion
operations per second on the Neural Engine
80% faster
Neural Engine
4-core GPU
11.8 billion
transistors
Improved memory compression
50% faster
CPU
Faster than any other smartphone chip
Machine learning controller
ML
Best machine learning platform in a smartphone
Secure Enclave

Image source: https://www.zdnet.com/article/apple-iphone-12-5g-upgrades-could-be-justified-by-a14-bionic/



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

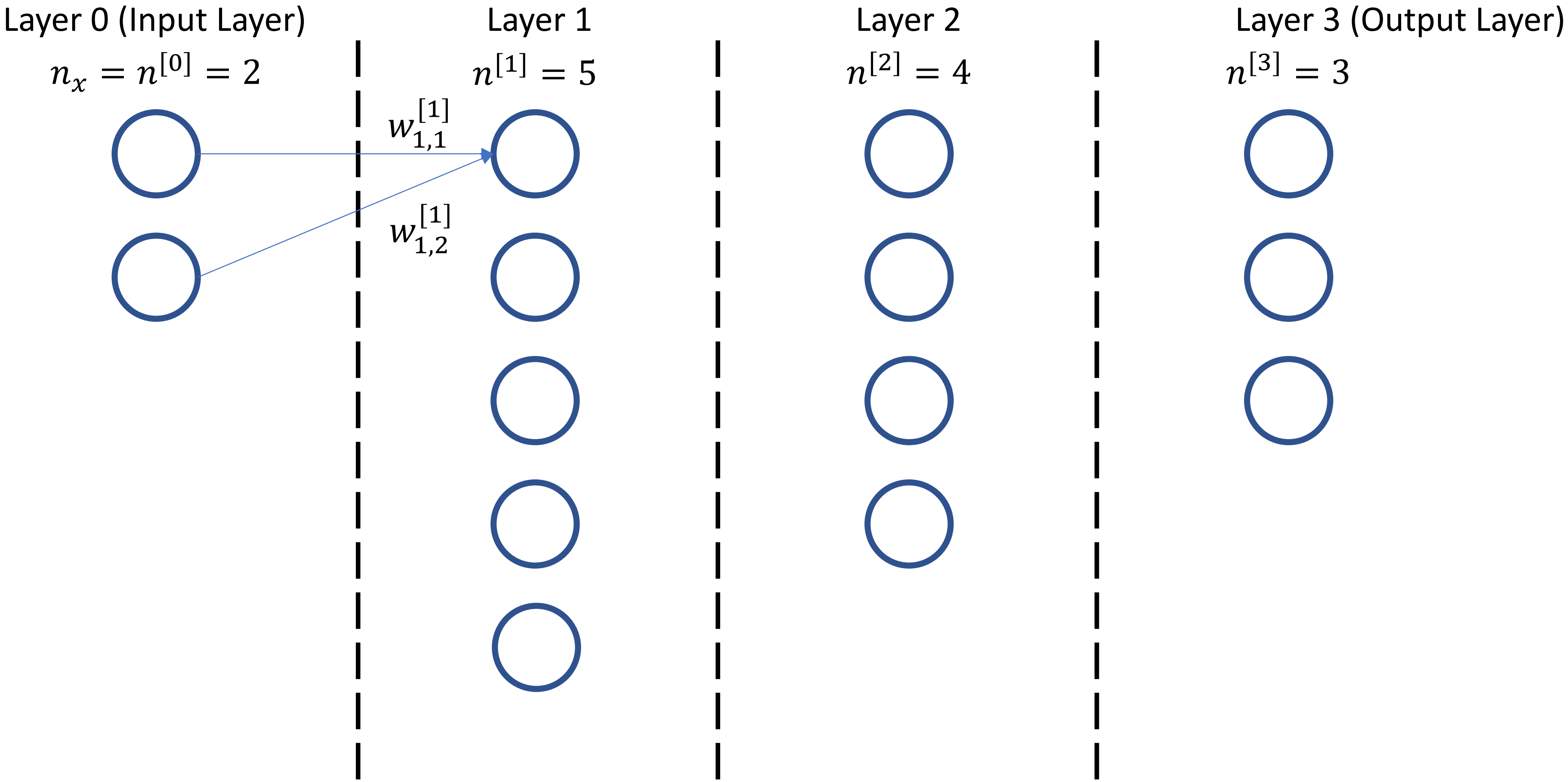
Layer 2

$n^{[2]} = 4$

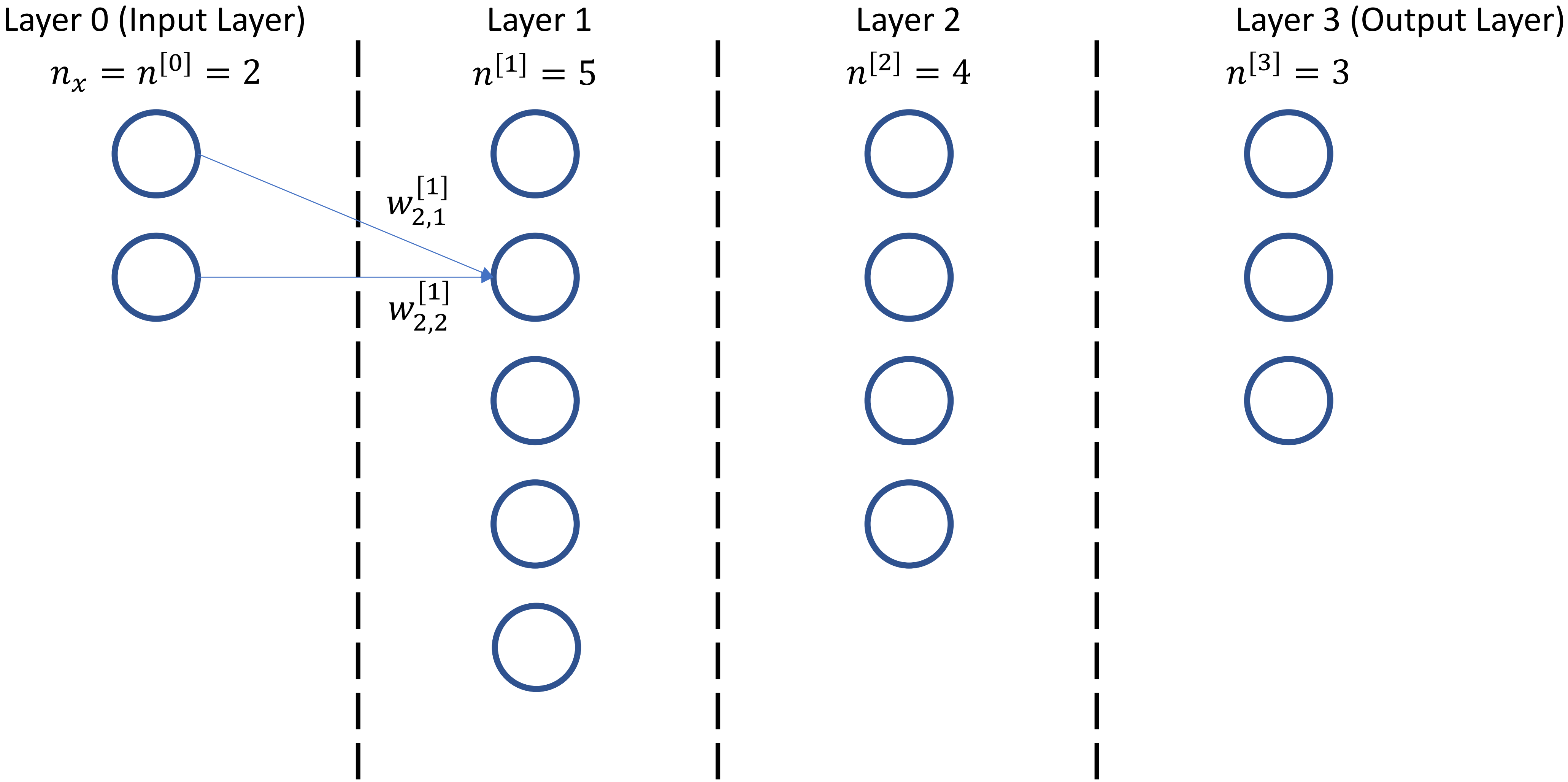
Layer 3 (Output Layer)

$n^{[3]} = 3$

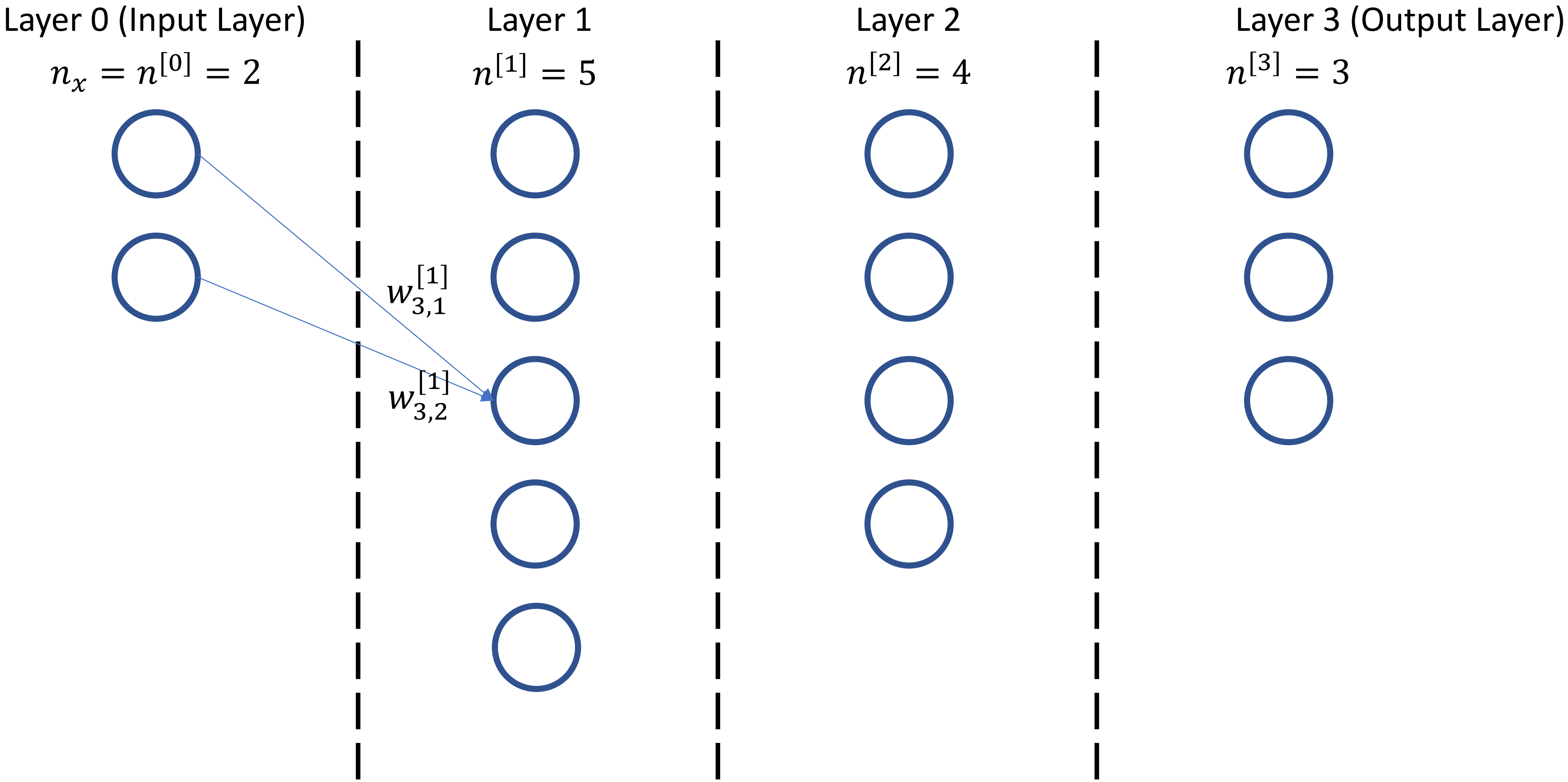
- Consider the following neural network
- This is not a compute graph. Each node represents a neuron



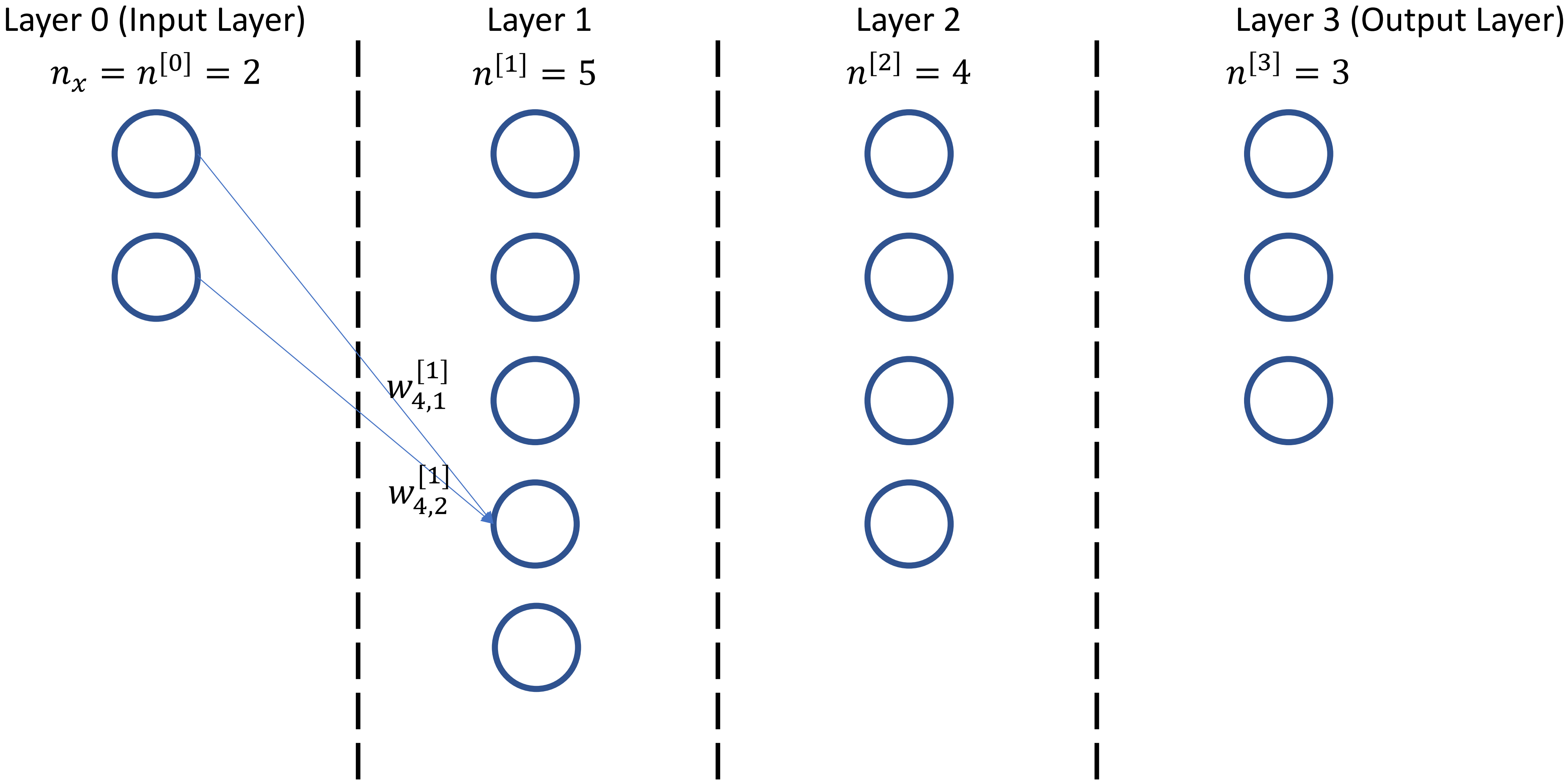
- Each node has a weight associated with a node from previous layer



- Each node has a weight associated with a node from previous layer



- Each node has a weight associated with a node from previous layer



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

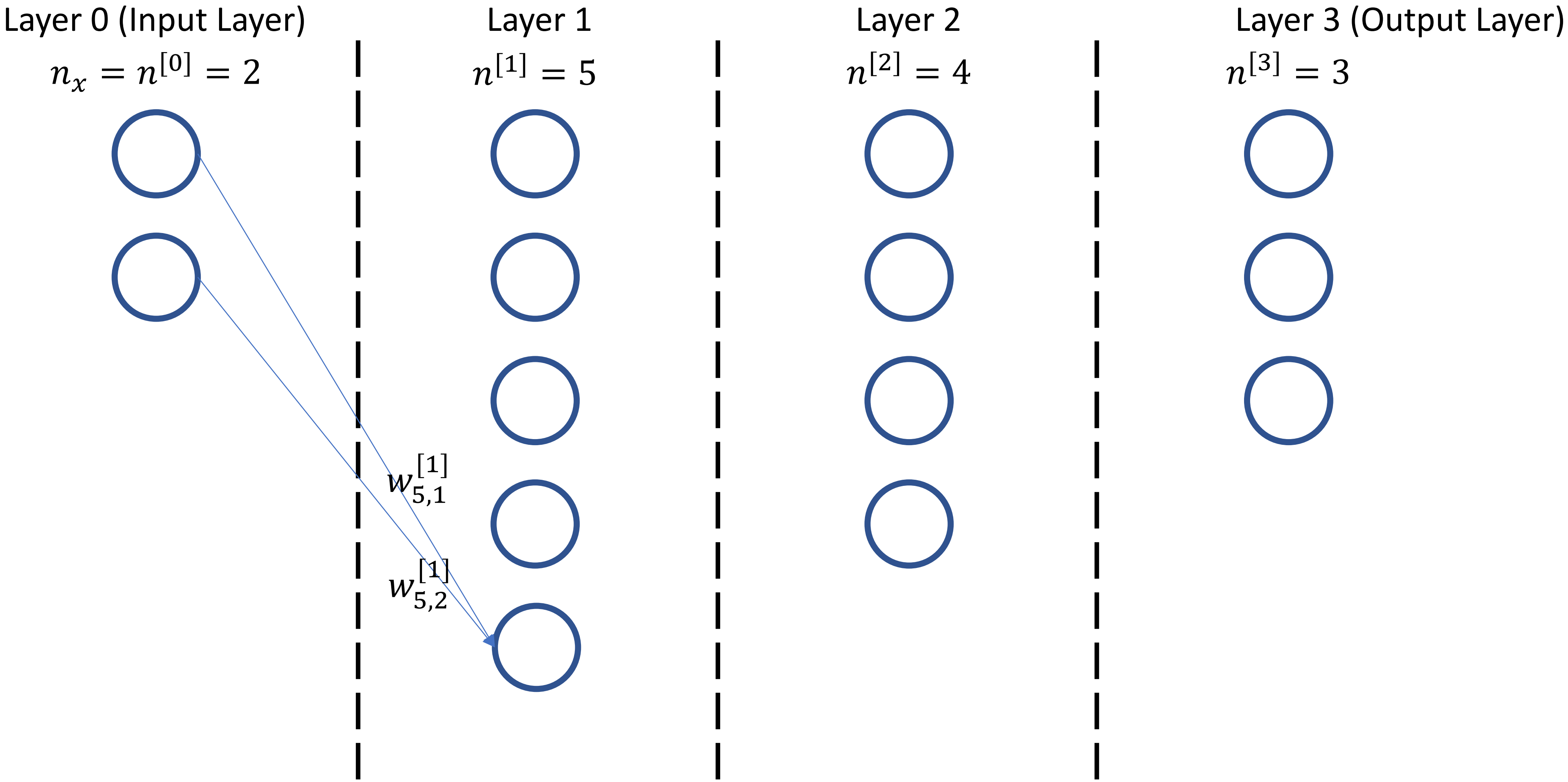
Layer 3 (Output Layer)

$n^{[3]} = 3$

$w_{4,1}^{[1]}$

$w_{4,2}^{[1]}$

- Each node has a weight associated with a node from previous layer



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$

$w_{5,1}^{[1]}$

$w_{5,2}^{[1]}$

- Each node has a weight associated with a node from previous layer
- i.e. each node in layer l has $n^{[l-1]}$ weights
- There are $n^{[l]}$nodes in layer l
- Therefore, there are $n^{[l]} * n^{[l-1]}$ weights associated with a layer

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

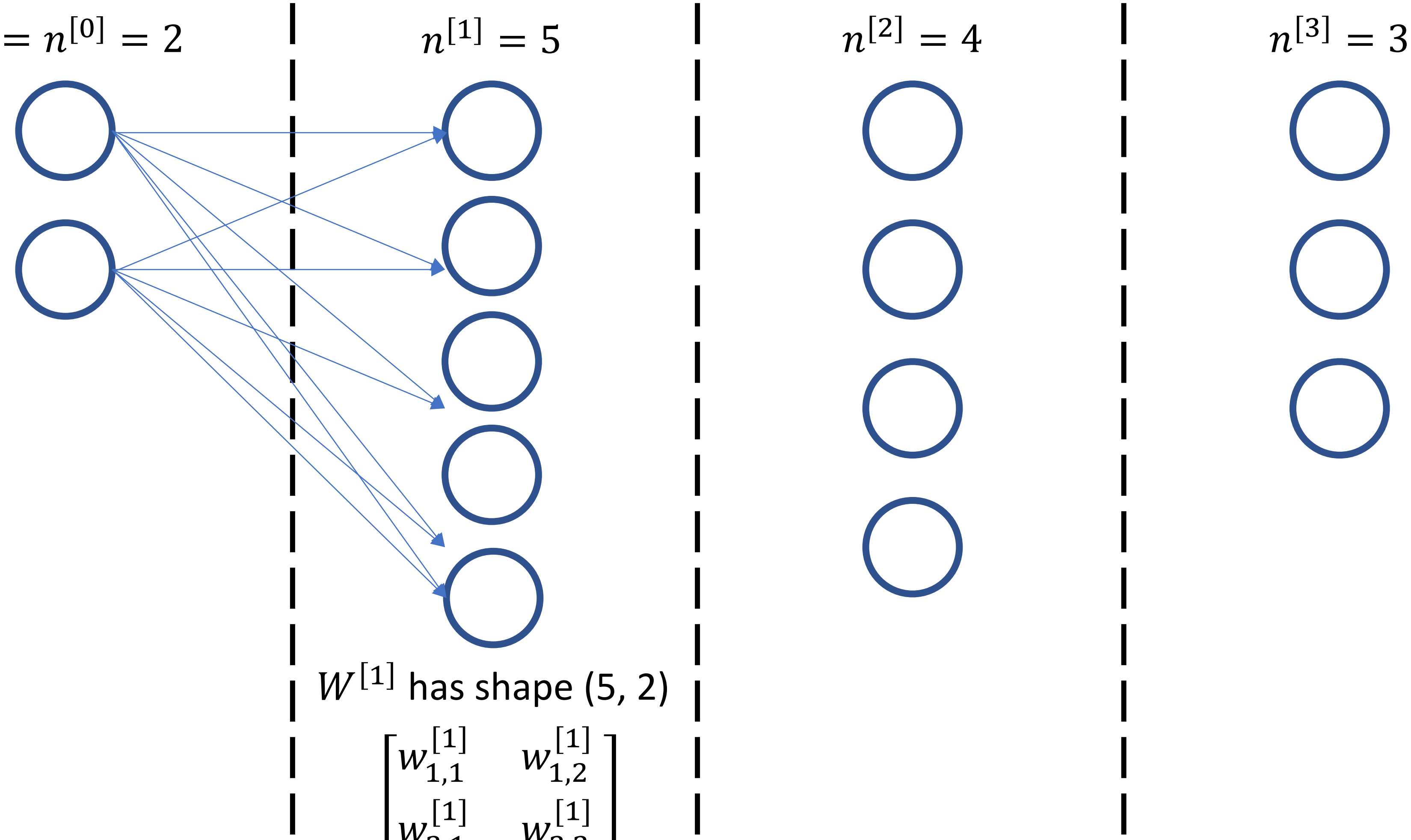
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$W^{[1]}$ has shape (5, 2)

$$W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} \\ w_{3,1}^{[1]} & w_{3,2}^{[1]} \\ w_{4,1}^{[1]} & w_{4,2}^{[1]} \\ w_{5,1}^{[1]} & w_{5,2}^{[1]} \end{bmatrix}$$

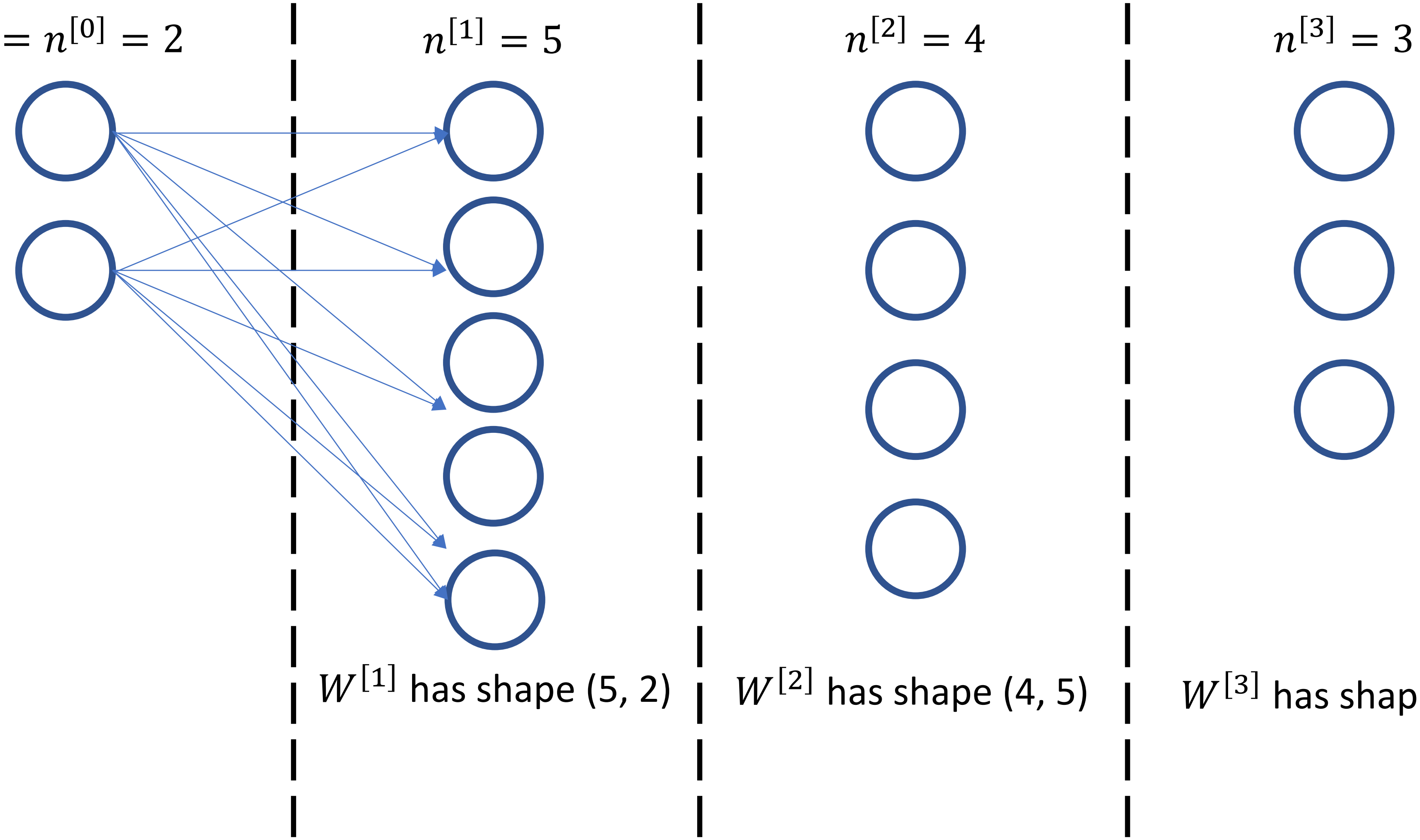
- We can write all the weights for a layer as a matrix with shape $\left(n^{[l]}, n^{[l-1]}\right)$

Layer 0 (Input Layer) $n_x = n^{[0]} = 2$

Layer 1 $n^{[1]} = 5$

Layer 2 $n^{[2]} = 4$

Layer 3 (Output Layer) $n^{[3]} = 3$

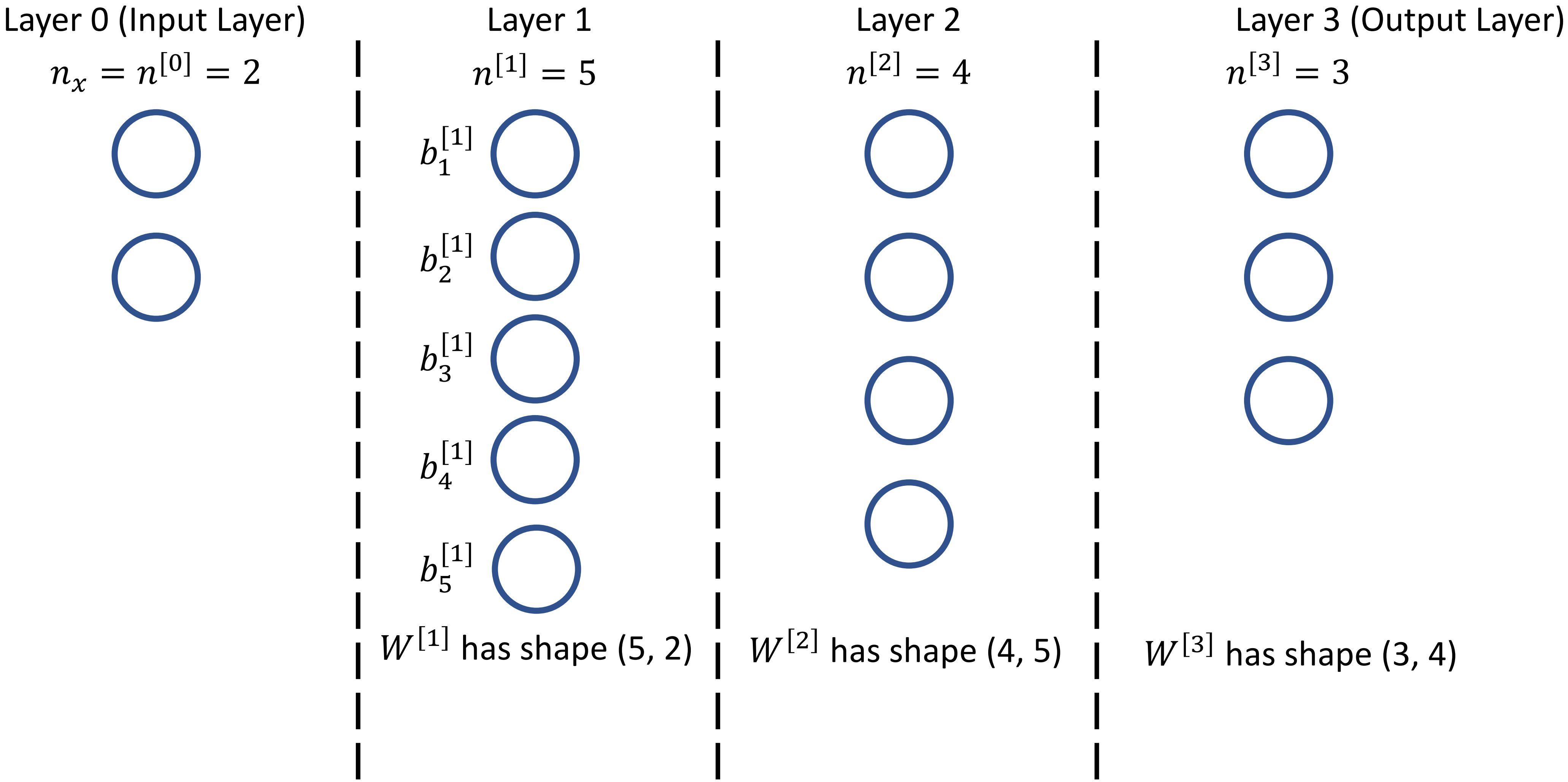


$W^{[1]}$ has shape (5, 2)

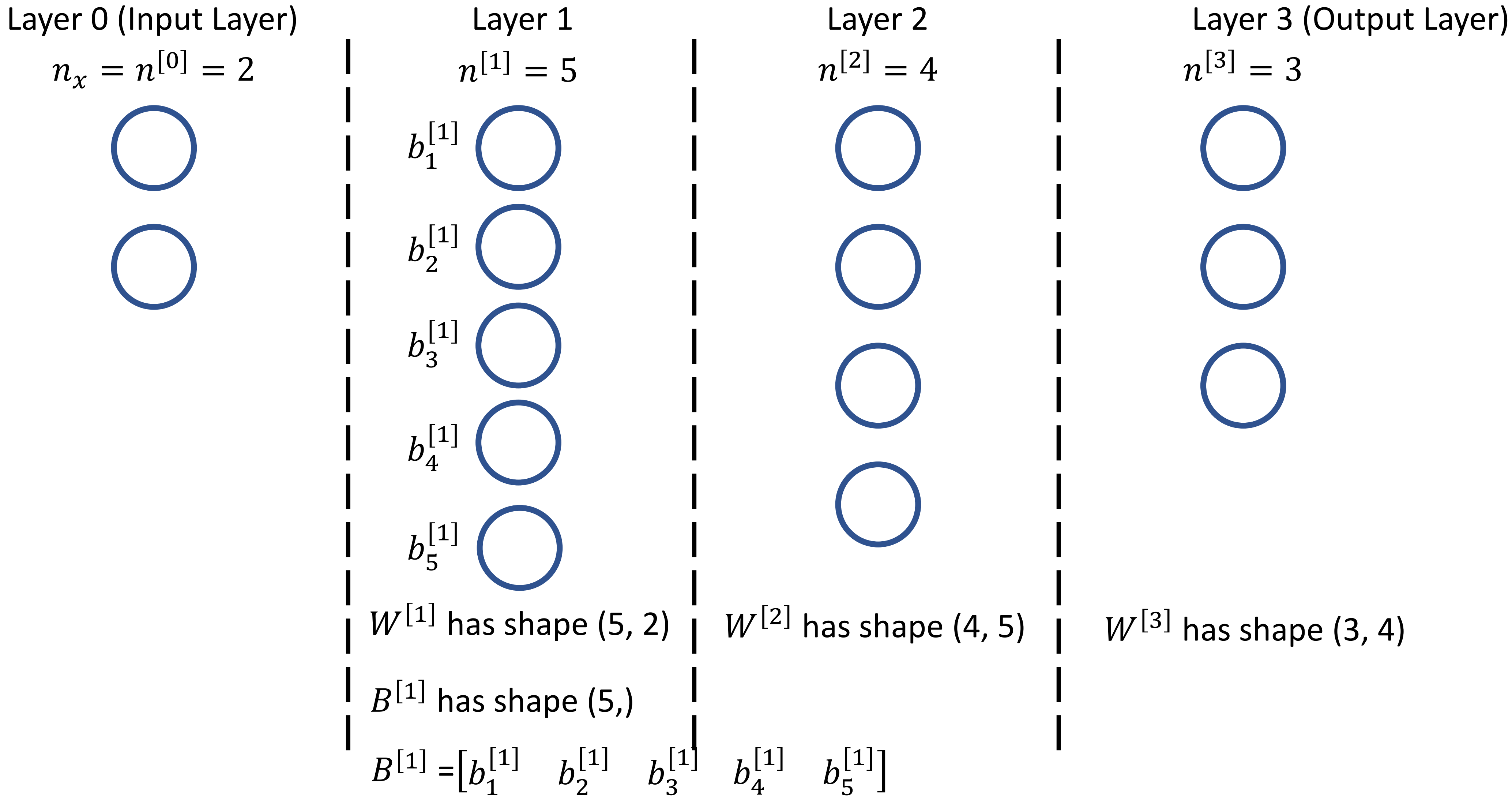
$W^{[2]}$ has shape (4, 5)

$W^{[3]}$ has shape (3, 4)

- We can write all the weights for a layer as a matrix with shape $\left(n^{[l]}, n^{[l-1]}\right)$



- Each node has its own bias parameter
- There are $n^{[l]}$nodes in layer l
- Therefore, there are $n^{[l]}$ biases associated with a layer



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

$b_1^{[1]}$

$b_2^{[1]}$

$b_3^{[1]}$

$b_4^{[1]}$

$b_5^{[1]}$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$B^{[1]} = \begin{bmatrix} b_1^{[1]} & b_2^{[1]} & b_3^{[1]} & b_4^{[1]} & b_5^{[1]} \end{bmatrix}$

Layer 2

$n^{[2]} = 4$

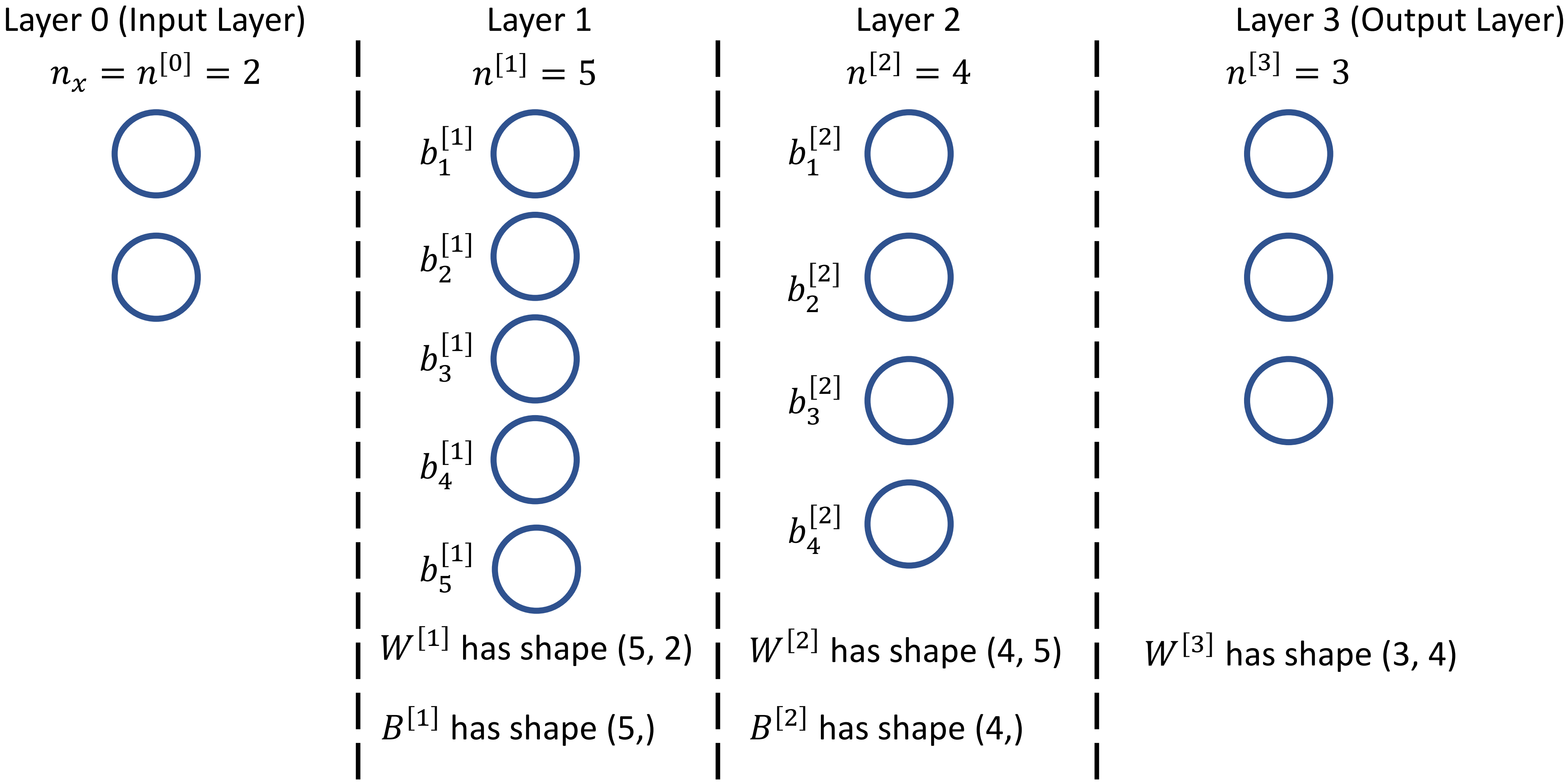
$W^{[2]}$ has shape (4, 5)

Layer 3 (Output Layer)

$n^{[3]} = 3$

$W^{[3]}$ has shape (3, 4)

- We can write all the biases for a layer as a vector with shape $(n^{[l]},)$



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

$b_1^{[1]}$ $b_2^{[1]}$ $b_3^{[1]}$ $b_4^{[1]}$ $b_5^{[1]}$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

Layer 2

$n^{[2]} = 4$

$b_1^{[2]}$ $b_2^{[2]}$ $b_3^{[2]}$ $b_4^{[2]}$

$W^{[2]}$ has shape (4, 5)

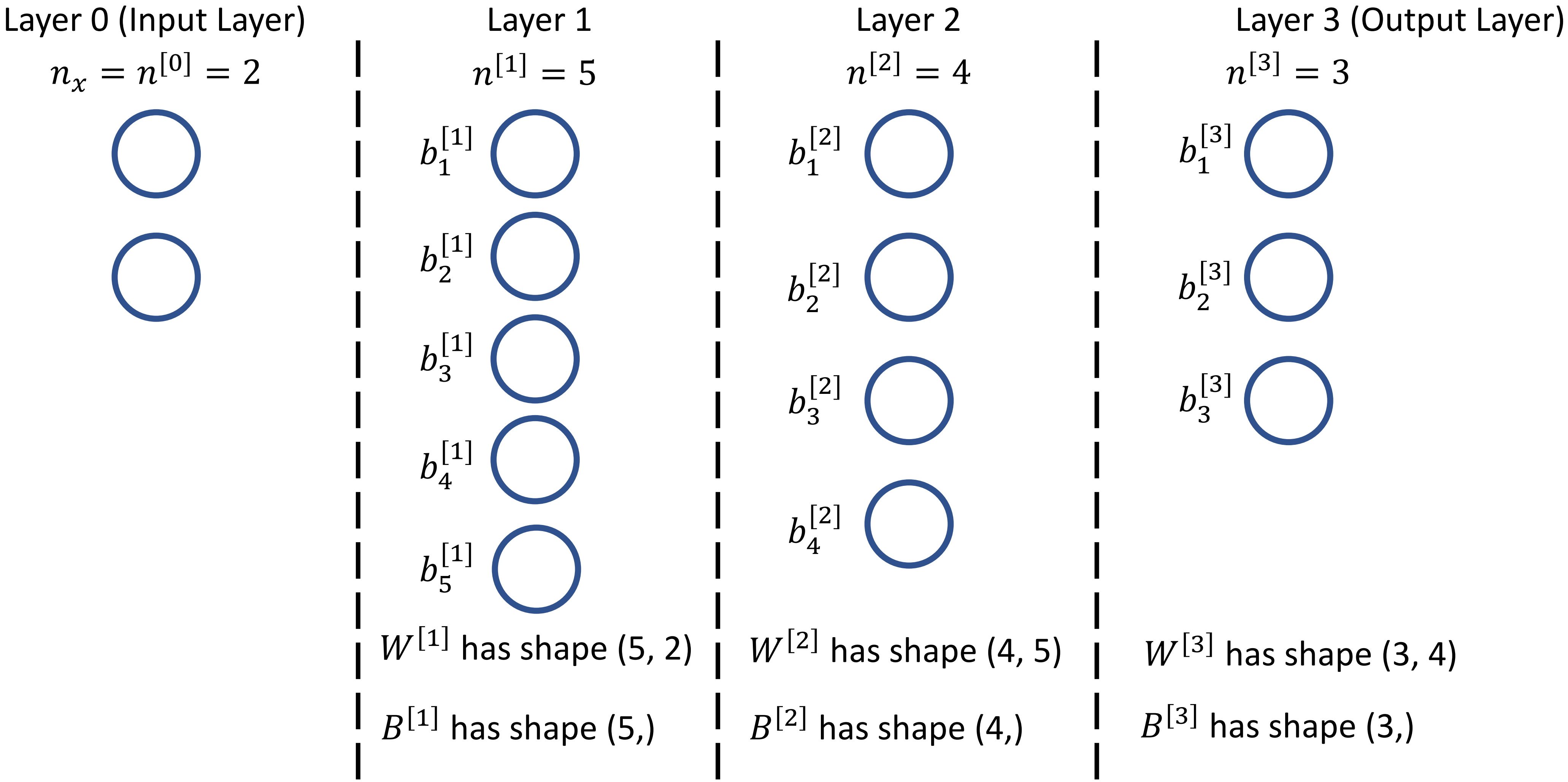
$B^{[2]}$ has shape (4,)

Layer 3 (Output Layer)

$n^{[3]} = 3$

$W^{[3]}$ has shape (3, 4)

- We can write all the biases for a layer as a vector with shape $\left(n^{[l]},\right)$



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

$b_1^{[1]}$ $b_2^{[1]}$ $b_3^{[1]}$ $b_4^{[1]}$ $b_5^{[1]}$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

Layer 2

$n^{[2]} = 4$

$b_1^{[2]}$ $b_2^{[2]}$ $b_3^{[2]}$ $b_4^{[2]}$

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

Layer 3 (Output Layer)

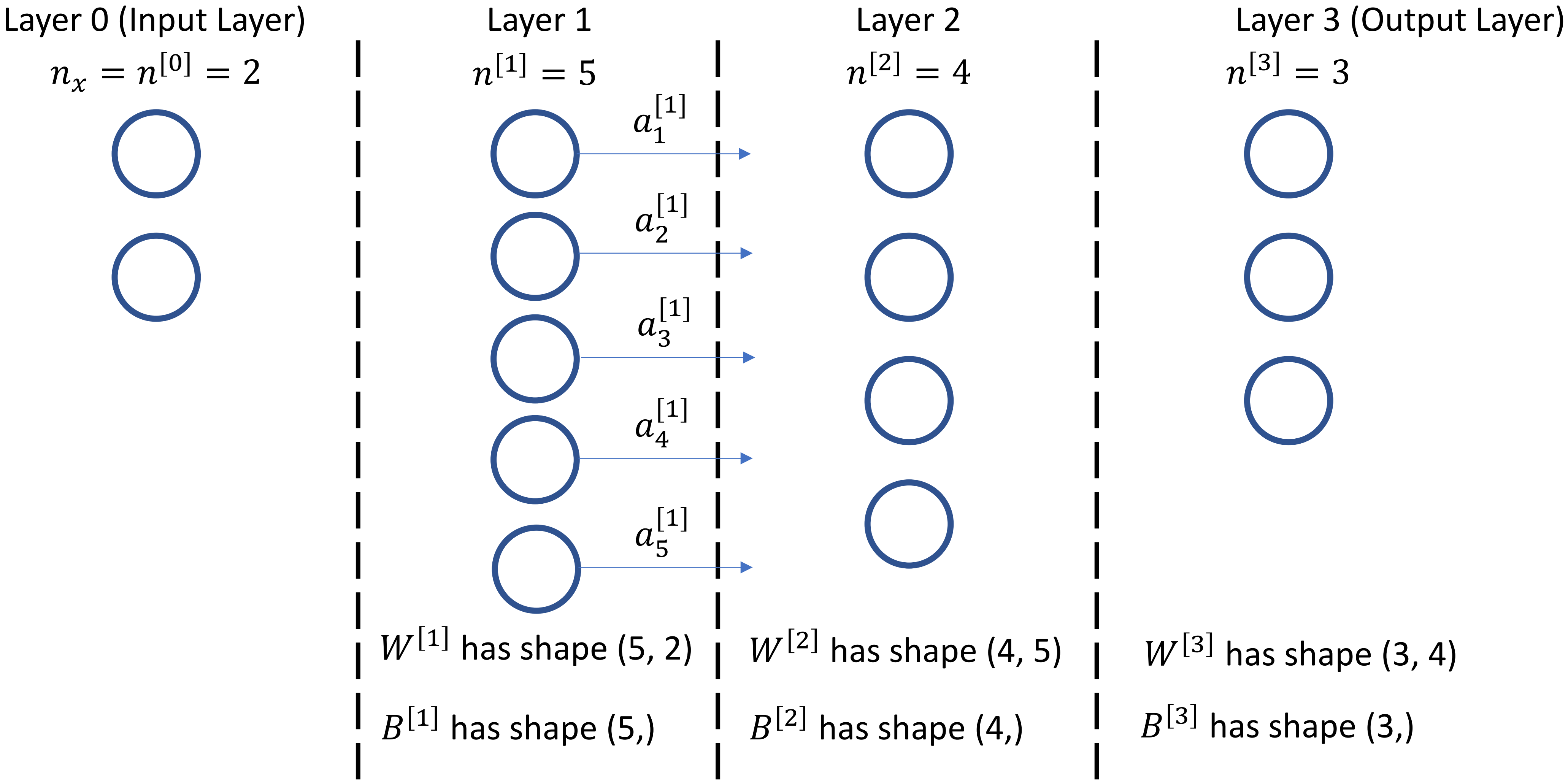
$n^{[3]} = 3$

$b_1^{[3]}$ $b_2^{[3]}$ $b_3^{[3]}$

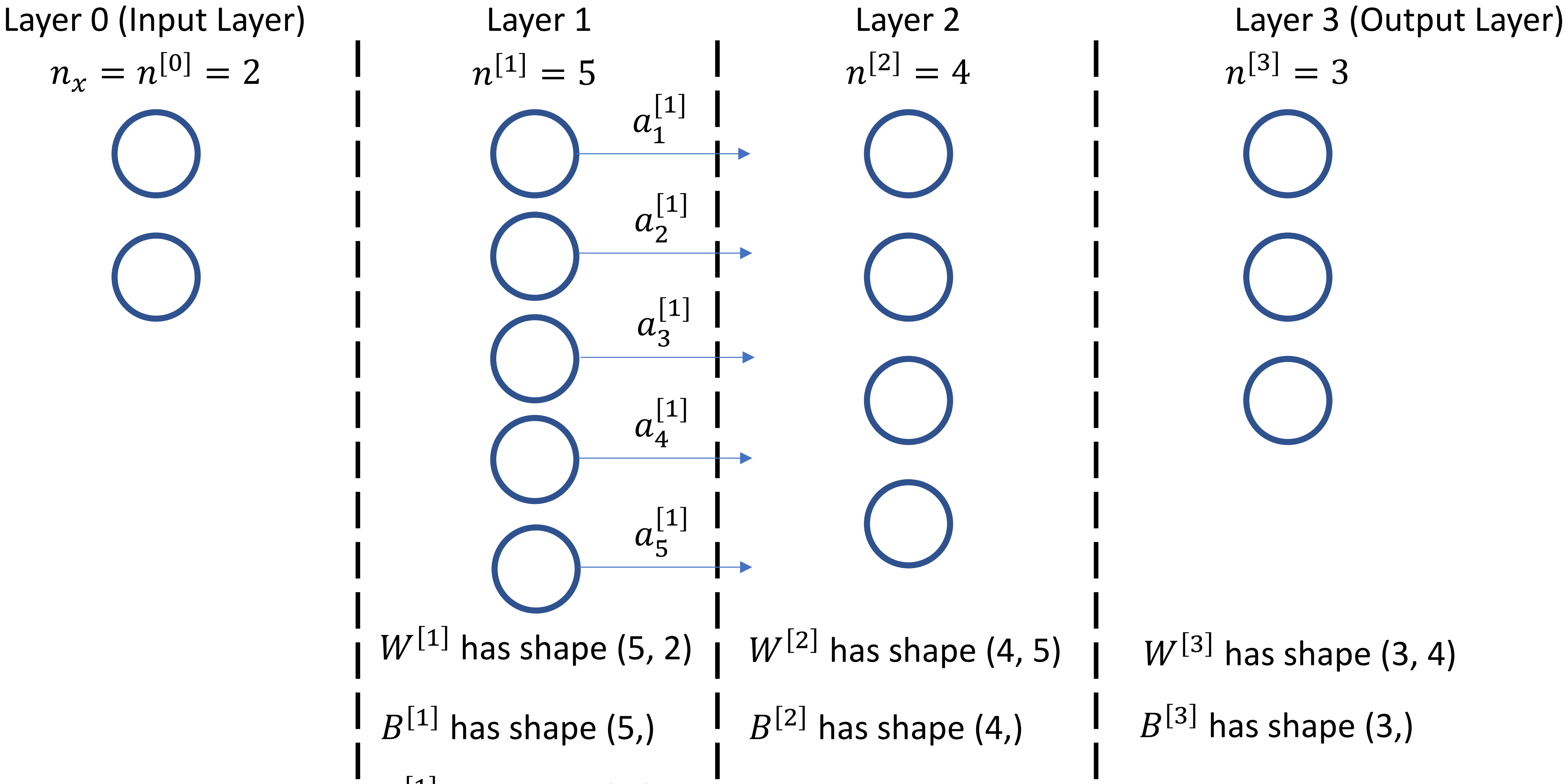
$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

- We can write all the biases for a layer as a vector with shape $(n^{[l]},)$



- Each node in a layer produces an output
- There are $n^{[l]}$nodes in layer l
- Therefore, there are $n^{[l]}$ outputs from each layer



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

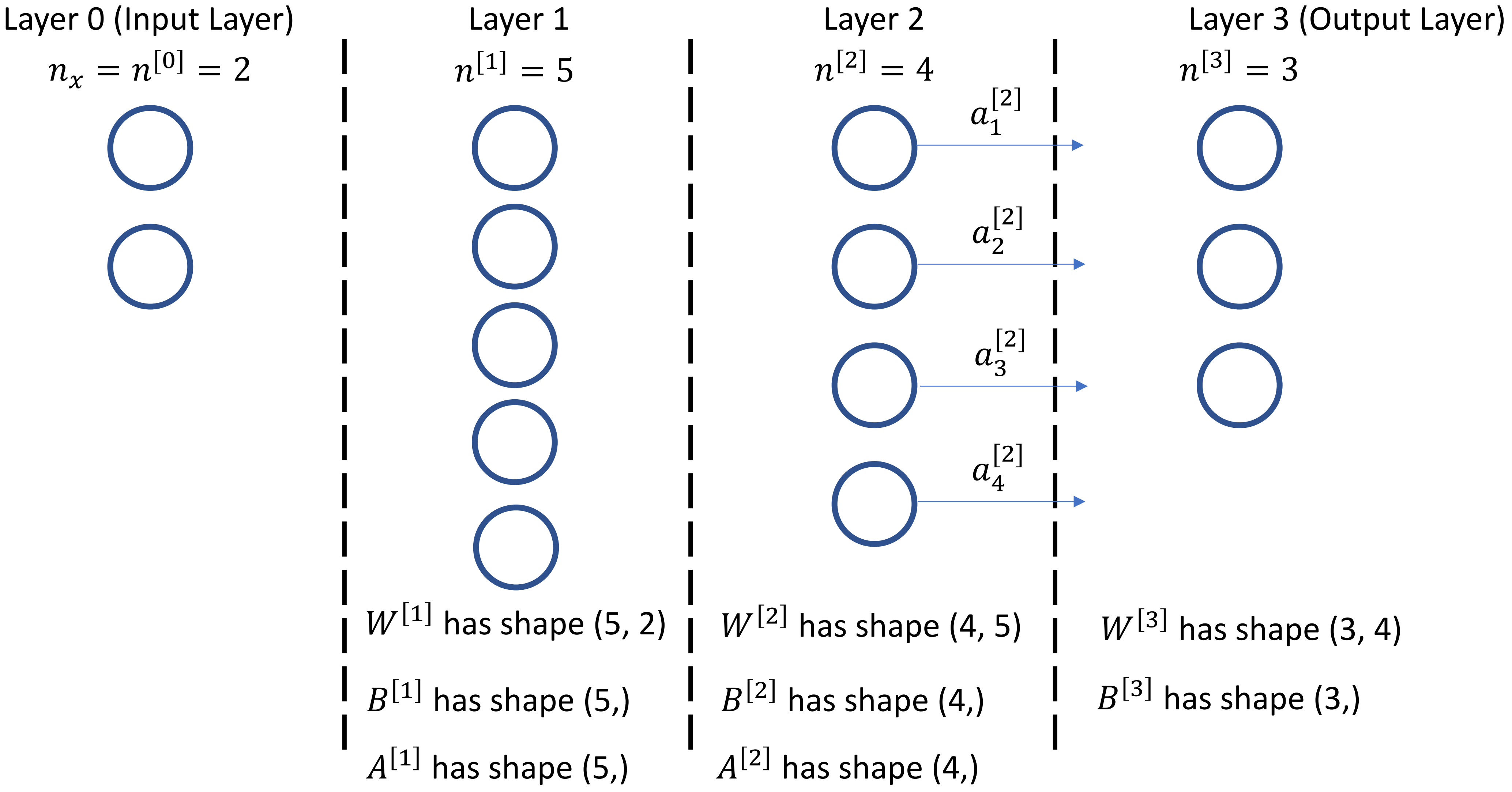
$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$A^{[1]}$ has shape (5,)

$A^{[1]} = [a_1^{[1]} \quad a_2^{[1]} \quad a_3^{[1]} \quad a_4^{[1]} \quad a_5^{[1]}]$

- We can write all the outputs for a layer as a vector with shape $(n^{[l]},)$



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

Layer 2

$n^{[2]} = 4$

$a_1^{[2]}$

$a_2^{[2]}$

$a_3^{[2]}$

$a_4^{[2]}$

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

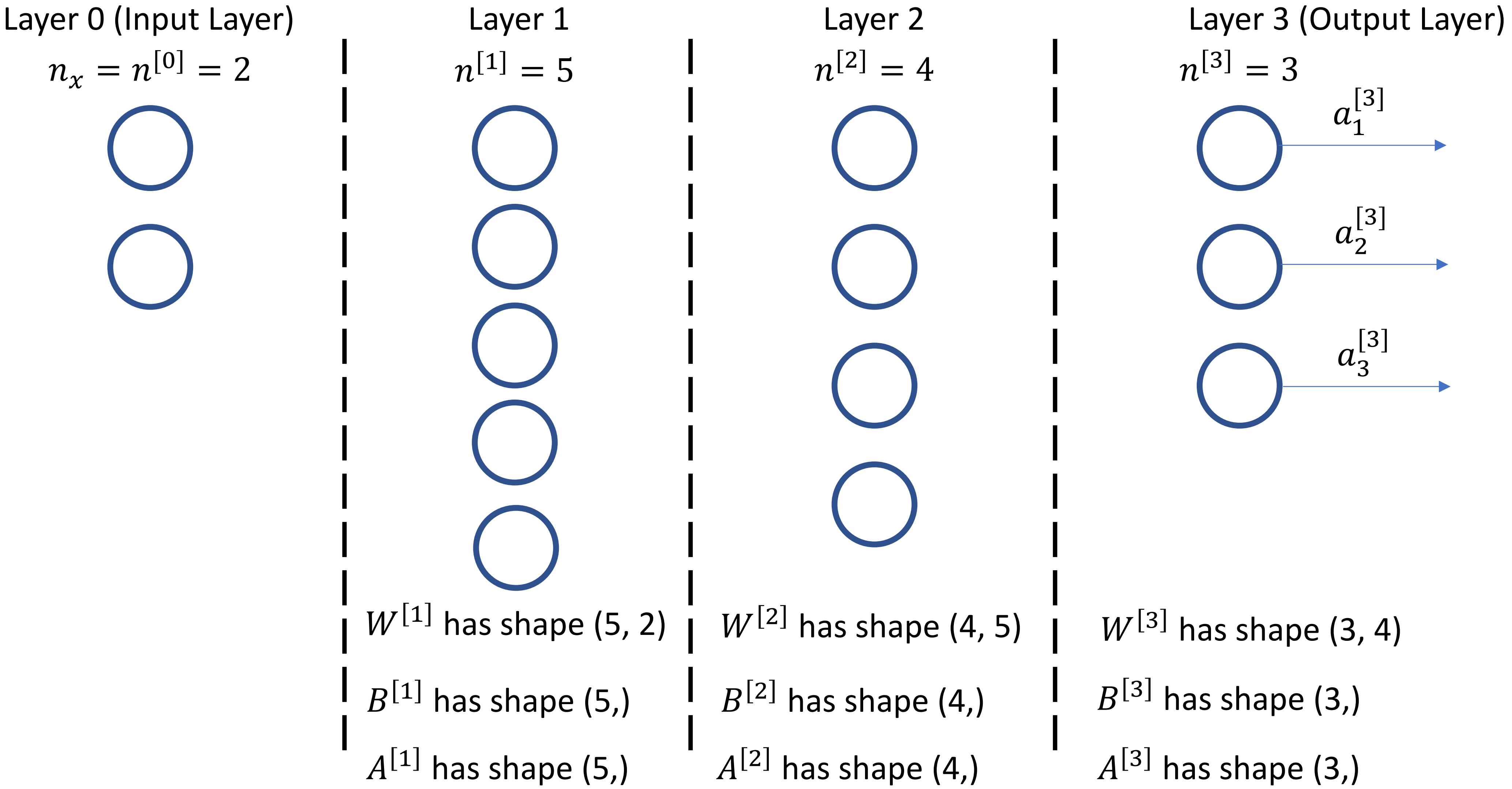
Layer 3 (Output Layer)

$n^{[3]} = 3$

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

- We can write all the outputs for a layer as a vector with shape $\left(n^{[l]},\right)$



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

Layer 2

$n^{[2]} = 4$

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

Layer 3 (Output Layer)

$n^{[3]} = 3$

$a_1^{[3]}$

$a_2^{[3]}$

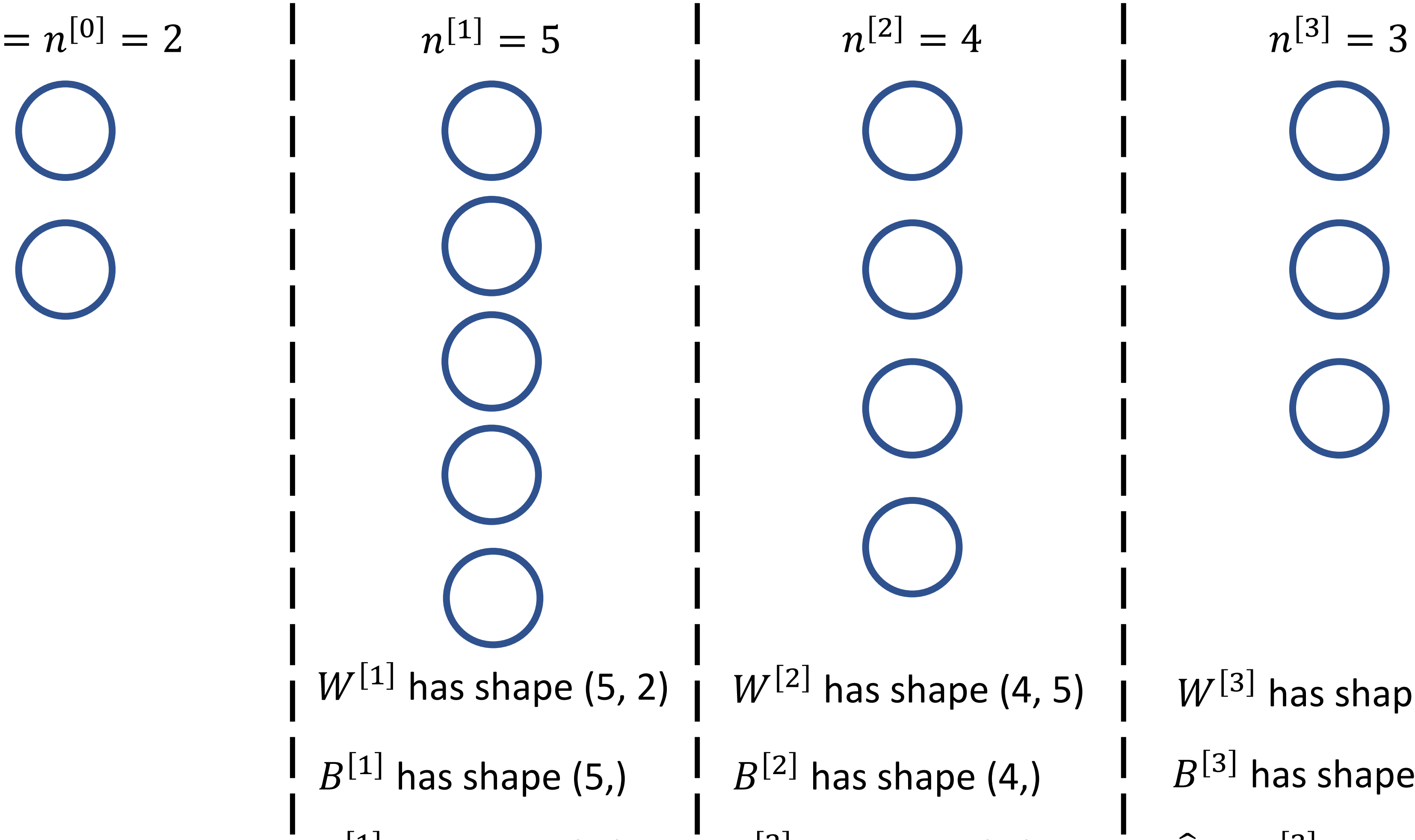
$a_3^{[3]}$

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$A^{[3]}$ has shape (3,)

- We can write all the outputs for a layer as a vector with shape $\left(n^{[l]},\right)$



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

$A^{[0]} = X$ has shape (2,)

Layer 1

$n^{[1]} = 5$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

Layer 2

$n^{[2]} = 4$

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

Layer 3 (Output Layer)

$n^{[3]} = 3$

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

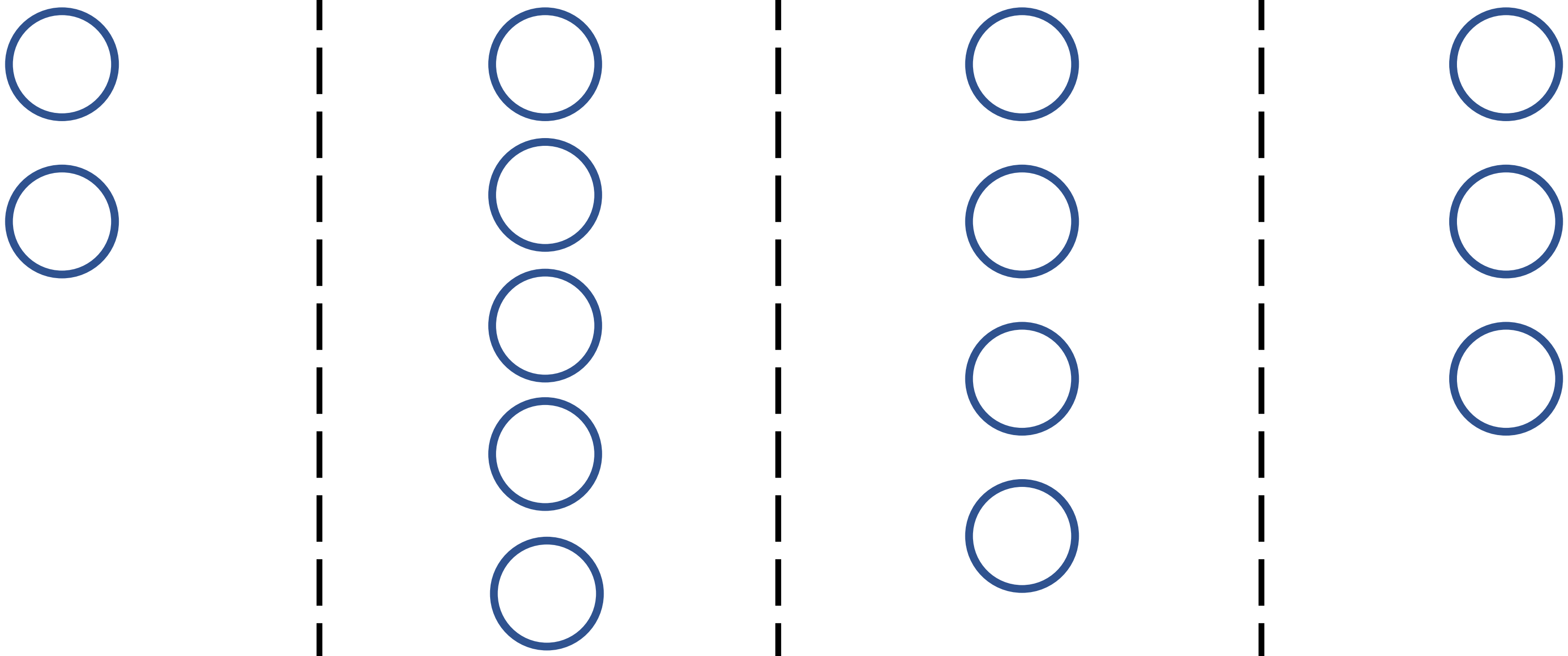
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$ has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

Computing the output of one i node in layer l:

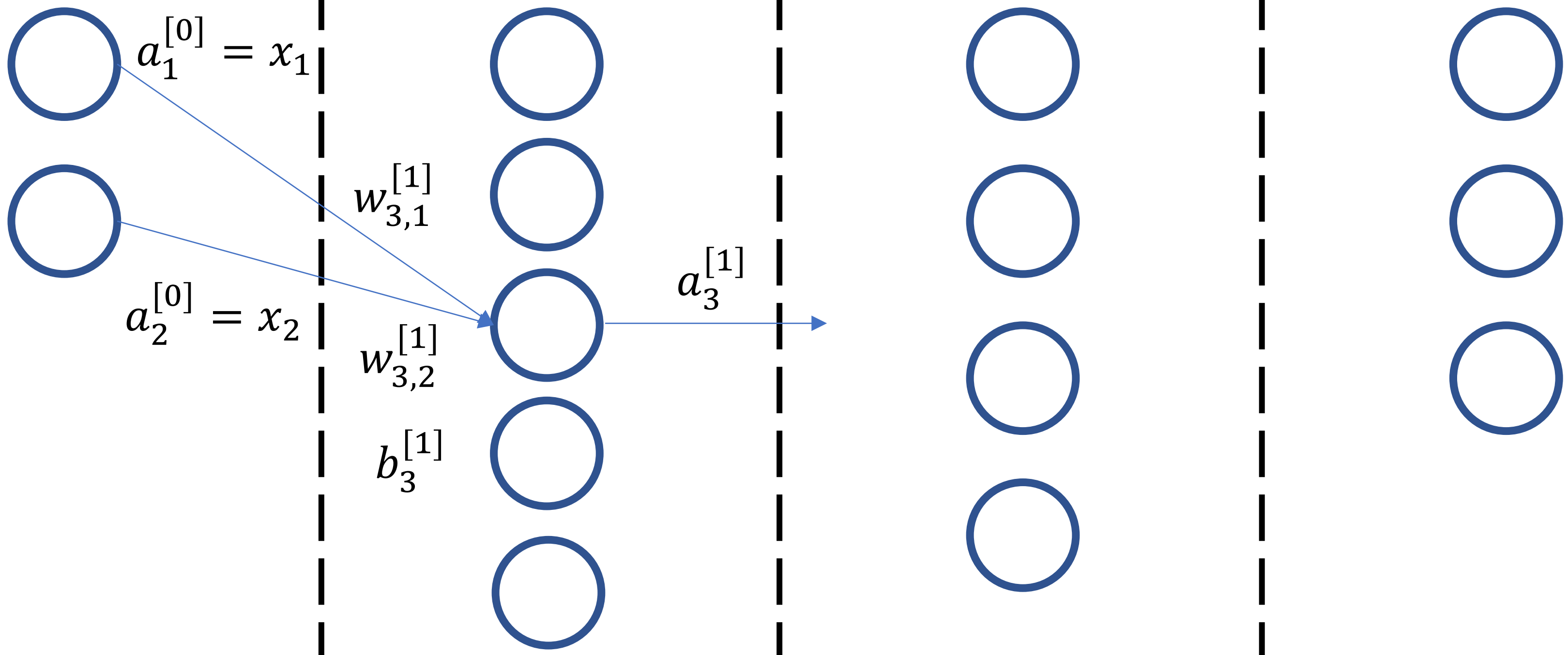
$$z_i^{[l]} = \sum_{j=1}^{n^{[l-1]}} w_{i,1}^{[l]}\, a_j^{[l-1]} + b_i^{[l]} \qquad a_i^{[l]} = g(z_i^{[l]})$$

Layer 0 (Input Layer) $n_x = n^{[0]} = 2$

Layer 1 $n^{[1]} = 5$

Layer 2 $n^{[2]} = 4$

Layer 3 (Output Layer) $n^{[3]} = 3$



$a_1^{[0]} = x_1$

$a_2^{[0]} = x_2$

$w_{3,1}^{[1]}$

$w_{3,2}^{[1]}$

$b_3^{[1]}$

$a_3^{[1]}$

$A^{[0]} = X$ has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

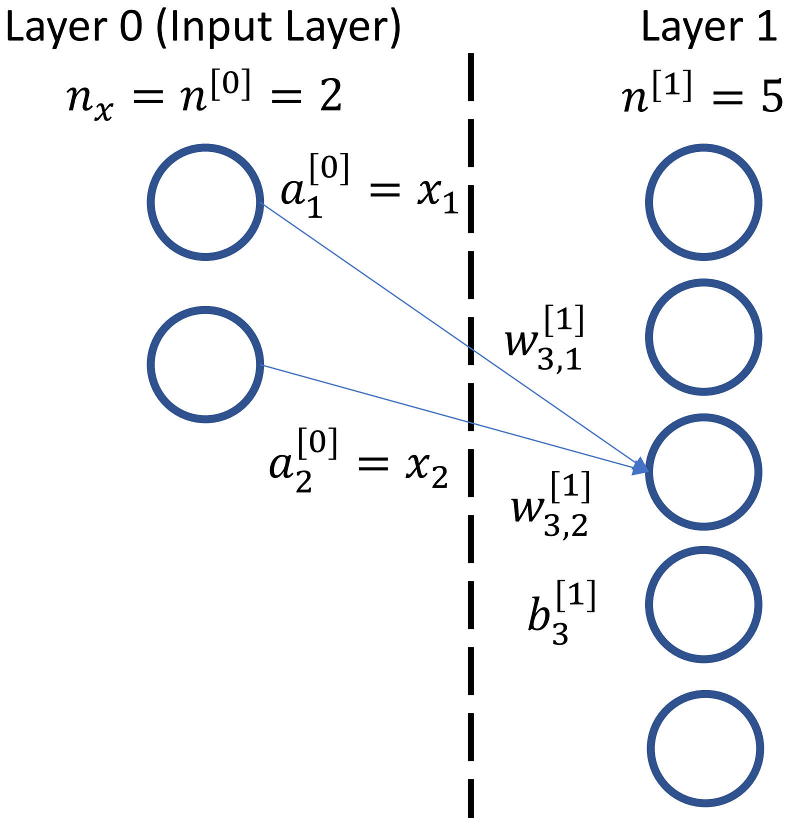
Computing the output of one i node in layer l:

$$z_i^{[l]} = \sum_{j=1}^{n^{[l-1]}} w_{i,1}^{[l]}\, a_j^{[l-1]} + b_i^{[l]} \qquad a_i^{[l]} = g(z_i^{[l]})$$

$$z_3^{[1]} = w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]}$$

$$a_3^{[1]} = g\left(z_3^{[1]}\right)$$

Brad Quinton, Scott Chin



$$z_1^{[1]} = w_{1,1}^{[1]}\, a_1^{[0]} + w_{1,2}^{[1]} a_2^{[0]} + b_1^{[1]}$$

$$z_2^{[1]} = w_{2,1}^{[1]}\, a_1^{[0]} + w_{2,2}^{[1]} a_2^{[0]} + b_2^{[1]}$$

$$z_3^{[1]} = w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]}$$

$$z_4^{[1]} = w_{4,1}^{[1]}\, a_1^{[0]} + w_{4,2}^{[1]} a_2^{[0]} + b_4^{[1]}$$

$$z_5^{[1]} = w_{5,1}^{[1]}\, a_1^{[0]} + w_{5,2}^{[1]} a_2^{[0]} + b_5^{[1]}$$

$W^{[1]}$ has shape (5, 2)

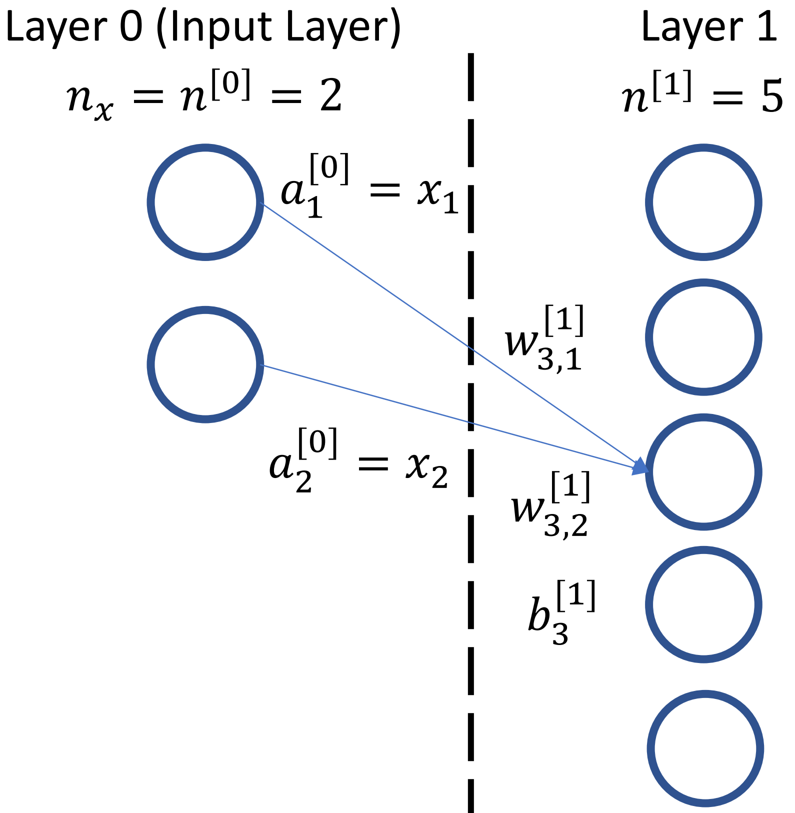
$B^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

- We can write all the z terms for a layer as a vector with shape $\left(n^{[l]},\right)$



$$z_1^{[1]} = w_{1,1}^{[1]}\, a_1^{[0]} + w_{1,2}^{[1]} a_2^{[0]} + b_1^{[1]}$$

$$z_2^{[1]} = w_{2,1}^{[1]}\, a_1^{[0]} + w_{2,2}^{[1]} a_2^{[0]} + b_2^{[1]}$$

$$z_3^{[1]} = w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]}$$

$$z_4^{[1]} = w_{4,1}^{[1]}\, a_1^{[0]} + w_{4,2}^{[1]} a_2^{[0]} + b_4^{[1]}$$

$$z_5^{[1]} = w_{5,1}^{[1]}\, a_1^{[0]} + w_{5,2}^{[1]} a_2^{[0]} + b_5^{[1]}$$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

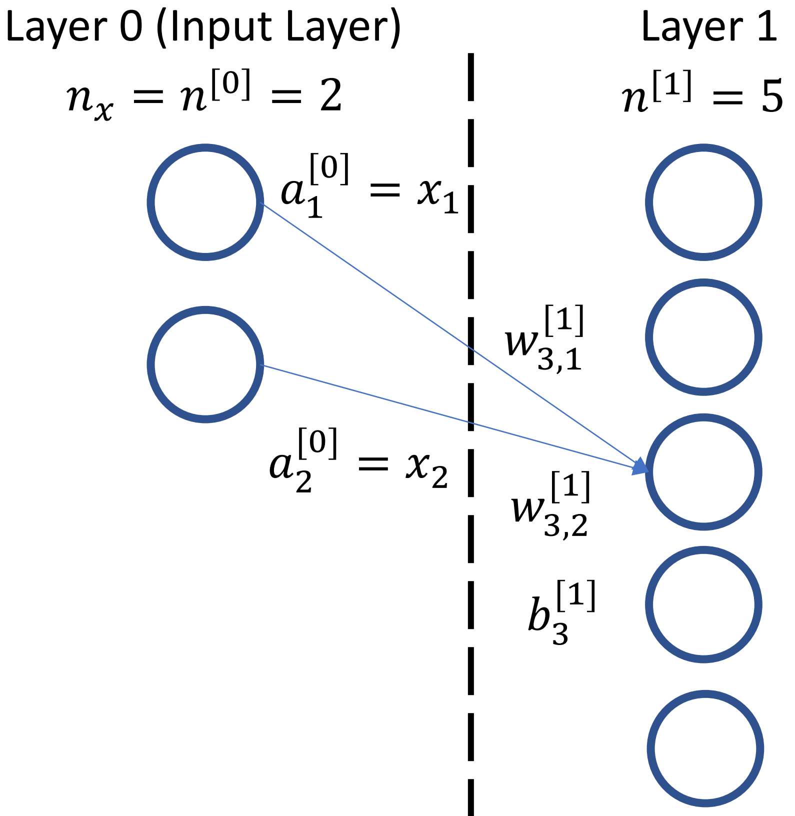
$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$$Z^{[1]} = matmul\left(W^{[1]}, A^{[0]}\right) + B^{[1]}$$

```
# In NumPy
Z1 = np.matmul(W1, A0) + B1
```

- Instead of computing each of these 5 z terms one at a time, we can do it in a single vectorized operation using matrix operations



$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$$z_1^{[1]} = w_{1,1}^{[1]}\, a_1^{[0]} + w_{1,2}^{[1]} a_2^{[0]} + b_1^{[1]}$$

$$z_2^{[1]} = w_{2,1}^{[1]}\, a_1^{[0]} + w_{2,2}^{[1]} a_2^{[0]} + b_2^{[1]}$$

$$z_3^{[1]} = w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]}$$

$$z_4^{[1]} = w_{4,1}^{[1]}\, a_1^{[0]} + w_{4,2}^{[1]} a_2^{[0]} + b_4^{[1]}$$

$$z_5^{[1]} = w_{5,1}^{[1]}\, a_1^{[0]} + w_{5,2}^{[1]} a_2^{[0]} + b_5^{[1]}$$

$$Z^{[1]} = matmul\left(W^{[1]}, A^{[0]}\right) + B^{[1]}$$

$$= \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} \\ w_{3,1}^{[1]} & w_{3,2}^{[1]} \\ w_{4,1}^{[1]} & w_{4,2}^{[1]} \\ w_{5,1}^{[1]} & w_{5,2}^{[1]} \end{bmatrix} \begin{bmatrix} a_1^{[0]} \\ a_2^{[0]} \end{bmatrix} + \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \\ b_4^{[1]} \\ b_5^{[1]} \end{bmatrix} = \begin{bmatrix} w_{1,1}^{[1]}\, a_1^{[0]} + w_{1,2}^{[1]} a_2^{[0]} + b_1^{[1]} \\ w_{2,1}^{[1]}\, a_1^{[0]} + w_{2,2}^{[1]} a_2^{[0]} + b_2^{[1]} \\ w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]} \\ w_{4,1}^{[1]}\, a_1^{[0]} + w_{4,2}^{[1]} a_2^{[0]} + b_4^{[1]} \\ w_{5,1}^{[1]}\, a_1^{[0]} + w_{5,2}^{[1]} a_2^{[0]} + b_5^{[1]} \end{bmatrix} = \begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \\ z_3^{[1]} \\ z_4^{[1]} \\ z_5^{[1]} \end{bmatrix}$$

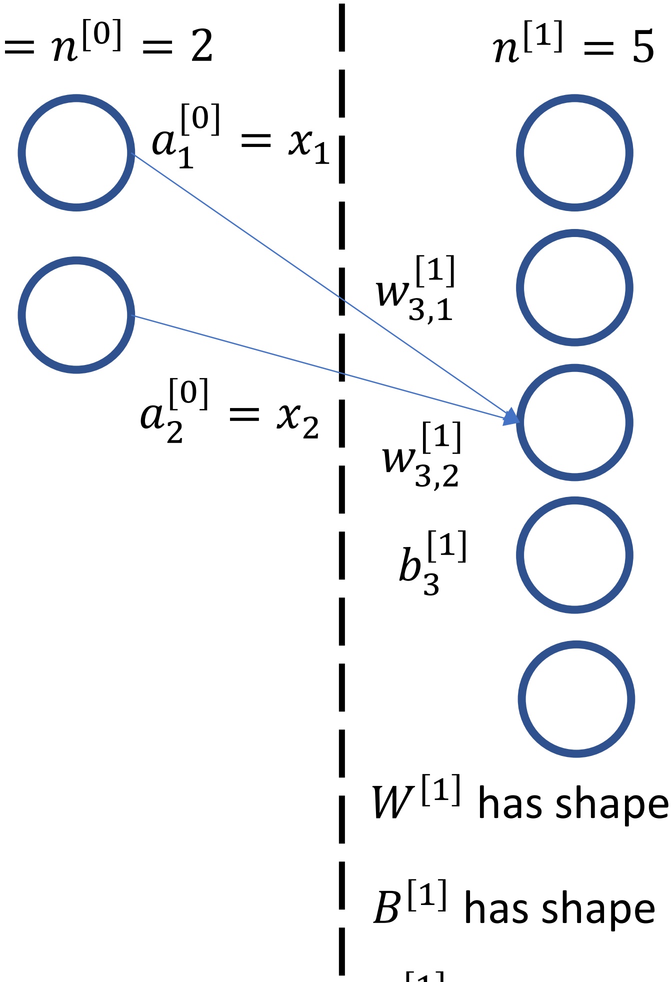
Brad Quinton, Scott Chin

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$a_1^{[0]} = x_1$

$w_{3,1}^{[1]}$

$a_2^{[0]} = x_2$

$w_{3,2}^{[1]}$

$b_3^{[1]}$

$$z_1^{[1]} = w_{1,1}^{[1]}\,a_1^{[0]} + w_{1,2}^{[1]}a_2^{[0]} + b_1^{[1]} \qquad a_1^{[1]} = g\left(z_1^{[1]}\right)$$

$$z_2^{[1]} = w_{2,1}^{[1]}\,a_1^{[0]} + w_{2,2}^{[1]}a_2^{[0]} + b_2^{[1]} \qquad a_2^{[1]} = g\left(z_2^{[1]}\right)$$

$$z_3^{[1]} = w_{3,1}^{[1]}\,a_1^{[0]} + w_{3,2}^{[1]}a_2^{[0]} + b_3^{[1]} \qquad a_3^{[1]} = g\left(z_3^{[1]}\right)$$

$$z_4^{[1]} = w_{4,1}^{[1]}\,a_1^{[0]} + w_{4,2}^{[1]}a_2^{[0]} + b_4^{[1]} \qquad a_4^{[1]} = g\left(z_4^{[1]}\right)$$

$$z_5^{[1]} = w_{5,1}^{[1]}\,a_1^{[0]} + w_{5,2}^{[1]}a_2^{[0]} + b_5^{[1]} \qquad a_5^{[1]} = g\left(z_5^{[1]}\right)$$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

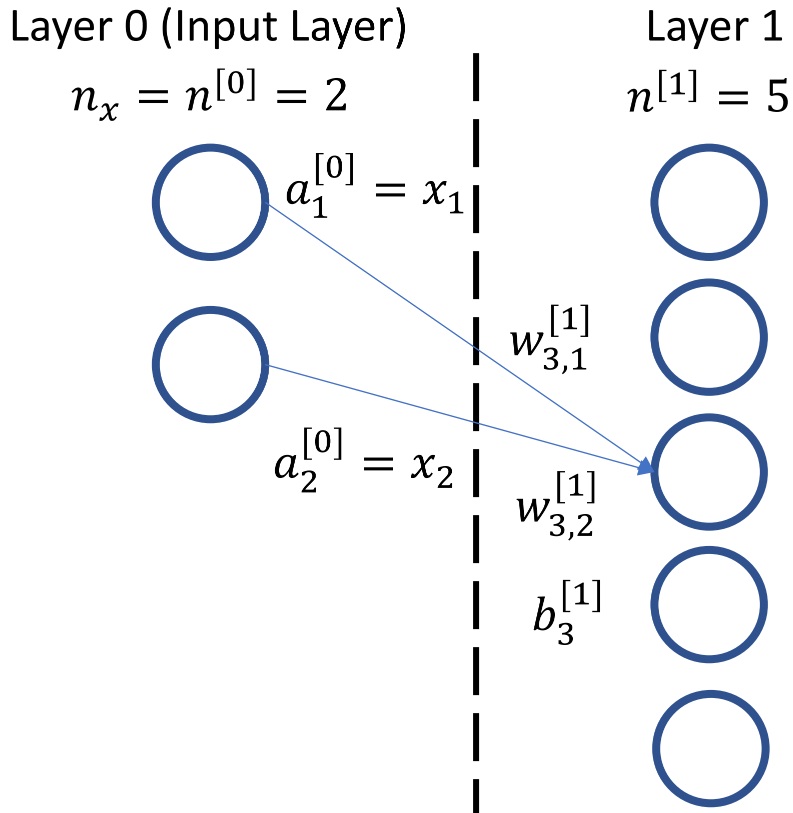
$A^{[0]} = X$ has shape (2,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$$A^{[1]} = g(Z^{[1]})$$

- We also want to use vectorized operations to compute all the activation outputs of the layer in one operation



$$z_1^{[1]} = w_{1,1}^{[1]}\, a_1^{[0]} + w_{1,2}^{[1]} a_2^{[0]} + b_1^{[1]} \qquad a_1^{[1]} = g\left(z_1^{[1]}\right)$$

$$z_2^{[1]} = w_{2,1}^{[1]}\, a_1^{[0]} + w_{2,2}^{[1]} a_2^{[0]} + b_2^{[1]} \qquad a_2^{[1]} = g\left(z_2^{[1]}\right)$$

$$z_3^{[1]} = w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]} \qquad a_3^{[1]} = g\left(z_3^{[1]}\right)$$

$$z_4^{[1]} = w_{4,1}^{[1]}\, a_1^{[0]} + w_{4,2}^{[1]} a_2^{[0]} + b_4^{[1]} \qquad a_4^{[1]} = g\left(z_4^{[1]}\right)$$

$$z_5^{[1]} = w_{5,1}^{[1]}\, a_1^{[0]} + w_{5,2}^{[1]} a_2^{[0]} + b_5^{[1]} \qquad a_5^{[1]} = g\left(z_5^{[1]}\right)$$

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$$A^{[1]} = g\left(Z^{[1]}\right)$$

$$= g\left(\begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \\ z_3^{[1]} \\ z_4^{[1]} \\ z_5^{[1]} \end{bmatrix}\right) = \begin{bmatrix} g\left(z_1^{[1]}\right) \\ g\left(z_2^{[1]}\right) \\ g\left(z_3^{[1]}\right) \\ g\left(z_4^{[1]}\right) \\ g\left(z_5^{[1]}\right) \end{bmatrix} = \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \\ a_4^{[1]} \\ a_5^{[1]} \end{bmatrix}$$

Using elementwise-vector operation

```
# In NumPy
A1 = np.tanh(Z1)
```

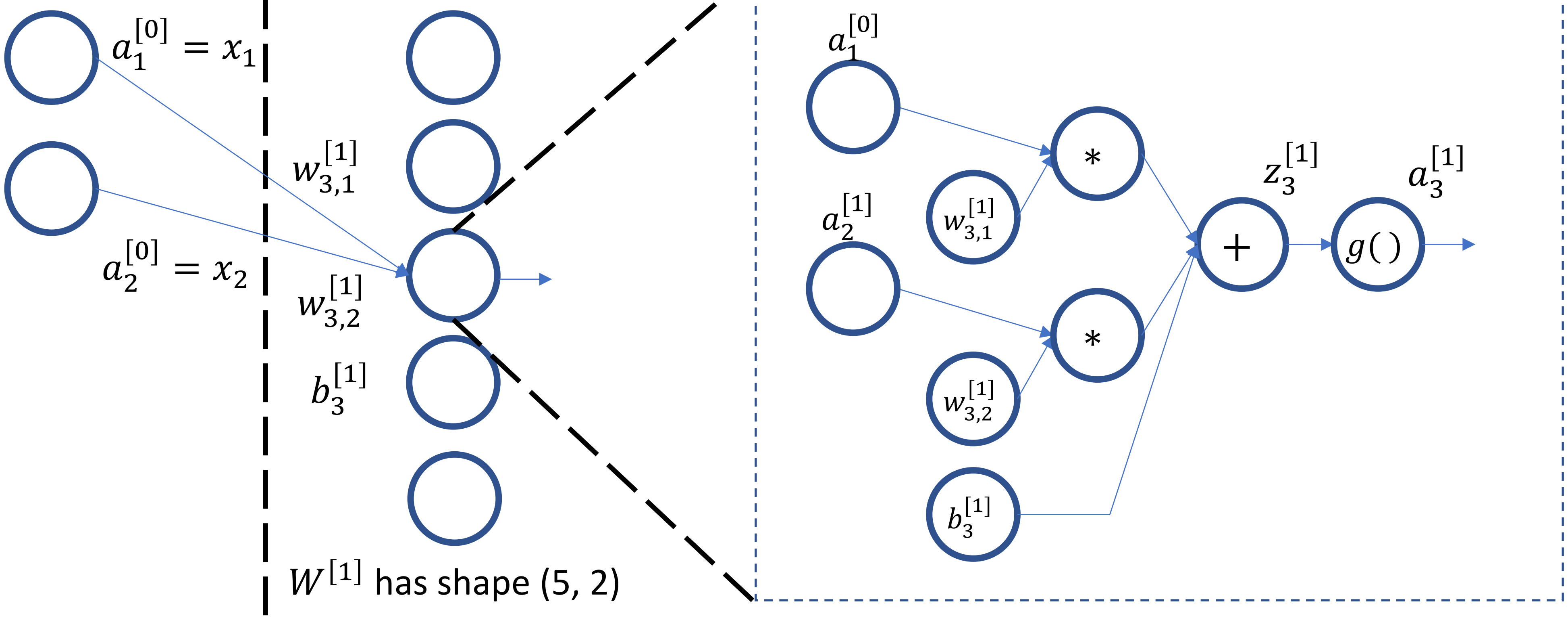
Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

A **Compute Graph** for one neuron



$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$$z_3^{[1]} = w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]}$$

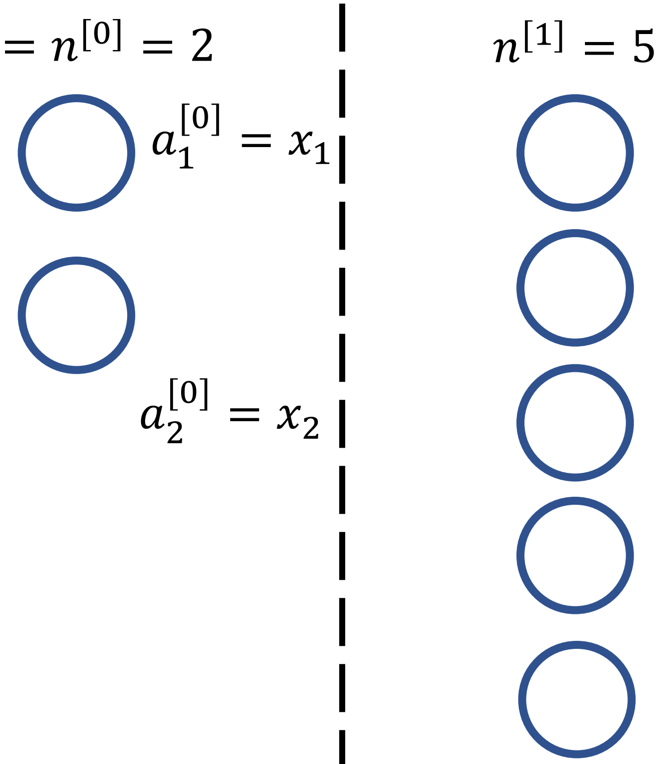
$$a_3^{[1]} = g\left(z_3^{[1]}\right)$$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$a_1^{[0]} = x_1$

$a_2^{[0]} = x_2$

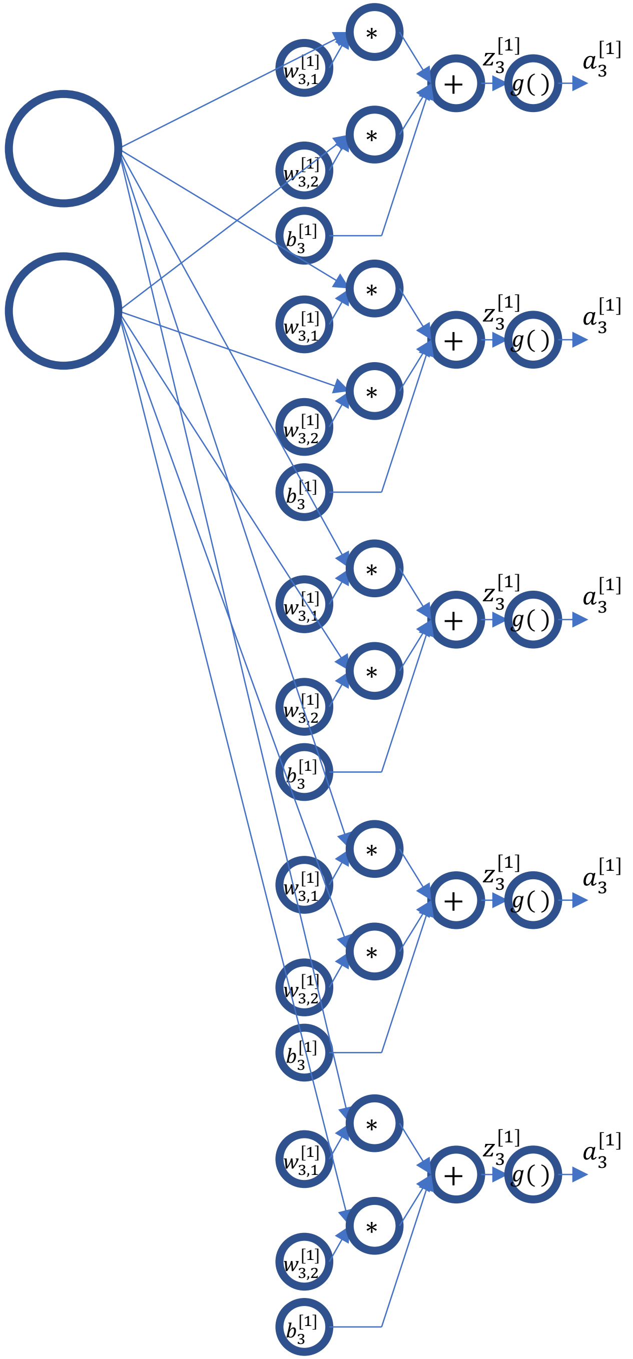
$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)



$$z_1^{[1]} = w_{1,1}^{[1]}\, a_1^{[0]} + w_{1,2}^{[1]} a_2^{[0]} + b_1^{[1]}$$

$$z_2^{[1]} = w_{2,1}^{[1]}\, a_1^{[0]} + w_{2,2}^{[1]} a_2^{[0]} + b_2^{[1]}$$

$$z_3^{[1]} = w_{3,1}^{[1]}\, a_1^{[0]} + w_{3,2}^{[1]} a_2^{[0]} + b_3^{[1]}$$

$$z_4^{[1]} = w_{4,1}^{[1]}\, a_1^{[0]} + w_{4,2}^{[1]} a_2^{[0]} + b_4^{[1]}$$

$$z_5^{[1]} = w_{5,1}^{[1]}\, a_1^{[0]} + w_{5,2}^{[1]} a_2^{[0]} + b_5^{[1]}$$

$$a_1^{[1]} = g\left(z_1^{[1]}\right)$$

$$a_2^{[1]} = g\left(z_2^{[1]}\right)$$

$$a_3^{[1]} = g\left(z_3^{[1]}\right)$$

$$a_4^{[1]} = g\left(z_4^{[1]}\right)$$

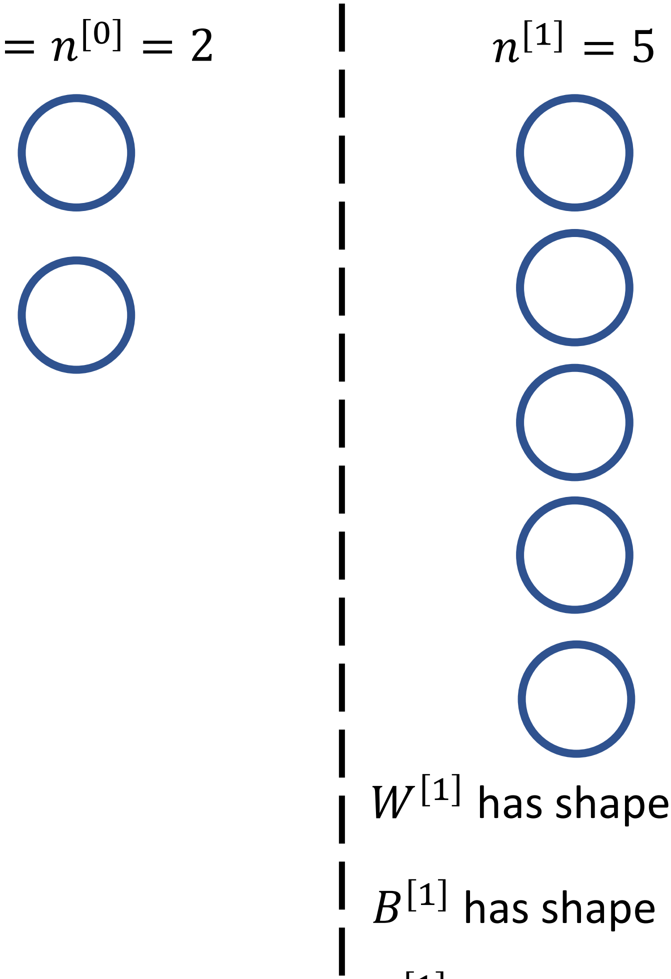
$$a_5^{[1]} = g\left(z_5^{[1]}\right)$$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

Since we know we can compute the outputs using the following vectorized operations, we can consider instead a compute graph that employs the corresponding vectorized operations

$$Z^{[1]} = matmul\left(W^{[1]}, A^{[0]}\right) + B^{[1]}$$

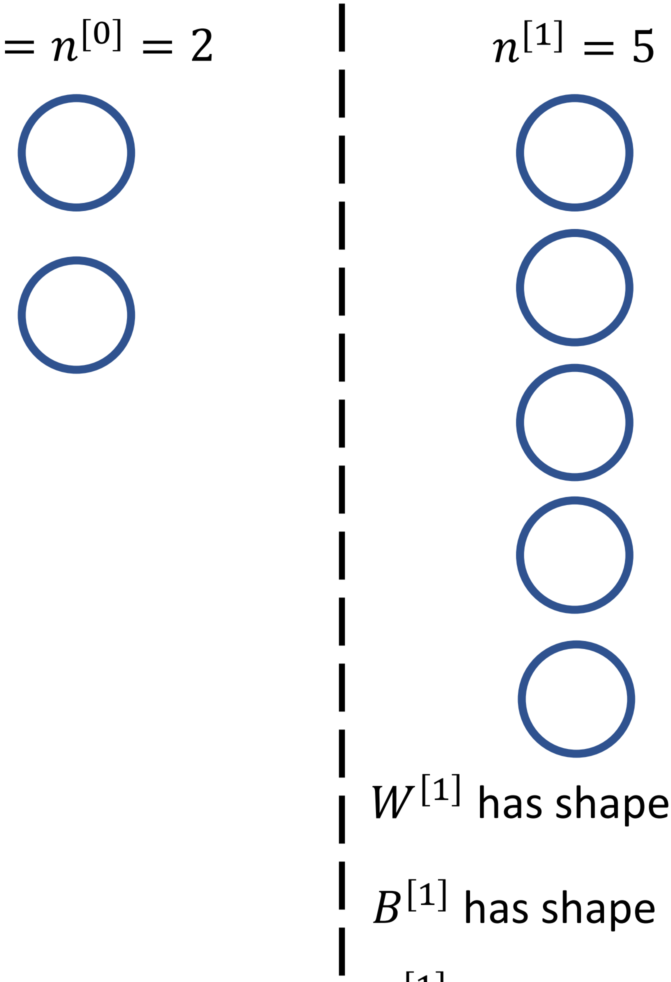
$$A^{[1]} = g\left(Z^{[1]}\right)$$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$W^{[1]}$ has shape (5, 2)

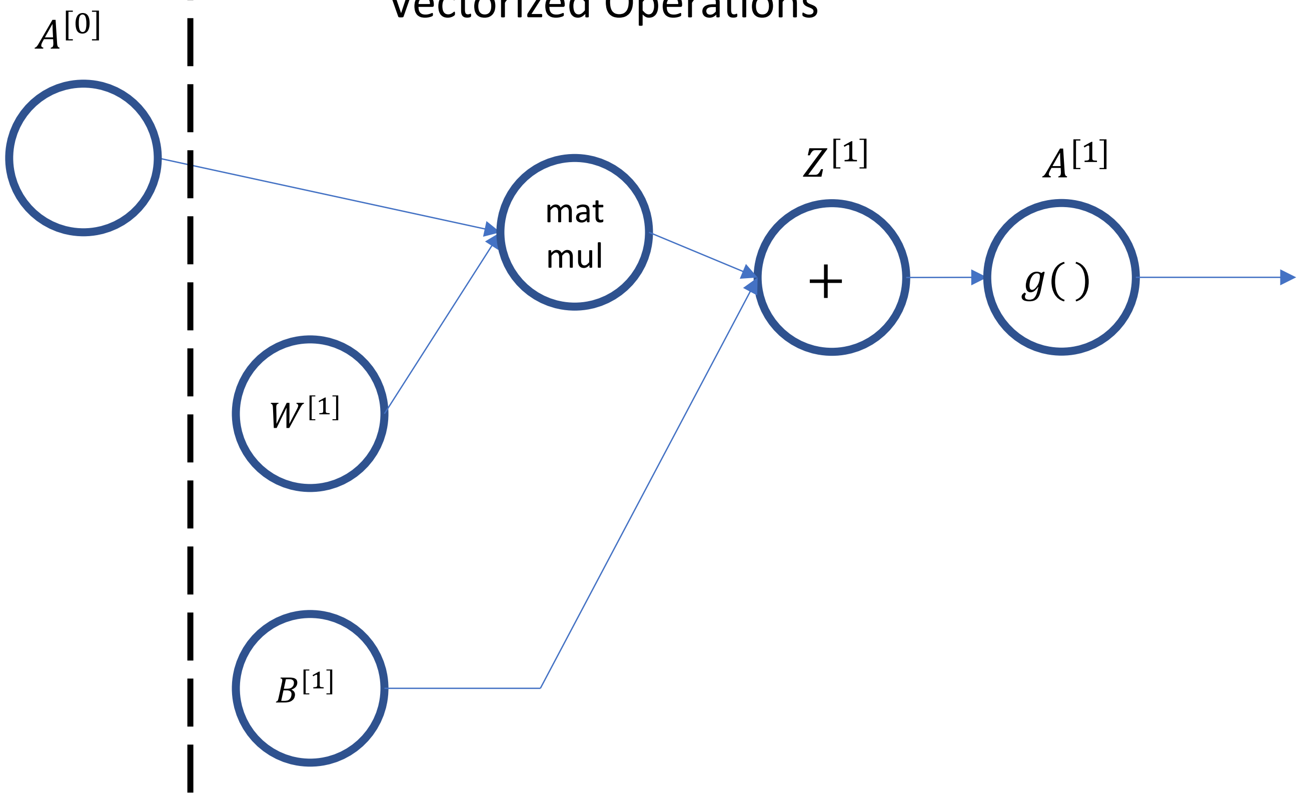
$B^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

Corresponding Compute Graph using Vectorized Operations



Since we know we can compute the outputs using the following vectorized operations, we can consider instead a compute graph that employs the corresponding vectorized operations

$$Z^{[1]} = matmul\left(W^{[1]}, A^{[0]}\right) + B^{[1]}$$

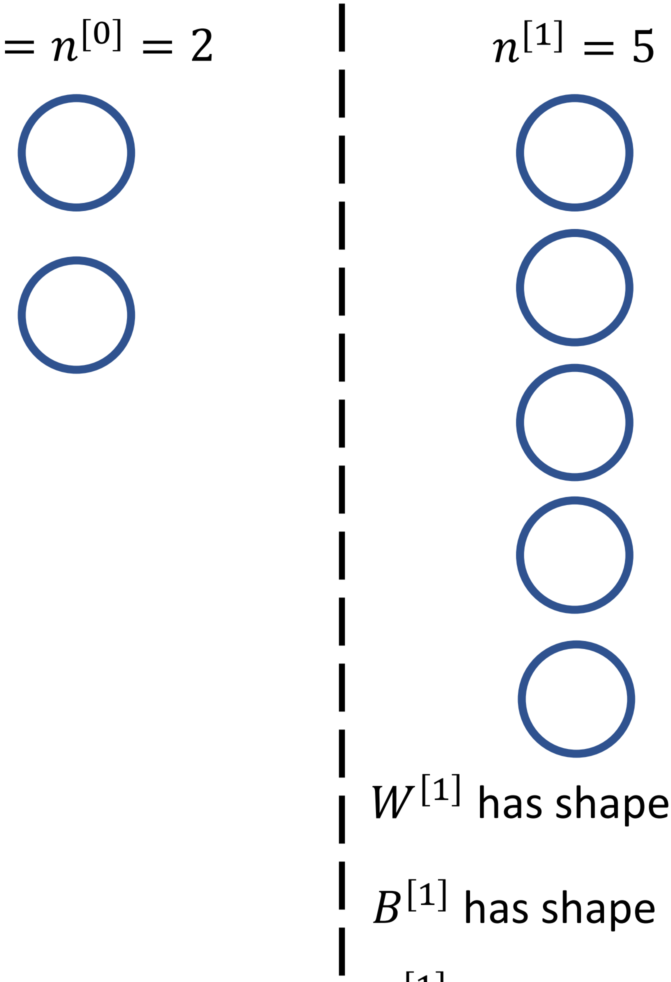
$$A^{[1]} = g\left(Z^{[1]}\right)$$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$W^{[1]}$ has shape (5, 2)

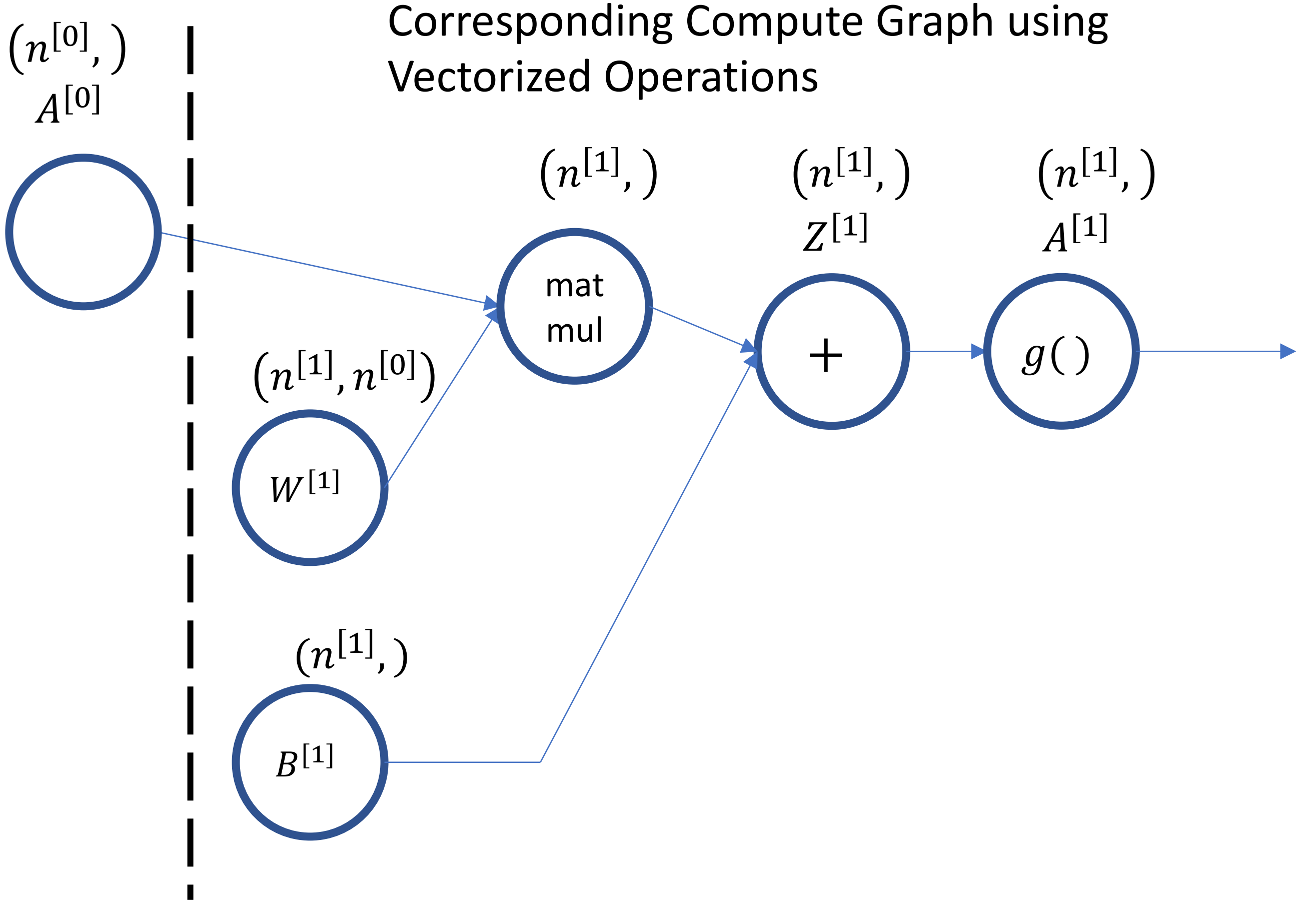
$B^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

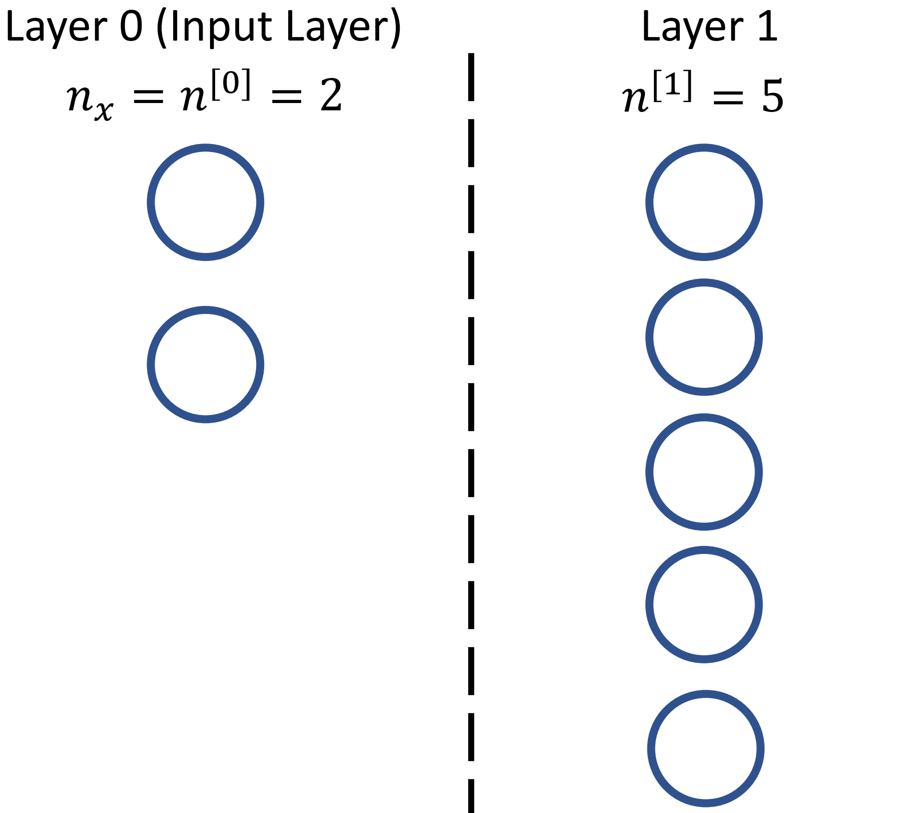
Corresponding Compute Graph using Vectorized Operations



Let’s now consider the shapes on the various compute graph nodes

$$Z^{[1]} = matmul\left(W^{[1]}, A^{[0]}\right) + B^{[1]}$$

$$A^{[1]} = g\left(Z^{[1]}\right)$$



$W^{[1]}$ has shape (5, 2)

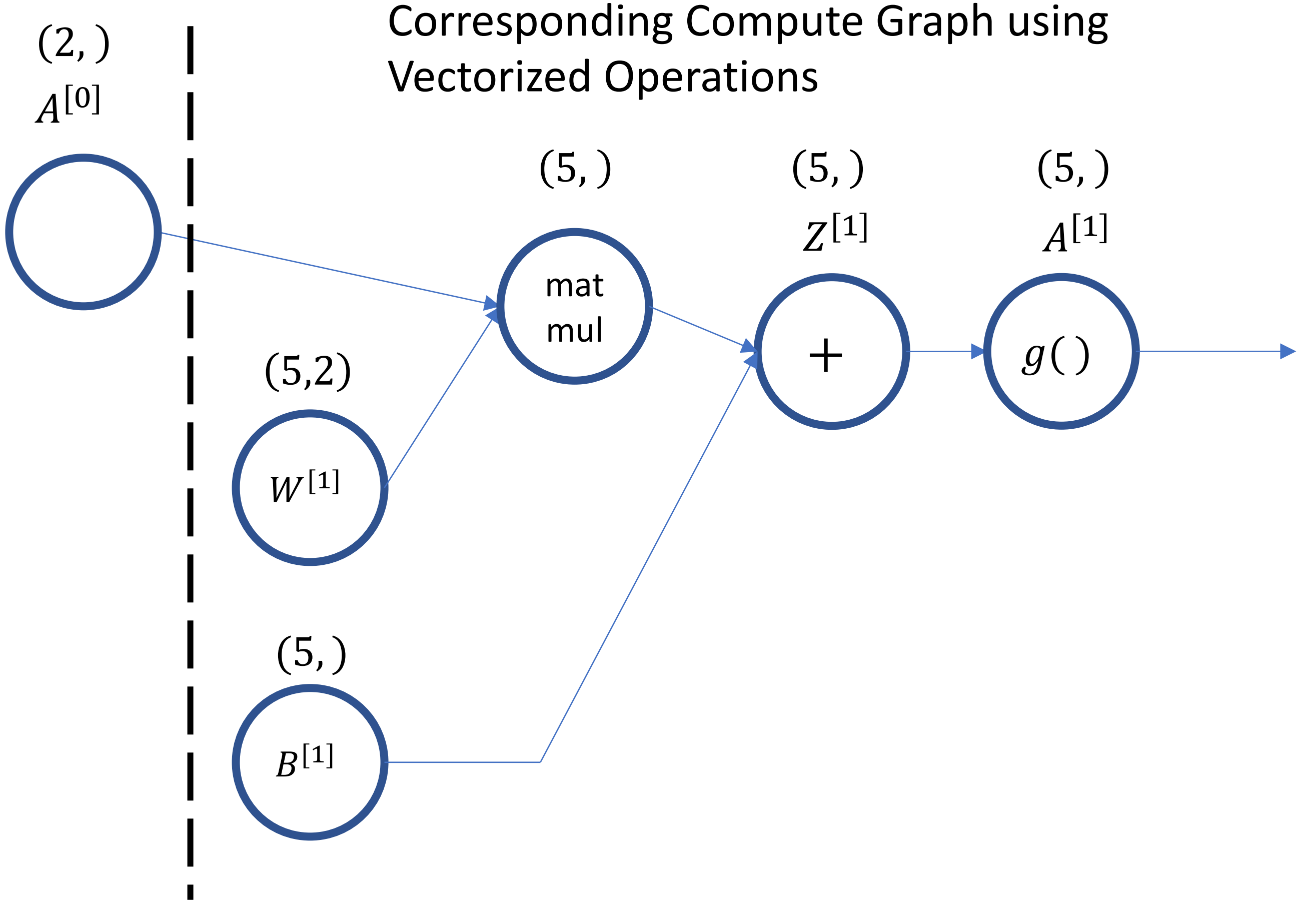
$B^{[1]}$ has shape (5,)

$A^{[0]} = X$ has shape (2,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

Corresponding Compute Graph using Vectorized Operations

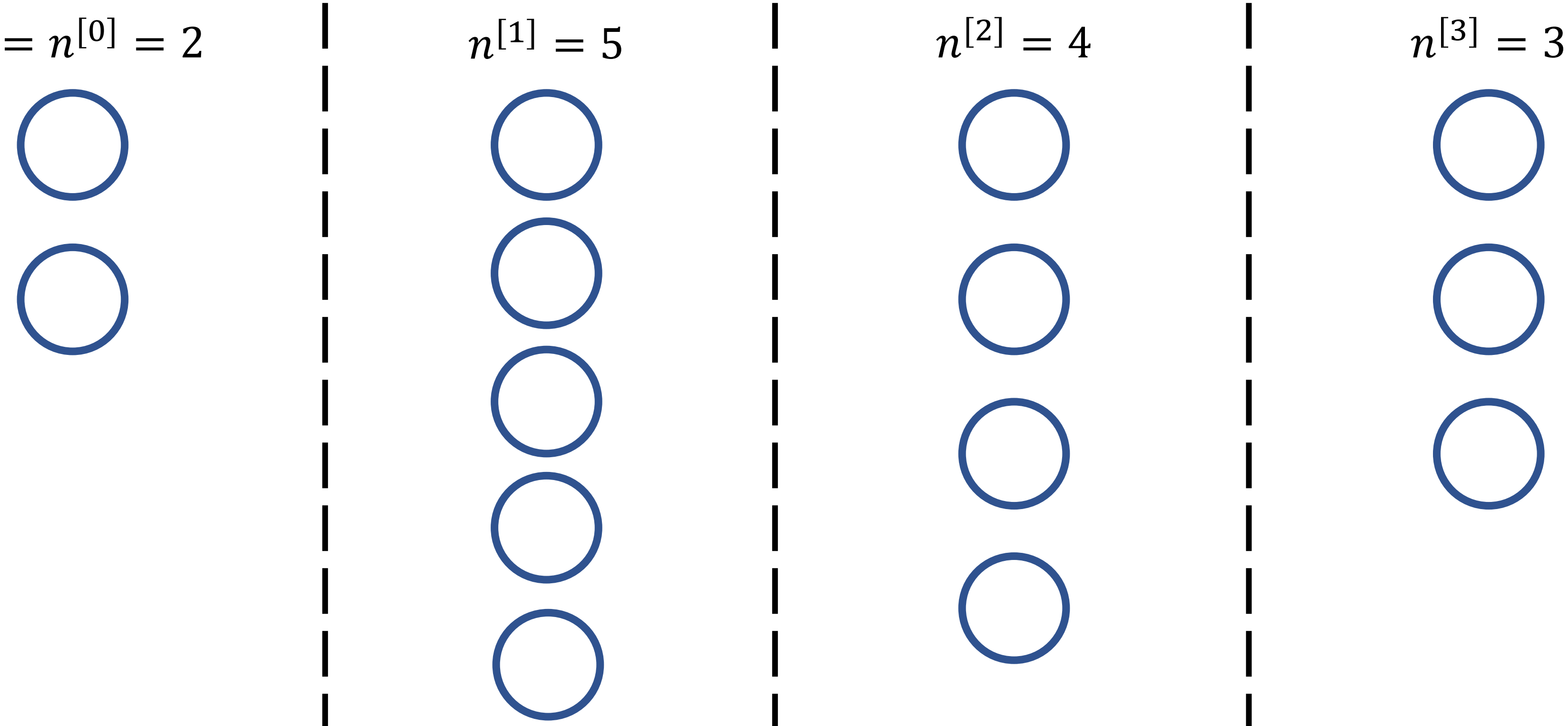


Let’s now consider the shapes on the various compute graph nodes

$$Z^{[1]} = matmul\left(W^{[1]}, A^{[0]}\right) + B^{[1]}$$

$$A^{[1]} = g\left(Z^{[1]}\right)$$

Layer 0 (Input Layer)	Layer 1	Layer 2	Layer 3 (Output Layer)
$n_x = n^{[0]} = 2$	$n^{[1]} = 5$	$n^{[2]} = 4$	$n^{[3]} = 3$



	$W^{[1]}$ has shape (5, 2)	$W^{[2]}$ has shape (4, 5)	$W^{[3]}$ has shape (3, 4)
	$B^{[1]}$ has shape (5,)	$B^{[2]}$ has shape (4,)	$B^{[3]}$ has shape (3,)
$A^{[0]} = X$ has shape (2,)	$A^{[1]}$ has shape (5,)	$A^{[2]}$ has shape (4,)	$\hat{Y} = A^{[3]}$ has shape (3,)
	$Z^{[1]}$ has shape (5,)	$Z^{[2]}$ has shape (4,)	$Z^{[3]}$ has shape (3,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

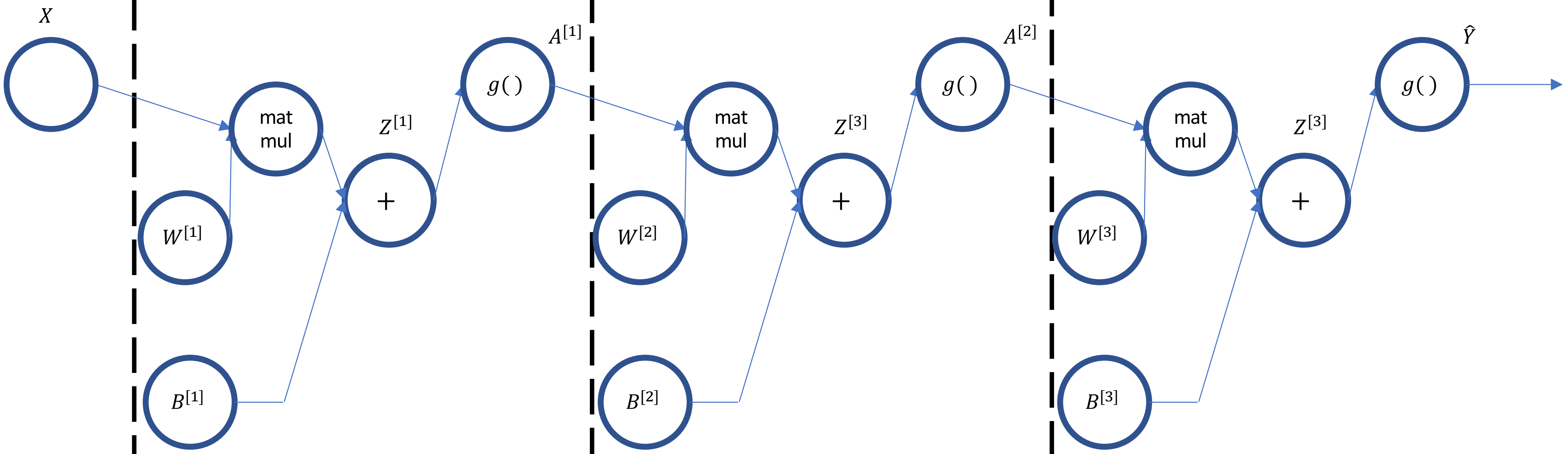
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

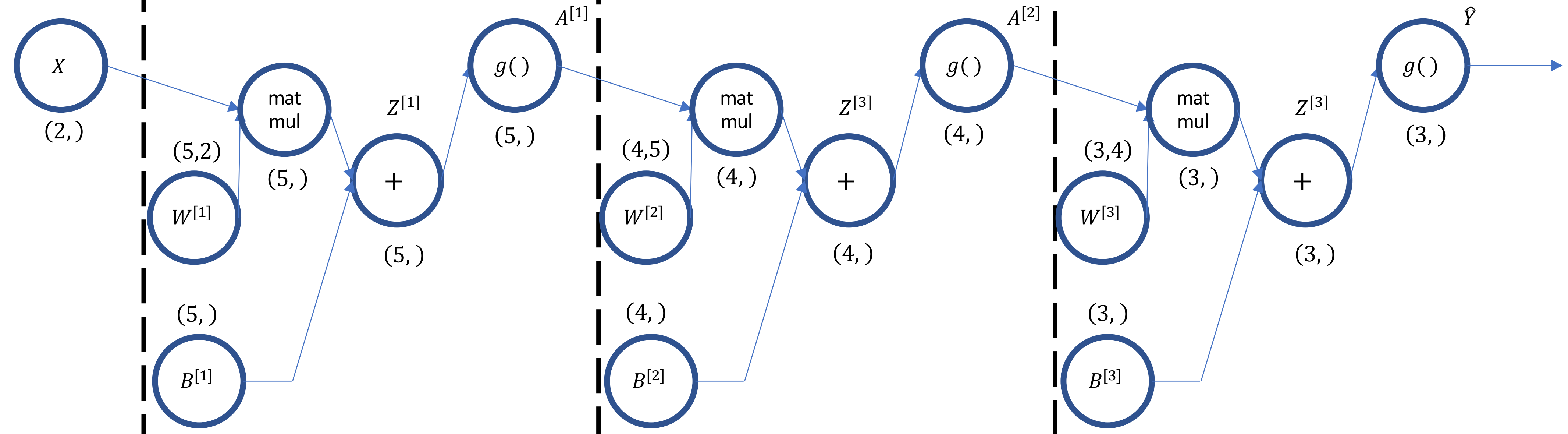
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

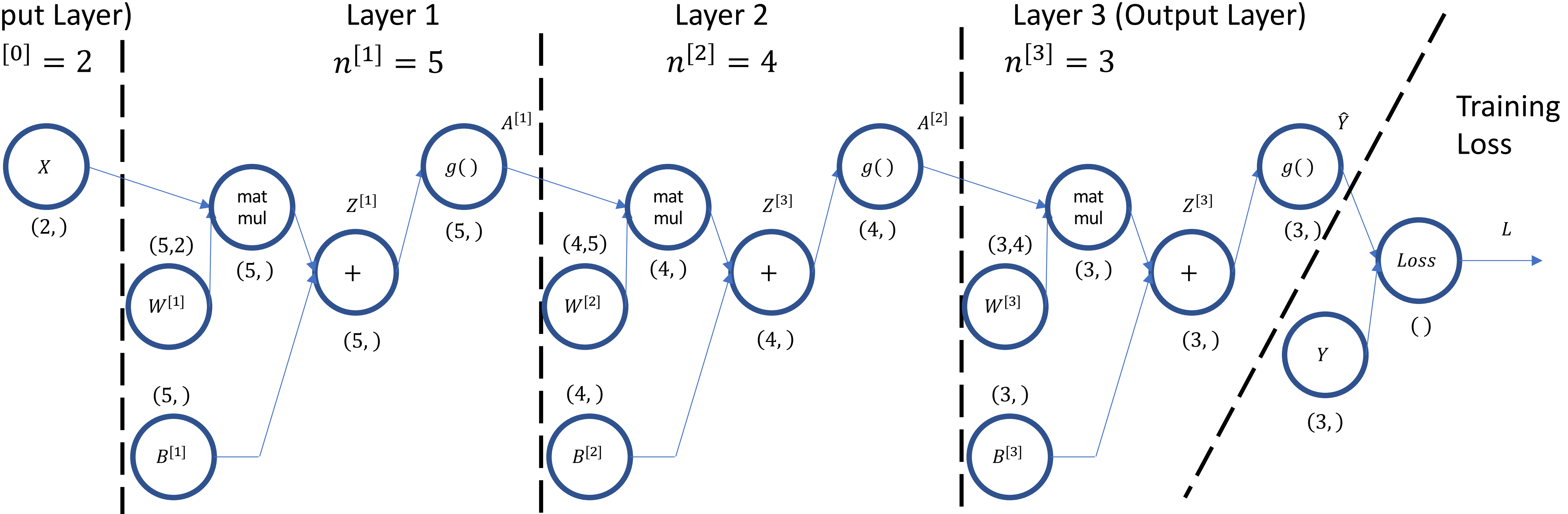
$Z^{[3]}$ has shape (3,)

Layer 0 (Input Layer) $n_x = n^{[0]} = 2$

Layer 1 $n^{[1]} = 5$

Layer 2 $n^{[2]} = 4$

Layer 3 (Output Layer) $n^{[3]} = 3$



$A^{[0]} = X$ has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

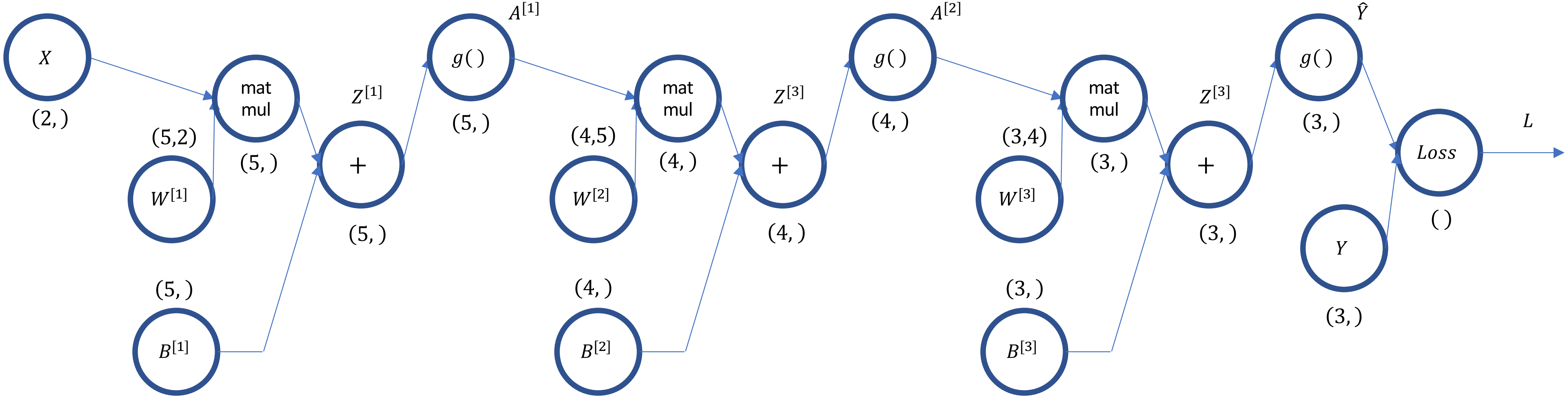
$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

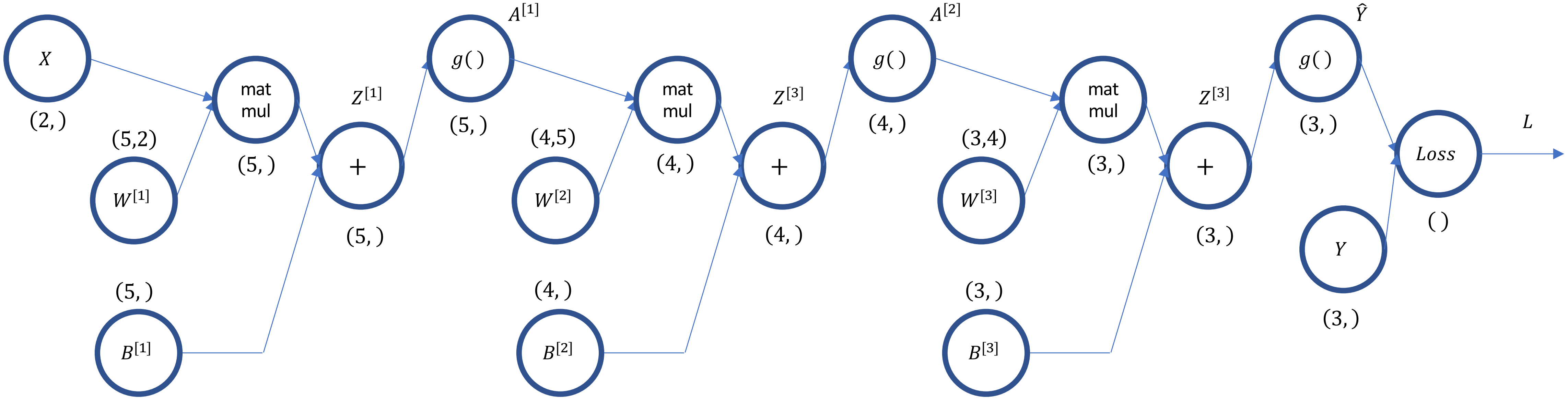
$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

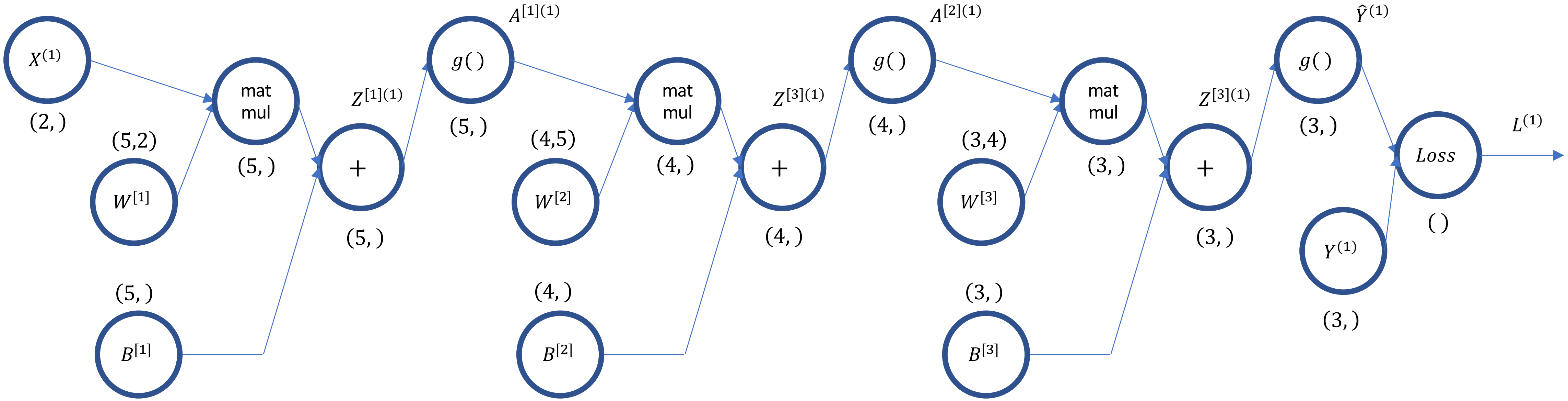

X
$(2,)$
mat mul
$(5,2)$
$W^{[1]}$
$(5,)$
$(5,)$
$B^{[1]}$
$Z^{[1]}$
$+$
$(5,)$
$g()$
$A^{[1]}$
$(5,)$
mat mul
$(4,5)$
$W^{[2]}$
$(4,)$
$(4,)$
$B^{[2]}$
$Z^{[3]}$
$+$
$(4,)$
$g()$
$A^{[2]}$
$(4,)$
mat mul
$(3,4)$
$W^{[3]}$
$(3,)$
$(3,)$
$B^{[3]}$
$Z^{[3]}$
$+$
$(3,)$
$g()$
$\hat{Y}$
$(3,)$
Y
$(3,)$
$Loss$
$()$
L

Consider two training sample, i.e. m=2



Consider two training sample, i.e. m=2

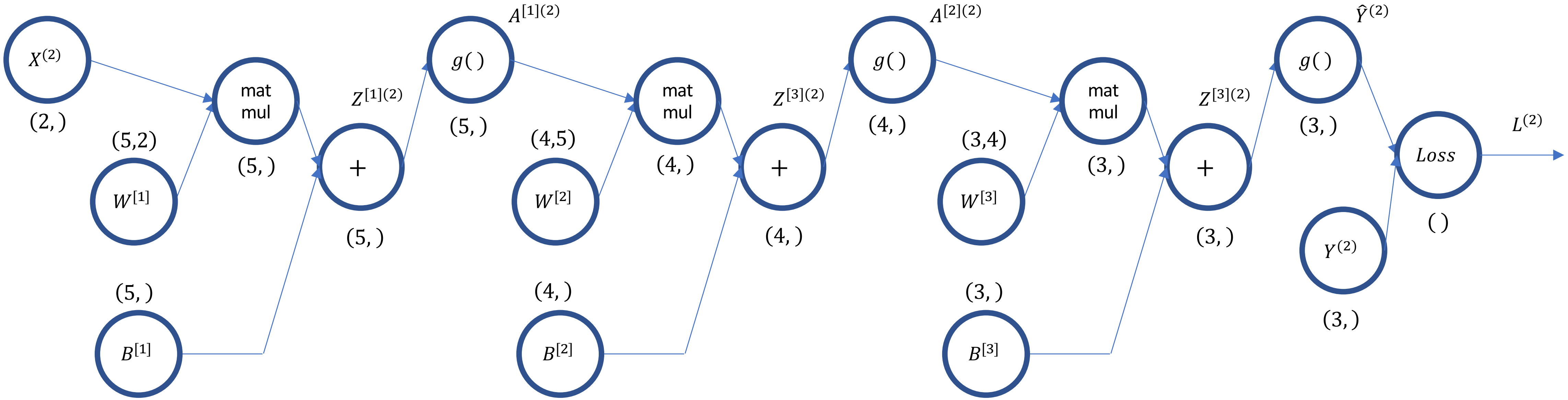
For sample 1, $X^{(1)}$



Brad Quinton, Scott Chin

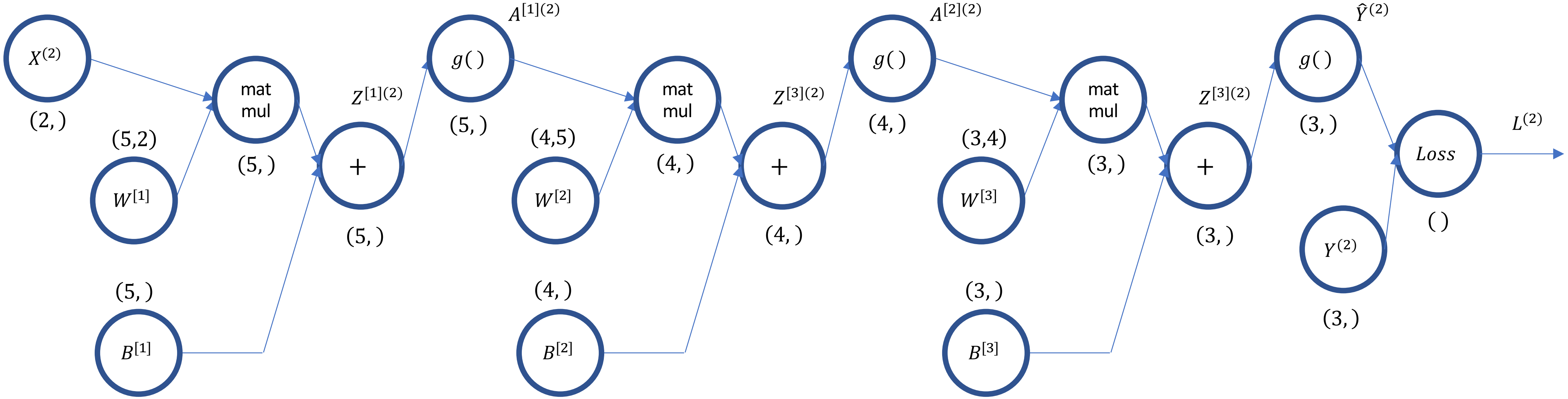
Consider two training sample, i.e. m=2

For sample 2, $X^{(2)}$



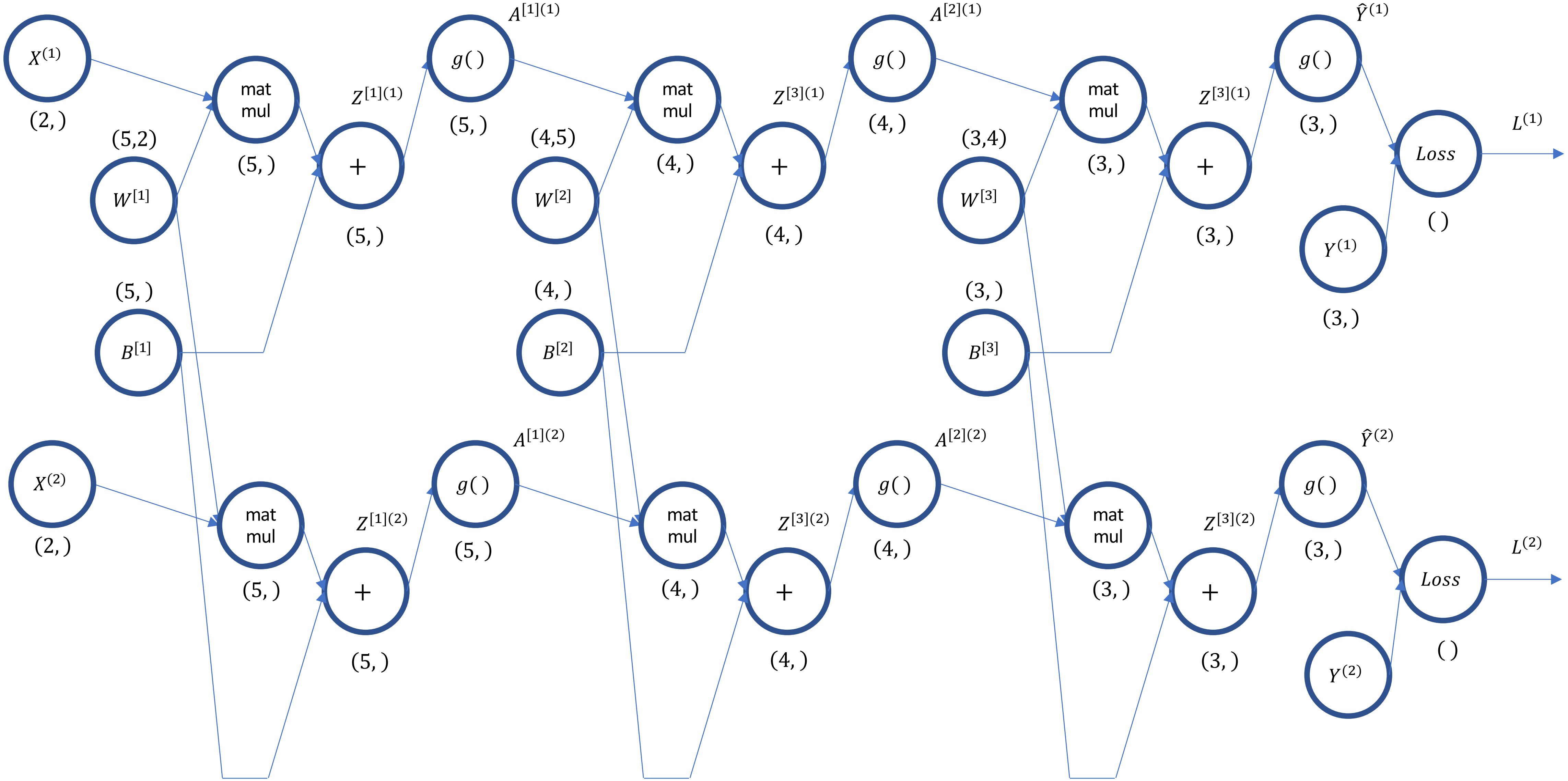
Consider two training sample, i.e. m=2

For sample 2, $X^{(1)}$



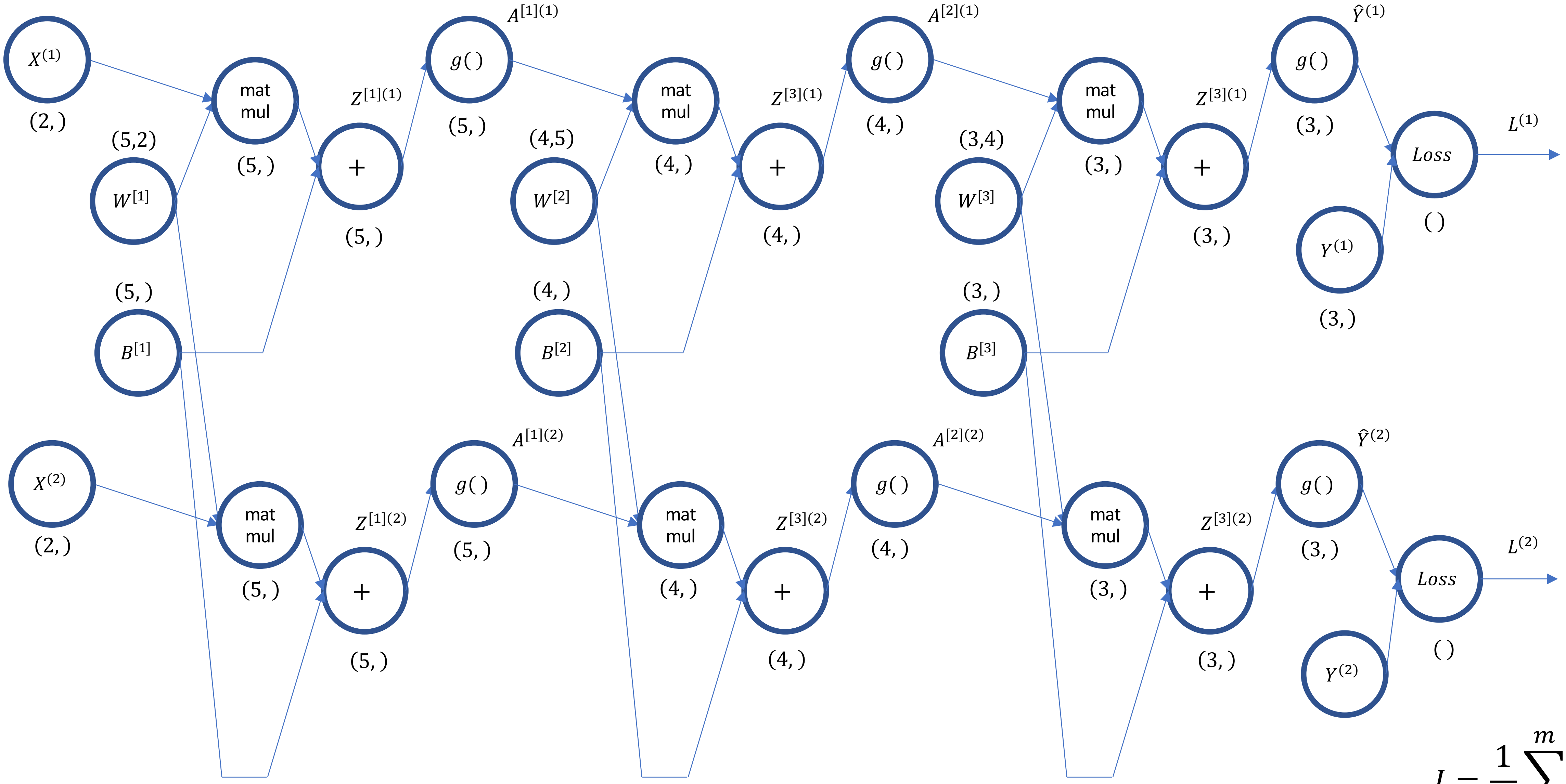
In practice, we don’t want to do this one at a time! Too slow!

Consider two training sample, i.e. m=2



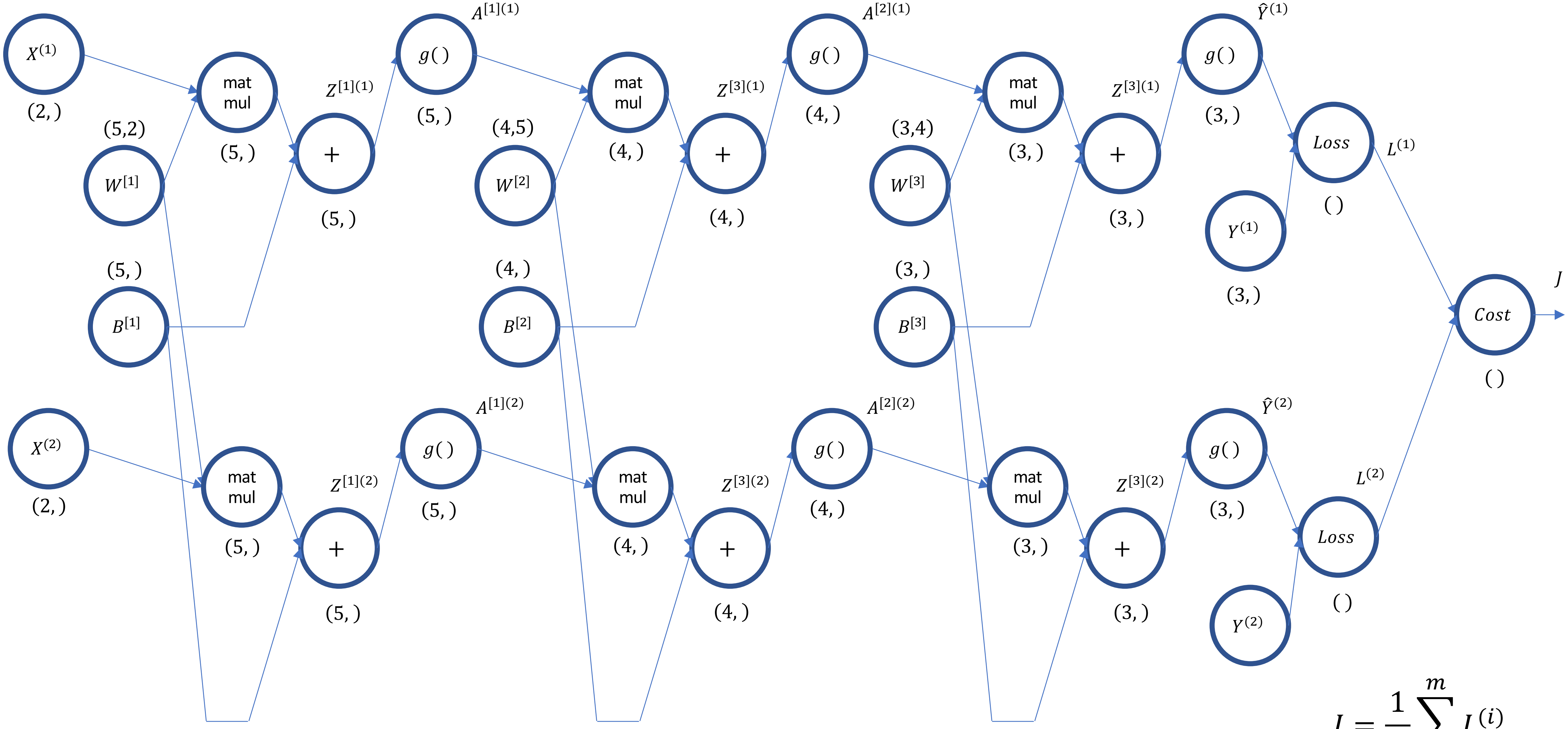
Brad Quinton, Scott Chin

Consider two training sample, i.e. m=2



$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

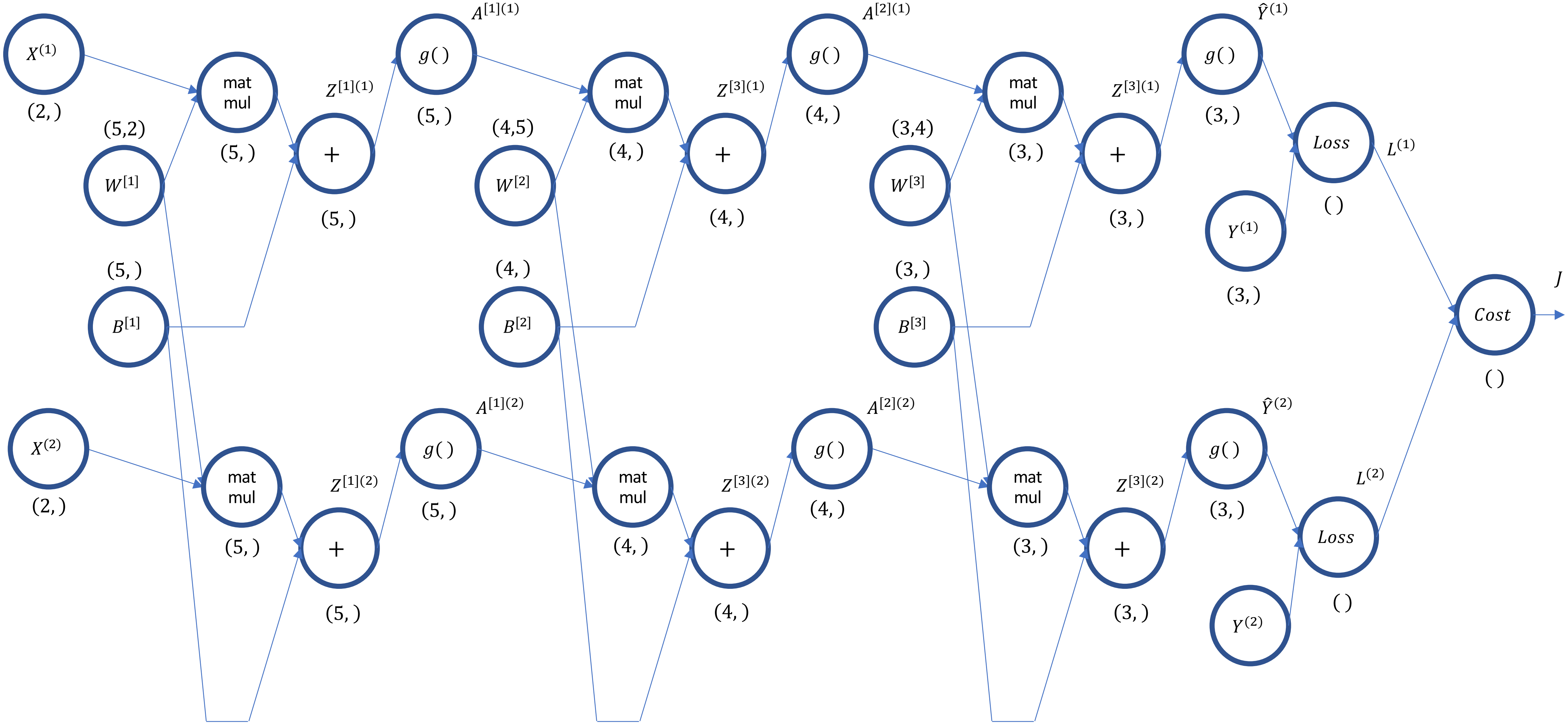
Consider two training sample, i.e. m=2



$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

Brad Quinton, Scott Chin

We can continue extending this graph for additional training samples
But we can do even more vectorization



Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

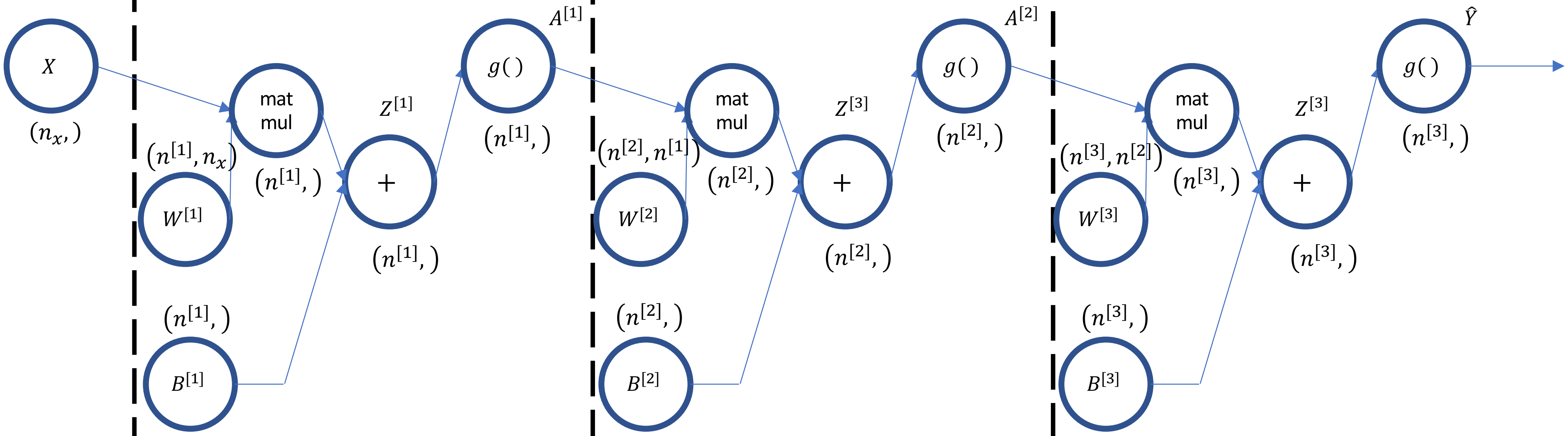
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

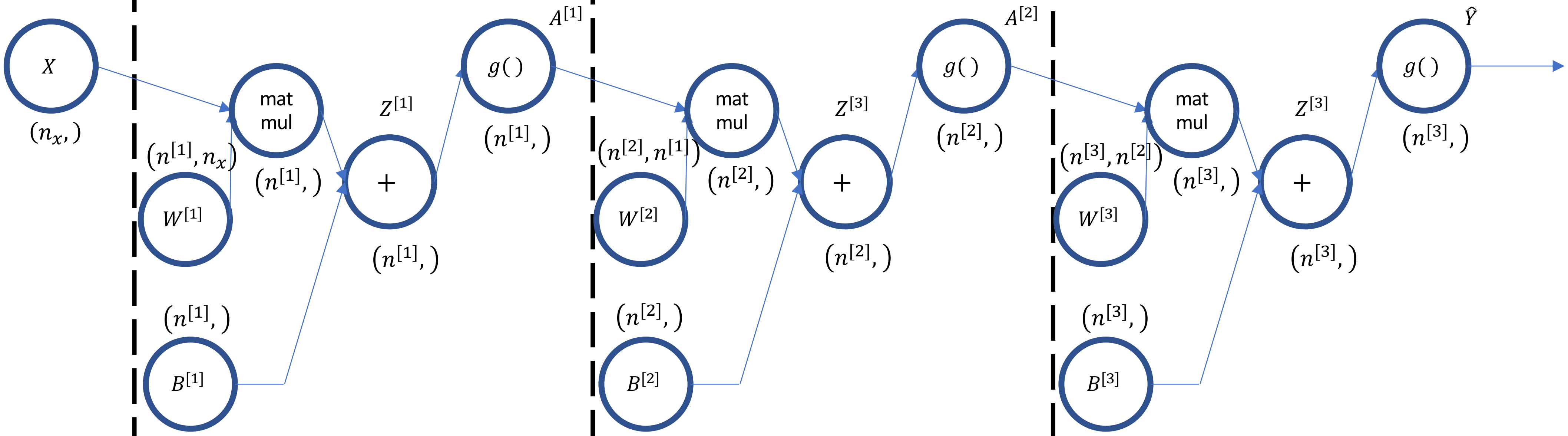
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

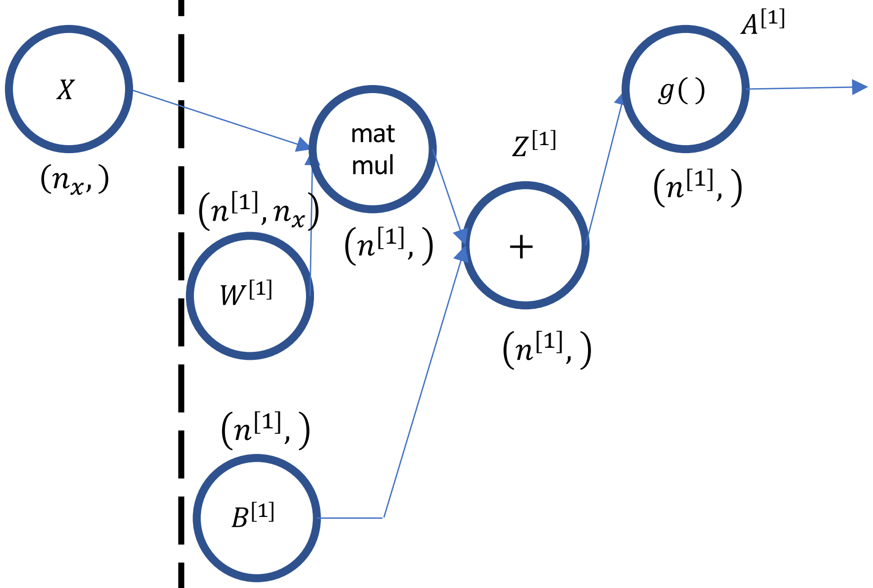
$Z^{[3]}$ has shape (3,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



Goal is to calculate:

$$z_1^{[1](1)} = w_{1,1}^{[1]}\, x_1^{(1)} + w_{1,2}^{[1]} x_2^{(1)} + b_1^{[1]} \qquad z_1^{[1](2)} = w_{1,1}^{[1]}\, x_1^{(2)} + w_{1,2}^{[1]} x_2^{(2)} + b_1^{[1]}$$

$$z_2^{[1](1)} = w_{2,1}^{[1]}\, x_1^{(1)} + w_{2,2}^{[1]} x_2^{(1)} + b_2^{[1]} \qquad z_2^{[1](2)} = w_{2,1}^{[1]}\, x_1^{(2)} + w_{2,2}^{[1]} x_2^{(2)} + b_2^{[1]}$$

$$z_3^{[1](1)} = w_{3,1}^{[1]}\, x_1^{(1)} + w_{3,2}^{[1]} x_2^{(1)} + b_3^{[1]} \qquad z_3^{[1](2)} = w_{3,1}^{[1]}\, x_1^{(2)} + w_{3,2}^{[1]} x_2^{(2)} + b_3^{[1]}$$

$$z_4^{[1](1)} = w_{4,1}^{[1]}\, x_1^{(1)} + w_{4,2}^{[1]} x_2^{(1)} + b_4^{[1]} \qquad z_4^{[1](2)} = w_{4,1}^{[1]}\, x_1^{(2)} + w_{4,2}^{[1]} x_2^{(2)} + b_4^{[1]}$$

$$z_5^{[1](1)} = w_{5,1}^{[1]}\, x_1^{(1)} + w_{5,2}^{[1]} x_2^{(1)} + b_5^{[1]} \qquad z_5^{[1](2)} = w_{5,1}^{[1]}\, x_1^{(2)} + w_{5,2}^{[1]} x_2^{(2)} + b_5^{[1]}$$

$A^{[0]} = X$
has shape (2,)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

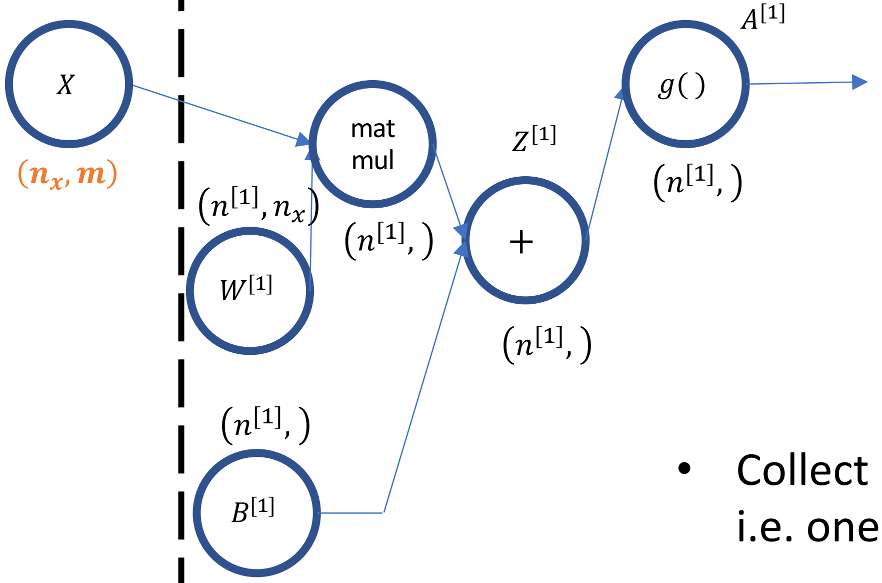
$A^{[1]}$ has shape (5,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



Goal is to calculate:

$$
\begin{array}{ll}
z_1^{[1](1)} = w_{1,1}^{[1]}\, x_1^{(1)} + w_{1,2}^{[1]} x_2^{(1)} + b_1^{[1]} & z_1^{[1](2)} = w_{1,1}^{[1]}\, x_1^{(2)} + w_{1,2}^{[1]} x_2^{(2)} + b_1^{[1]} \\
z_2^{[1](1)} = w_{2,1}^{[1]}\, x_1^{(1)} + w_{2,2}^{[1]} x_2^{(1)} + b_2^{[1]} & z_2^{[1](2)} = w_{2,1}^{[1]}\, x_1^{(2)} + w_{2,2}^{[1]} x_2^{(2)} + b_2^{[1]} \\
z_3^{[1](1)} = w_{3,1}^{[1]}\, x_1^{(1)} + w_{3,2}^{[1]} x_2^{(1)} + b_3^{[1]} & z_3^{[1](2)} = w_{3,1}^{[1]}\, x_1^{(2)} + w_{3,2}^{[1]} x_2^{(2)} + b_3^{[1]} \\
z_4^{[1](1)} = w_{4,1}^{[1]}\, x_1^{(1)} + w_{4,2}^{[1]} x_2^{(1)} + b_4^{[1]} & z_4^{[1](2)} = w_{4,1}^{[1]}\, x_1^{(2)} + w_{4,2}^{[1]} x_2^{(2)} + b_4^{[1]} \\
z_5^{[1](1)} = w_{5,1}^{[1]}\, x_1^{(1)} + w_{5,2}^{[1]} x_2^{(1)} + b_5^{[1]} & z_5^{[1](2)} = w_{5,1}^{[1]}\, x_1^{(2)} + w_{5,2}^{[1]} x_2^{(2)} + b_5^{[1]}
\end{array}
$$

- Collect all inputs into matrix X so now its shape goes from$(n_x,)$ to (n_x, m) i.e. one column for each sample

$A^{[0]} = X$

has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

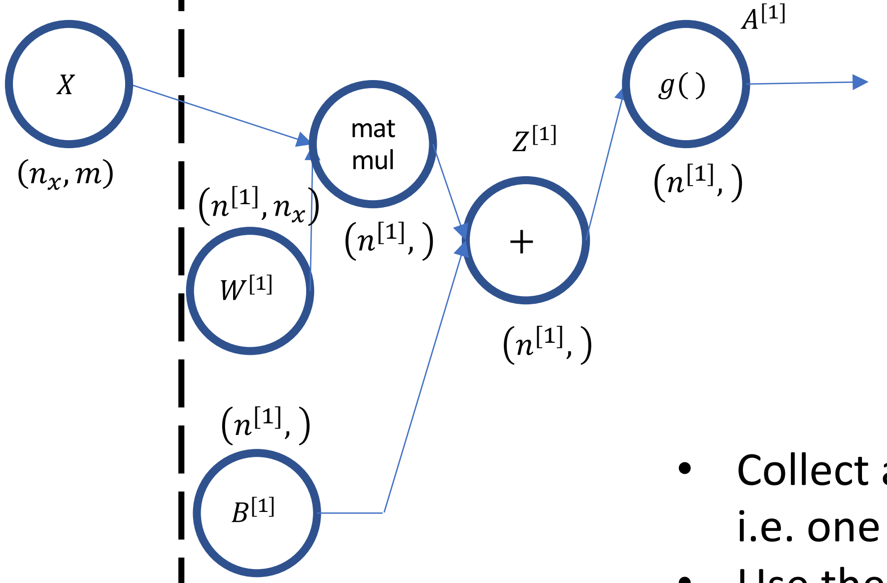
$A^{[1]}$ has shape (5,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



Goal is to calculate:

$$z_1^{[1](1)} = w_{1,1}^{[1]} x_1^{(1)} + w_{1,2}^{[1]} x_2^{(1)} + b_1^{[1]} \qquad z_1^{[1](2)} = w_{1,1}^{[1]} x_1^{(2)} + w_{1,2}^{[1]} x_2^{(2)} + b_1^{[1]}$$

$$z_2^{[1](1)} = w_{2,1}^{[1]} x_1^{(1)} + w_{2,2}^{[1]} x_2^{(1)} + b_2^{[1]} \qquad z_2^{[1](2)} = w_{2,1}^{[1]} x_1^{(2)} + w_{2,2}^{[1]} x_2^{(2)} + b_2^{[1]}$$

$$z_3^{[1](1)} = w_{3,1}^{[1]} x_1^{(1)} + w_{3,2}^{[1]} x_2^{(1)} + b_3^{[1]} \qquad z_3^{[1](2)} = w_{3,1}^{[1]} x_1^{(2)} + w_{3,2}^{[1]} x_2^{(2)} + b_3^{[1]}$$

$$z_4^{[1](1)} = w_{4,1}^{[1]} x_1^{(1)} + w_{4,2}^{[1]} x_2^{(1)} + b_4^{[1]} \qquad z_4^{[1](2)} = w_{4,1}^{[1]} x_1^{(2)} + w_{4,2}^{[1]} x_2^{(2)} + b_4^{[1]}$$

$$z_5^{[1](1)} = w_{5,1}^{[1]} x_1^{(1)} + w_{5,2}^{[1]} x_2^{(1)} + b_5^{[1]} \qquad z_5^{[1](2)} = w_{5,1}^{[1]} x_1^{(2)} + w_{5,2}^{[1]} x_2^{(2)} + b_5^{[1]}$$

- Collect all inputs into matrix X so now its shape goes from$(n_x,)$ to (n_x, m) i.e. one column for each sample
- Use the same vectorized operation as before

$$Z^{[1]} = matmul\left(W^{[1]}, X\right) + B^{[1]}$$

$A^{[0]} = X$
has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

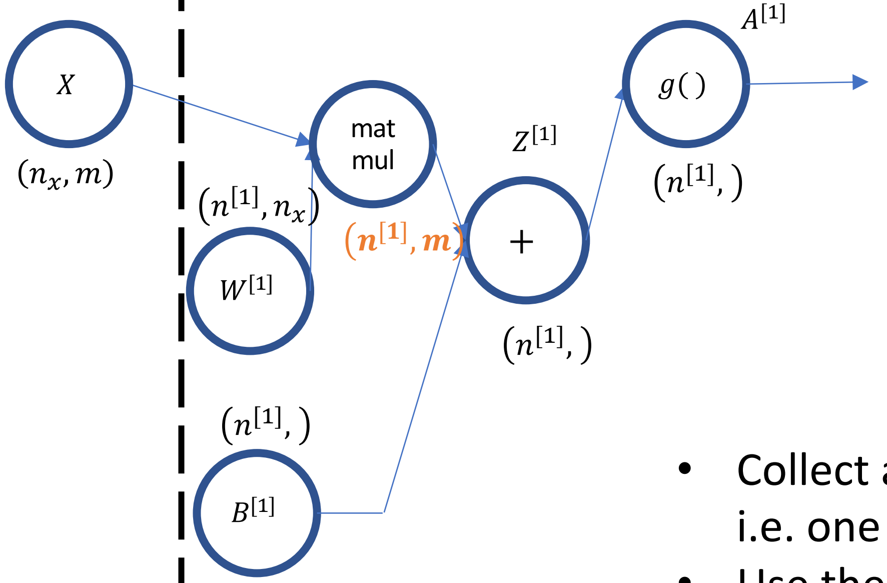
$A^{[1]}$ has shape (5,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$A^{[0]} = X$
has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

Goal is to calculate:

$$\begin{aligned}
z_1^{[1](1)} &= w_{1,1}^{[1]}\, x_1^{(1)} + w_{1,2}^{[1]} x_2^{(1)} + b_1^{[1]} & \quad z_1^{[1](2)} &= w_{1,1}^{[1]}\, x_1^{(2)} + w_{1,2}^{[1]} x_2^{(2)} + b_1^{[1]} \\
z_2^{[1](1)} &= w_{2,1}^{[1]}\, x_1^{(1)} + w_{2,2}^{[1]} x_2^{(1)} + b_2^{[1]} & \quad z_2^{[1](2)} &= w_{2,1}^{[1]}\, x_1^{(2)} + w_{2,2}^{[1]} x_2^{(2)} + b_2^{[1]} \\
z_3^{[1](1)} &= w_{3,1}^{[1]}\, x_1^{(1)} + w_{3,2}^{[1]} x_2^{(1)} + b_3^{[1]} & \quad z_3^{[1](2)} &= w_{3,1}^{[1]}\, x_1^{(2)} + w_{3,2}^{[1]} x_2^{(2)} + b_3^{[1]} \\
z_4^{[1](1)} &= w_{4,1}^{[1]}\, x_1^{(1)} + w_{4,2}^{[1]} x_2^{(1)} + b_4^{[1]} & \quad z_4^{[1](2)} &= w_{4,1}^{[1]}\, x_1^{(2)} + w_{4,2}^{[1]} x_2^{(2)} + b_4^{[1]} \\
z_5^{[1](1)} &= w_{5,1}^{[1]}\, x_1^{(1)} + w_{5,2}^{[1]} x_2^{(1)} + b_5^{[1]} & \quad z_5^{[1](2)} &= w_{5,1}^{[1]}\, x_1^{(2)} + w_{5,2}^{[1]} x_2^{(2)} + b_5^{[1]}
\end{aligned}$$

- Collect all inputs into matrix X so now its shape goes from$(n_x,)$ to (n_x, m) i.e. one column for each sample
- Use the same vectorized operation as before

$$Z^{[1]} = matmul\left(W^{[1]}, X\right) + B^{[1]}$$

$$= \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} \\ w_{3,1}^{[1]} & w_{3,2}^{[1]} \\ w_{4,1}^{[1]} & w_{4,2}^{[1]} \\ w_{5,1}^{[1]} & w_{5,2}^{[1]} \end{bmatrix} \begin{bmatrix} x_1^{(1)} & x_1^{(2)} \\ x_2^{(1)} & x_2^{(2)} \end{bmatrix} + \begin{bmatrix} b_1^{[1]} & b_1^{[1]} \\ b_2^{[1]} & b_2^{[1]} \\ b_3^{[1]} & b_3^{[1]} \\ b_4^{[1]} & b_4^{[1]} \\ b_5^{[1]} & b_5^{[1]} \end{bmatrix}$$

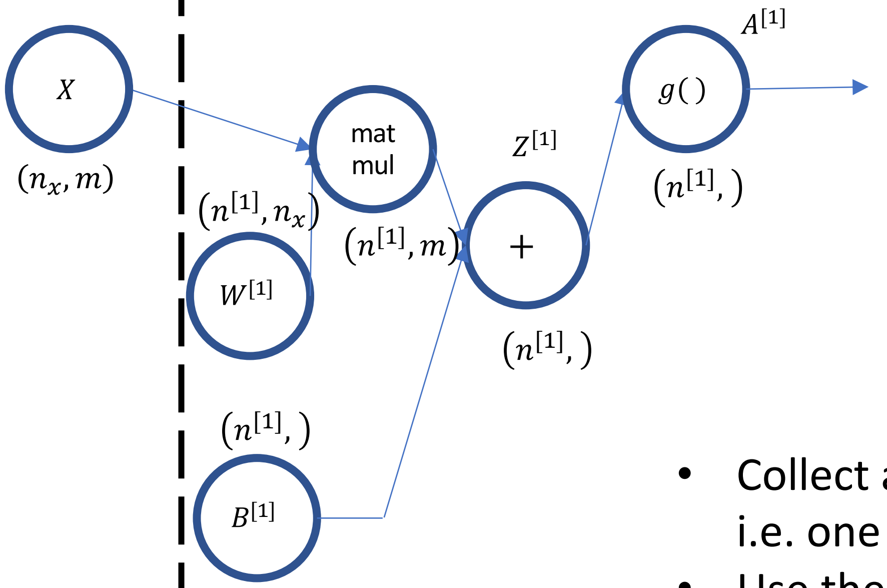
- Shape of $W^{[1]}$ doesn't change
- The result of the matrix multiply is shape $\left(n^{[1]}, m\right)$ (i.e. one column per sample)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



Goal is to calculate:

$$z_1^{[1](1)} = w_{1,1}^{[1]}\, x_1^{(1)} + w_{1,2}^{[1]} x_2^{(1)} + b_1^{[1]} \qquad z_1^{[1](2)} = w_{1,1}^{[1]}\, x_1^{(2)} + w_{1,2}^{[1]} x_2^{(2)} + b_1^{[1]}$$

$$z_2^{[1](1)} = w_{2,1}^{[1]}\, x_1^{(1)} + w_{2,2}^{[1]} x_2^{(1)} + b_2^{[1]} \qquad z_2^{[1](2)} = w_{2,1}^{[1]}\, x_1^{(2)} + w_{2,2}^{[1]} x_2^{(2)} + b_2^{[1]}$$

$$z_3^{[1](1)} = w_{3,1}^{[1]}\, x_1^{(1)} + w_{3,2}^{[1]} x_2^{(1)} + b_3^{[1]} \qquad z_3^{[1](2)} = w_{3,1}^{[1]}\, x_1^{(2)} + w_{3,2}^{[1]} x_2^{(2)} + b_3^{[1]}$$

$$z_4^{[1](1)} = w_{4,1}^{[1]}\, x_1^{(1)} + w_{4,2}^{[1]} x_2^{(1)} + b_4^{[1]} \qquad z_4^{[1](2)} = w_{4,1}^{[1]}\, x_1^{(2)} + w_{4,2}^{[1]} x_2^{(2)} + b_4^{[1]}$$

$$z_5^{[1](1)} = w_{5,1}^{[1]}\, x_1^{(1)} + w_{5,2}^{[1]} x_2^{(1)} + b_5^{[1]} \qquad z_5^{[1](2)} = w_{5,1}^{[1]}\, x_1^{(2)} + w_{5,2}^{[1]} x_2^{(2)} + b_5^{[1]}$$

- Collect all inputs into matrix X so now its shape goes from$(n_x,)$ to (n_x, m) i.e. one column for each sample
- Use the same vectorized operation as before

$A^{[0]} = X$ has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$$Z^{[1]} = matmul\left(W^{[1]}, X\right) + B^{[1]}$$

$$= \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} \\ w_{3,1}^{[1]} & w_{3,2}^{[1]} \\ w_{4,1}^{[1]} & w_{4,2}^{[1]} \\ w_{5,1}^{[1]} & w_{5,2}^{[1]} \end{bmatrix} \begin{bmatrix} x_1^{(1)} & x_1^{(2)} \\ x_2^{(1)} & x_2^{(2)} \end{bmatrix} + \begin{bmatrix} b_1^{[1]} & b_1^{[1]} \\ b_2^{[1]} & b_2^{[1]} \\ b_3^{[1]} & b_3^{[1]} \\ b_4^{[1]} & b_4^{[1]} \\ b_5^{[1]} & b_5^{[1]} \end{bmatrix}$$

- Shape of $W^{[1]}$ doesn't change
- The result of the matrix multiply is shape $\left(n^{[1]}, m\right)$ (i.e. one column per sample)
- Conceptionally, shape of $B^{[1]}$, doesn't change, but to do the math, we need to stack copies of $B^{[1]}$. In practice, this is done via broadcasting

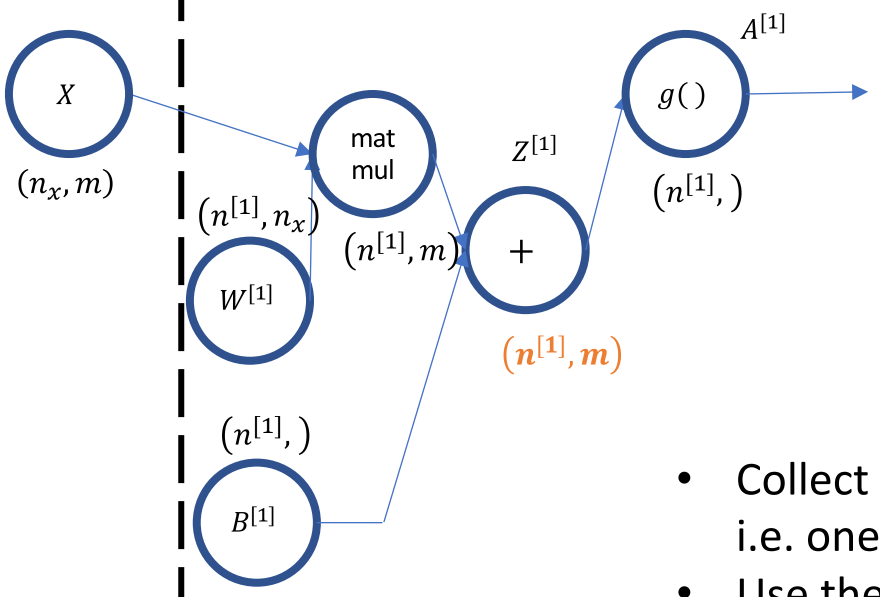
```
# In NumPy
Z1 = np.matmul(W1, A0) + B1
```

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$A^{[0]} = X$
has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5, 2)

$A^{[1]}$ has shape (5,)

Goal is to calculate:

$$
\begin{aligned}
z_1^{[1](1)} &= w_{1,1}^{[1]}\, x_1^{(1)} + w_{1,2}^{[1]} x_2^{(1)} + b_1^{[1]} & \quad z_1^{[1](2)} &= w_{1,1}^{[1]}\, x_1^{(2)} + w_{1,2}^{[1]} x_2^{(2)} + b_1^{[1]} \\
z_2^{[1](1)} &= w_{2,1}^{[1]}\, x_1^{(1)} + w_{2,2}^{[1]} x_2^{(1)} + b_2^{[1]} & \quad z_2^{[1](2)} &= w_{2,1}^{[1]}\, x_1^{(2)} + w_{2,2}^{[1]} x_2^{(2)} + b_2^{[1]} \\
z_3^{[1](1)} &= w_{3,1}^{[1]}\, x_1^{(1)} + w_{3,2}^{[1]} x_2^{(1)} + b_3^{[1]} & \quad z_3^{[1](2)} &= w_{3,1}^{[1]}\, x_1^{(2)} + w_{3,2}^{[1]} x_2^{(2)} + b_3^{[1]} \\
z_4^{[1](1)} &= w_{4,1}^{[1]}\, x_1^{(1)} + w_{4,2}^{[1]} x_2^{(1)} + b_4^{[1]} & \quad z_4^{[1](2)} &= w_{4,1}^{[1]}\, x_1^{(2)} + w_{4,2}^{[1]} x_2^{(2)} + b_4^{[1]} \\
z_5^{[1](1)} &= w_{5,1}^{[1]}\, x_1^{(1)} + w_{5,2}^{[1]} x_2^{(1)} + b_5^{[1]} & \quad z_5^{[1](2)} &= w_{5,1}^{[1]}\, x_1^{(2)} + w_{5,2}^{[1]} x_2^{(2)} + b_5^{[1]}
\end{aligned}
$$

- Collect all inputs into matrix X so now its shape goes from$(n_x,)$ to (n_x, m) i.e. one column for each sample
- Use the same vectorized operations as before

$$Z^{[1]} = matmul\left(W^{[1]}, X\right) + B^{[1]}$$

$$
= \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} \\ w_{3,1}^{[1]} & w_{3,2}^{[1]} \\ w_{4,1}^{[1]} & w_{4,2}^{[1]} \\ w_{5,1}^{[1]} & w_{5,2}^{[1]} \end{bmatrix}
\begin{bmatrix} x_1^{(1)} & x_1^{(2)} \\ x_2^{(1)} & x_2^{(2)} \end{bmatrix}
+ \begin{bmatrix} b_1^{[1]} & b_1^{[1]} \\ b_2^{[1]} & b_2^{[1]} \\ b_3^{[1]} & b_3^{[1]} \\ b_4^{[1]} & b_4^{[1]} \\ b_5^{[1]} & b_5^{[1]} \end{bmatrix}
= \begin{bmatrix} z_1^{[1](1)} & z_1^{[1](2)} \\ z_2^{[1](1)} & z_2^{[1](2)} \\ z_3^{[1](1)} & z_3^{[1](2)} \\ z_4^{[1](1)} & z_4^{[1](2)} \\ z_5^{[1](1)} & z_5^{[1](2)} \end{bmatrix}
$$

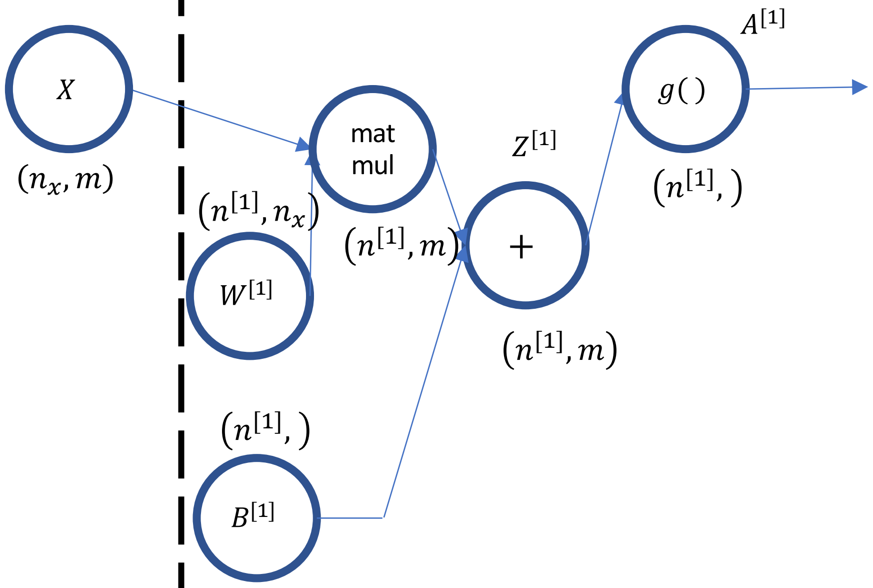
- Shape of $Z^{[1]}$ now has more 2 columns $(n^{[1]}, m)$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



Goal is to calculate:

$$a_1^{[1](1)} = g\left(z_1^{[1](1)}\right) \qquad a_1^{[1](2)} = g\left(z_1^{[1](2)}\right)$$

$$a_2^{[1](1)} = g\left(z_2^{[1](1)}\right) \qquad a_2^{[1](2)} = g\left(z_2^{[1](2)}\right)$$

$$a_3^{[1](1)} = g\left(z_3^{[1](1)}\right) \qquad a_3^{[1](2)} = g\left(z_3^{[1](2)}\right)$$

$$a_4^{[1](1)} = g\left(z_4^{[1](1)}\right) \qquad a_4^{[1](2)} = g\left(z_4^{[1](2)}\right)$$

$$a_5^{[1](1)} = g\left(z_5^{[1](1)}\right) \qquad a_5^{[1](2)} = g\left(z_5^{[1](2)}\right)$$

$A^{[0]} = X$
has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5, 2)

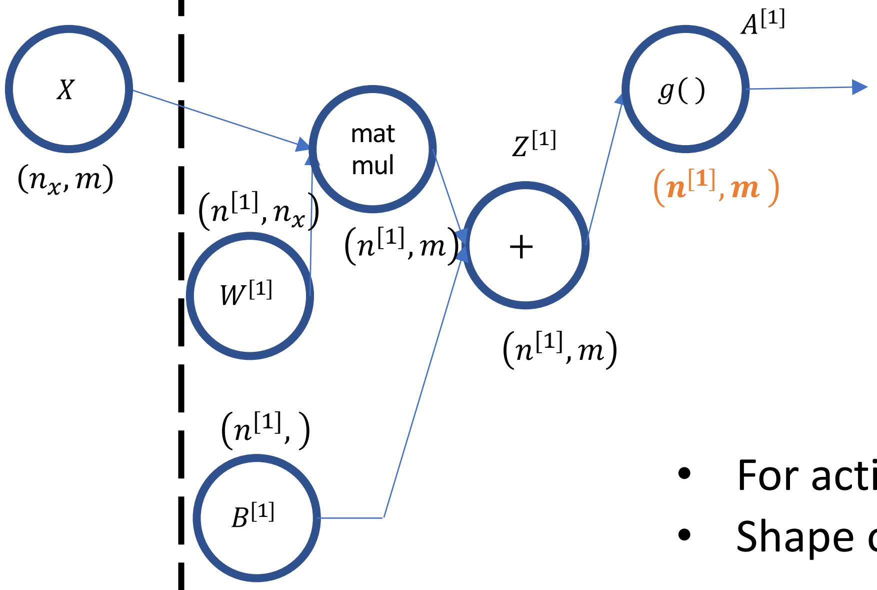
$A^{[1]}$ has shape (5,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



$A^{[0]} = X$
has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5, 2)

$A^{[1]}$ has shape (5, 2)

Goal is to calculate:

$$
\begin{aligned}
a_1^{[1](1)} &= g\left(z_1^{[1](1)}\right) & a_1^{[1](2)} &= g\left(z_1^{[1](2)}\right)\\
a_2^{[1](1)} &= g\left(z_2^{[1](1)}\right) & a_2^{[1](2)} &= g\left(z_2^{[1](2)}\right)\\
a_3^{[1](1)} &= g\left(z_3^{[1](1)}\right) & a_3^{[1](2)} &= g\left(z_3^{[1](2)}\right)\\
a_4^{[1](1)} &= g\left(z_4^{[1](1)}\right) & a_4^{[1](2)} &= g\left(z_4^{[1](2)}\right)\\
a_5^{[1](1)} &= g\left(z_5^{[1](1)}\right) & a_5^{[1](2)} &= g\left(z_5^{[1](2)}\right)
\end{aligned}
$$

- For activation, use elementwise-vector operation as before
- Shape of $A^{[1]}$ is now $(n^{[1]}, m)$

$$A^{[1]} = g(Z^{[1]})$$

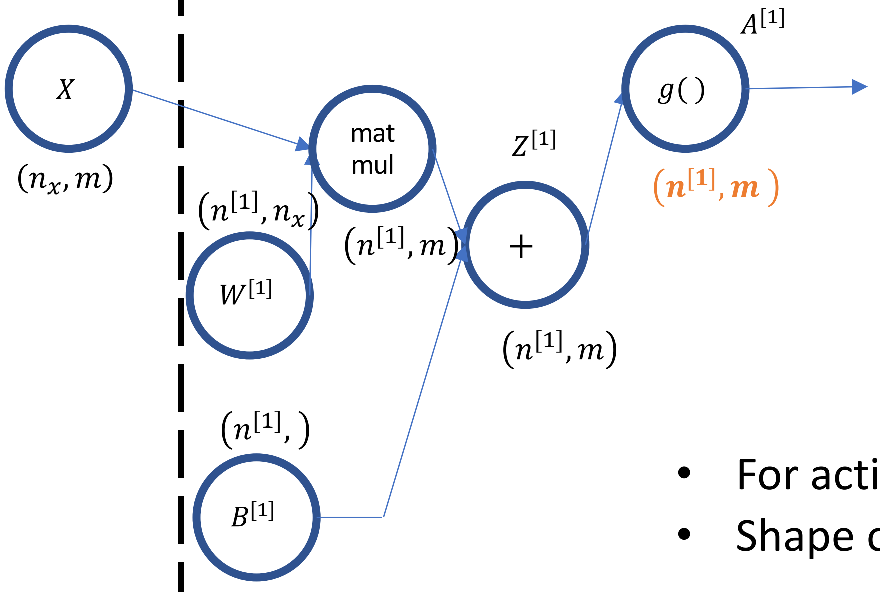
```
# In NumPy
A1 = np.tanh(Z1)
```

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$



Goal is to calculate:

$$a_1^{[1](1)} = g\left(z_1^{[1](1)}\right) \qquad a_1^{[1](2)} = g\left(z_1^{[1](2)}\right)$$
$$a_2^{[1](1)} = g\left(z_2^{[1](1)}\right) \qquad a_2^{[1](2)} = g\left(z_2^{[1](2)}\right)$$
$$a_3^{[1](1)} = g\left(z_3^{[1](1)}\right) \qquad a_3^{[1](2)} = g\left(z_3^{[1](2)}\right)$$
$$a_4^{[1](1)} = g\left(z_4^{[1](1)}\right) \qquad a_4^{[1](2)} = g\left(z_4^{[1](2)}\right)$$
$$a_5^{[1](1)} = g\left(z_5^{[1](1)}\right) \qquad a_5^{[1](2)} = g\left(z_5^{[1](2)}\right)$$

- For activation, use elementwise-vector operation as before
- Shape of $A^{[1]}$ is now $(n^{[1]}, m)$

```
# In NumPy
A1 = np.tanh(Z1)
```

$A^{[0]} = X$
has shape (2, 2)

Consider m=2

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5, 2)

$A^{[1]}$ has shape (5, 2)

$$A^{[1]} = g\left(Z^{[1]}\right)$$

$$= g\left(\begin{bmatrix} z_1^{[1](1)} & z_1^{[1](2)} \\ z_2^{[1](1)} & z_2^{[1](2)} \\ z_3^{[1](1)} & z_3^{[1](2)} \\ z_4^{[1](1)} & z_4^{[1](2)} \\ z_5^{[1](1)} & z_5^{[1](2)} \end{bmatrix}\right) = \begin{bmatrix} g\left(z_1^{[1](1)}\right) & g\left(z_1^{[1](2)}\right) \\ g\left(z_2^{[1](1)}\right) & g\left(z_2^{[1](2)}\right) \\ g\left(z_3^{[1](1)}\right) & g\left(z_3^{[1](2)}\right) \\ g\left(z_4^{[1](1)}\right) & g\left(z_4^{[1](2)}\right) \\ g\left(z_5^{[1](1)}\right) & g\left(z_5^{[1](2)}\right) \end{bmatrix} = \begin{bmatrix} a_1^{[1](1)} & a_1^{[1](2)} \\ a_2^{[1](1)} & a_2^{[1](2)} \\ a_3^{[1](1)} & a_3^{[1](2)} \\ a_4^{[1](1)} & a_4^{[1](2)} \\ a_5^{[1](1)} & a_5^{[1](2)} \end{bmatrix}$$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

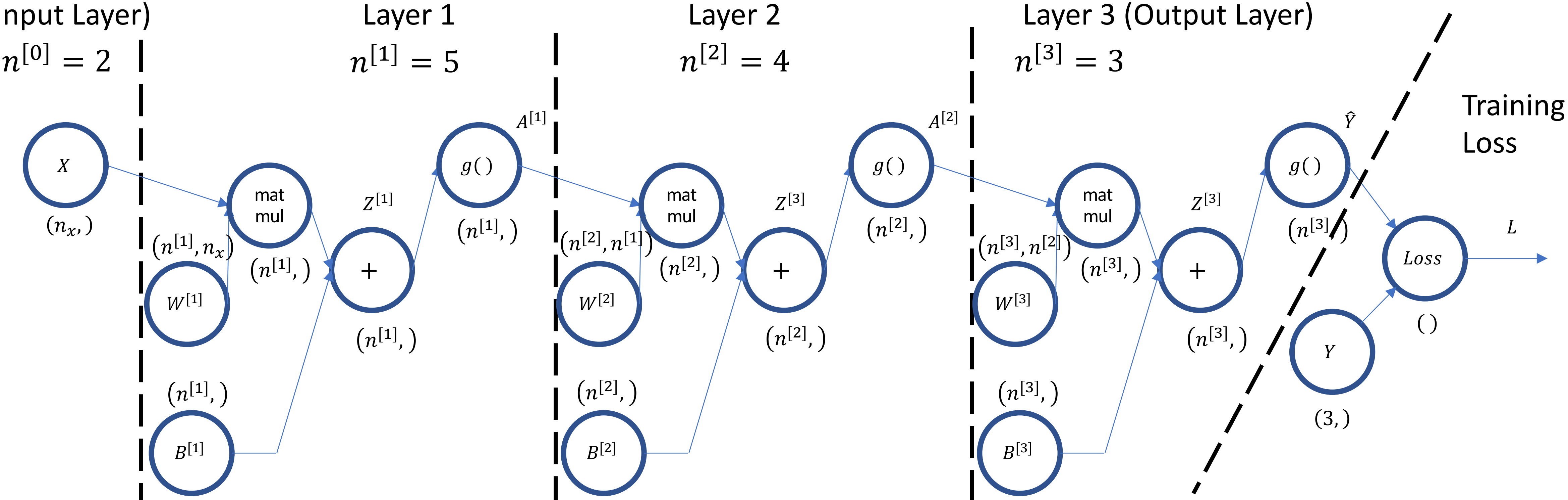
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

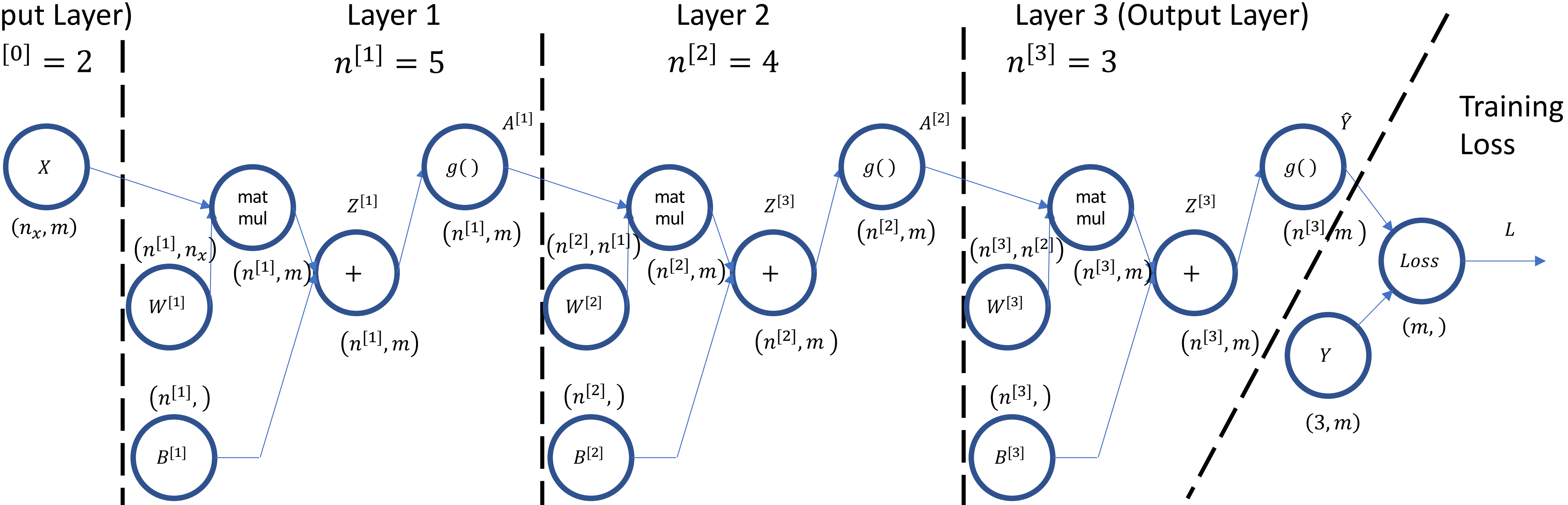
$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

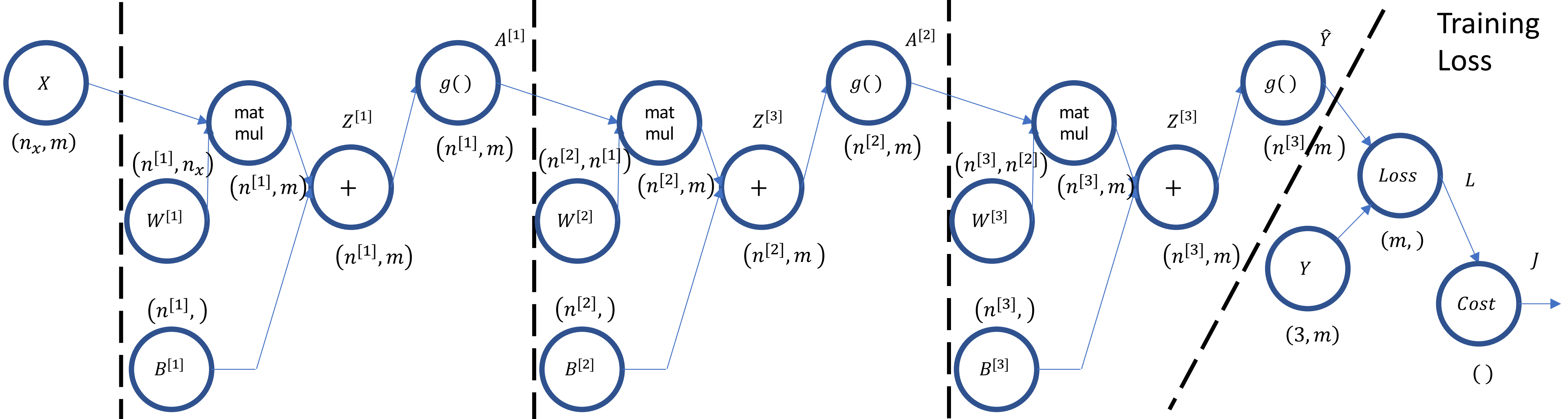
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

Vectorized Backpropagation

Don’t be intimidated!

- There’s nothing crazy (except maybe notation) about vectorized backprop compared to what we have learned for scalar operations
- A vectorized operation is just a bunch of scalar operations done at the same time
- So all our understandings of scalar backprop apply.
  We are just considering multiple scalar operations at once.
- Cost is still a scalar

From First Lectures on Neural Networks

- Vectorized backprop equations in Lecture 5 for a 2-layer neural network. You used these in Assignment 2.
- We will now learn how these are derived.

$$dZ^{[2]} = (\hat{Y} - Y)$$

$$dW^{[2]} = \frac{1}{m} dZ^{[2]} A^{[1]T}$$

$$dB^{[2]} = \frac{1}{m} \sum_{rows} dZ^{[2]}$$

$$dZ^{[1]} = W^{[2]T} dZ^{[2]} * \mathrm{g}'(Z^{[1]})$$

$$dW^{[1]} = \frac{1}{m} dZ^{[1]} X^{T}$$

$$dB^{[1]} = \frac{1}{m} \sum_{rows} dZ^{[1]}$$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

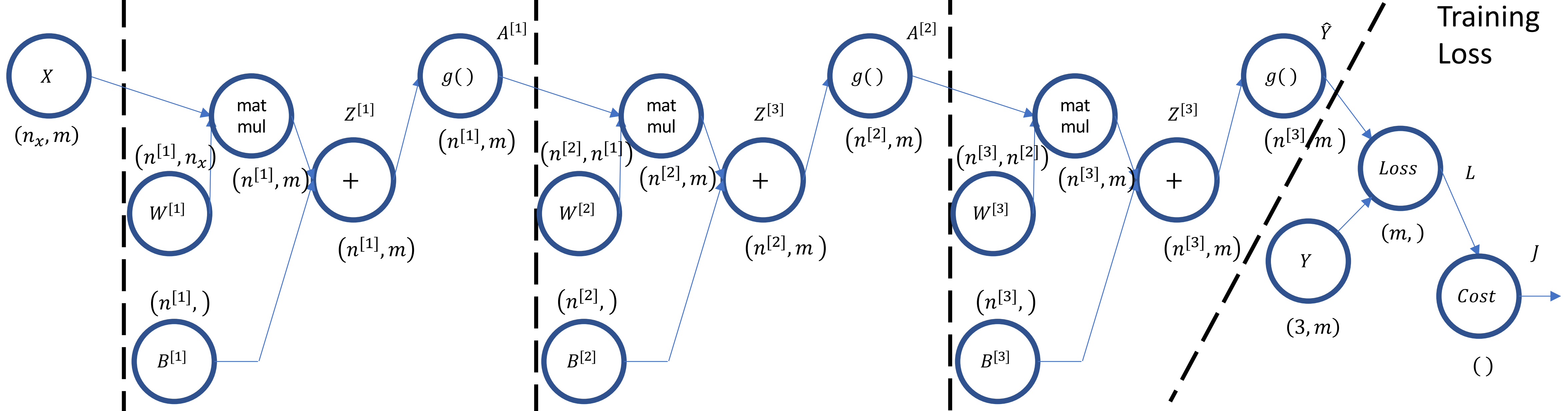
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$ has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

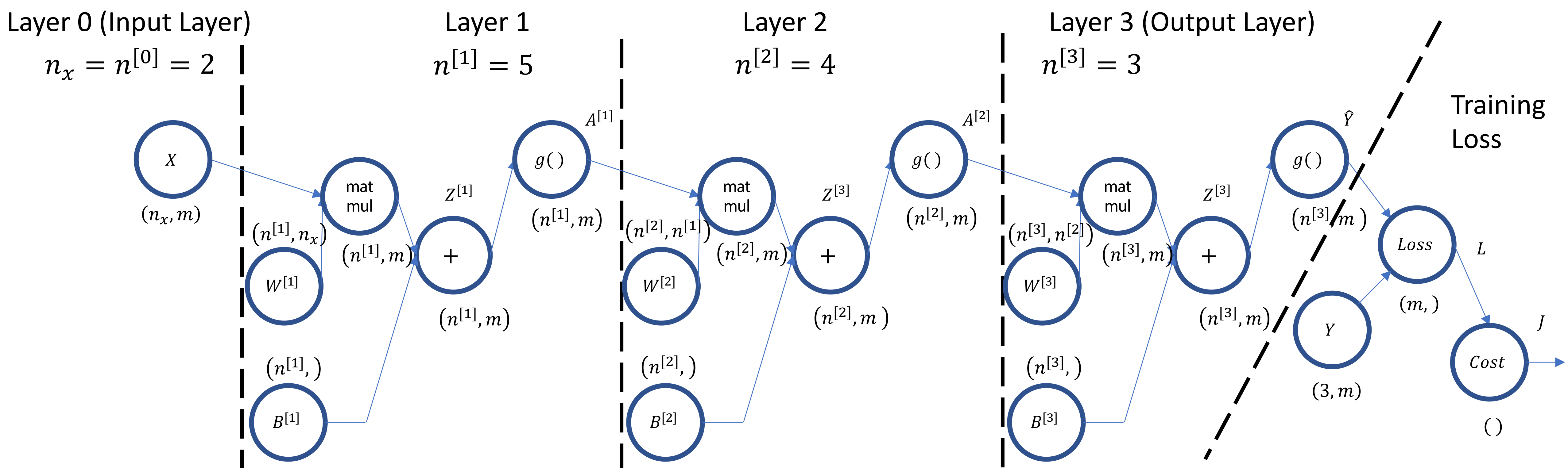
$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

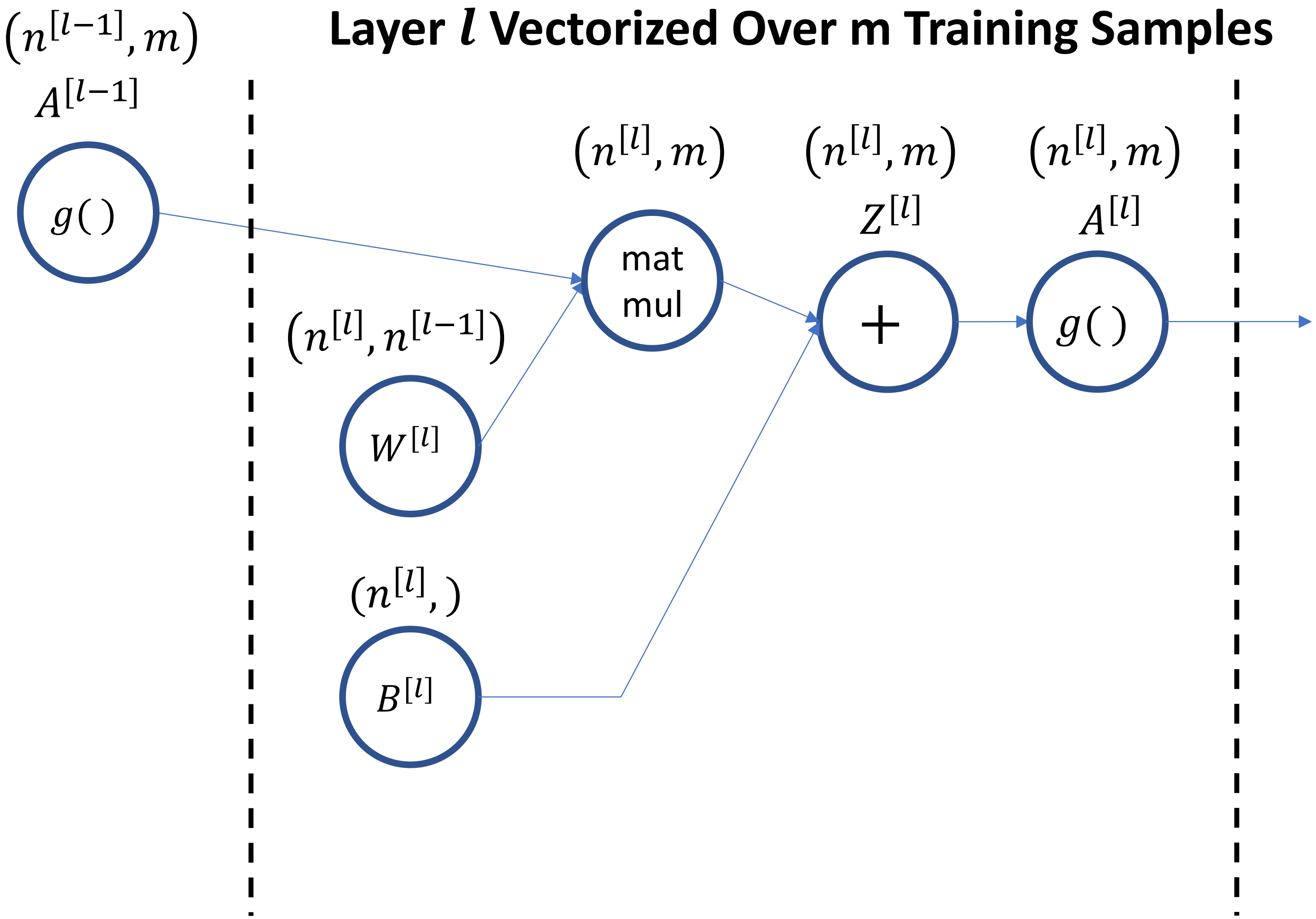
$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

Purpose of backprop is to compute partial derivative of cost J w.r.t. each parameter
This is pretty much the whole essence of the neural network training algorithm

Layer l Vectorized Over m Training Samples



How do we extend our process of backprop to vector operations?

Math – Derivative of a Scalar by a Vector

- Consider a function f with
  - Vector input $x = [x_1, \ldots, x_n]$
  - Scalar output
- How does a change in each input x_i affect the output?
- Gradient Vector

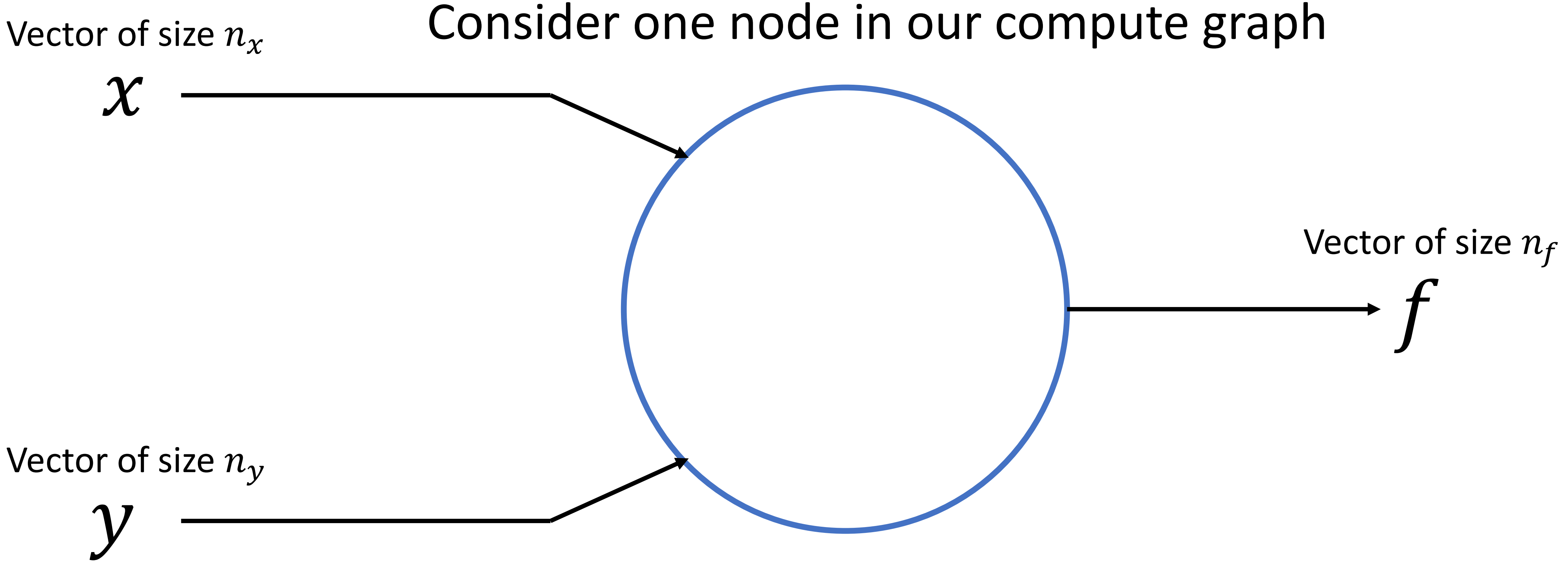
$$\frac{\partial f}{\partial x} = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right]$$

Math – Derivative of a Vector by a Vector

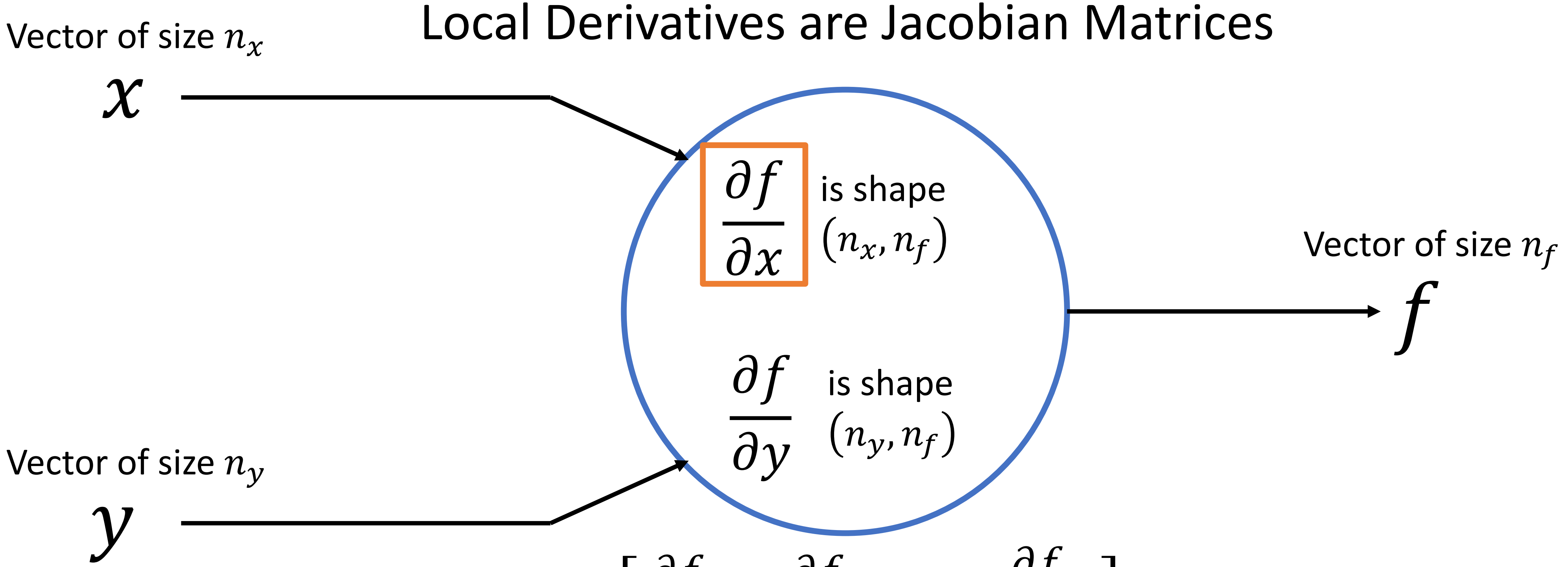
- Consider a function f with
  - Vector input $x = [x_1, \dots, x_n]$
  - Vector output $f = [f_1, \dots, f_m]$
- How does a change in each input affect each output?
- Jacobian Matrix

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_1} \\ \frac{\partial f_1}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_m}{\partial x_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_n} & \frac{\partial f_2}{\partial x_n} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Consider one node in our compute graph



Local Derivatives are Jacobian Matrices



Vector of size n_x

x

$\frac{\partial f}{\partial x}$ is shape (n_x, n_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, n_f)

Vector of size n_f

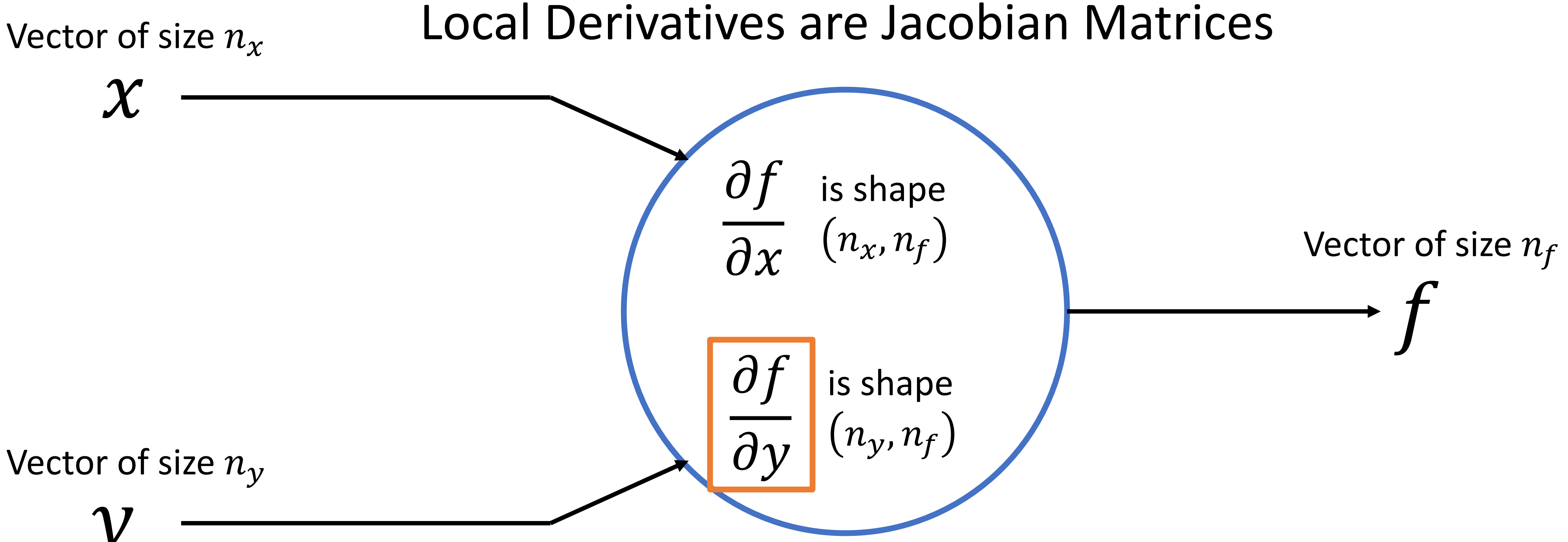
f

Vector of size n_y

y

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_{n_f}}{\partial x_1} \\ \frac{\partial f_1}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_{n_f}}{\partial x_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_{n_x}} & \frac{\partial f_2}{\partial x_{n_x}} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix}$$

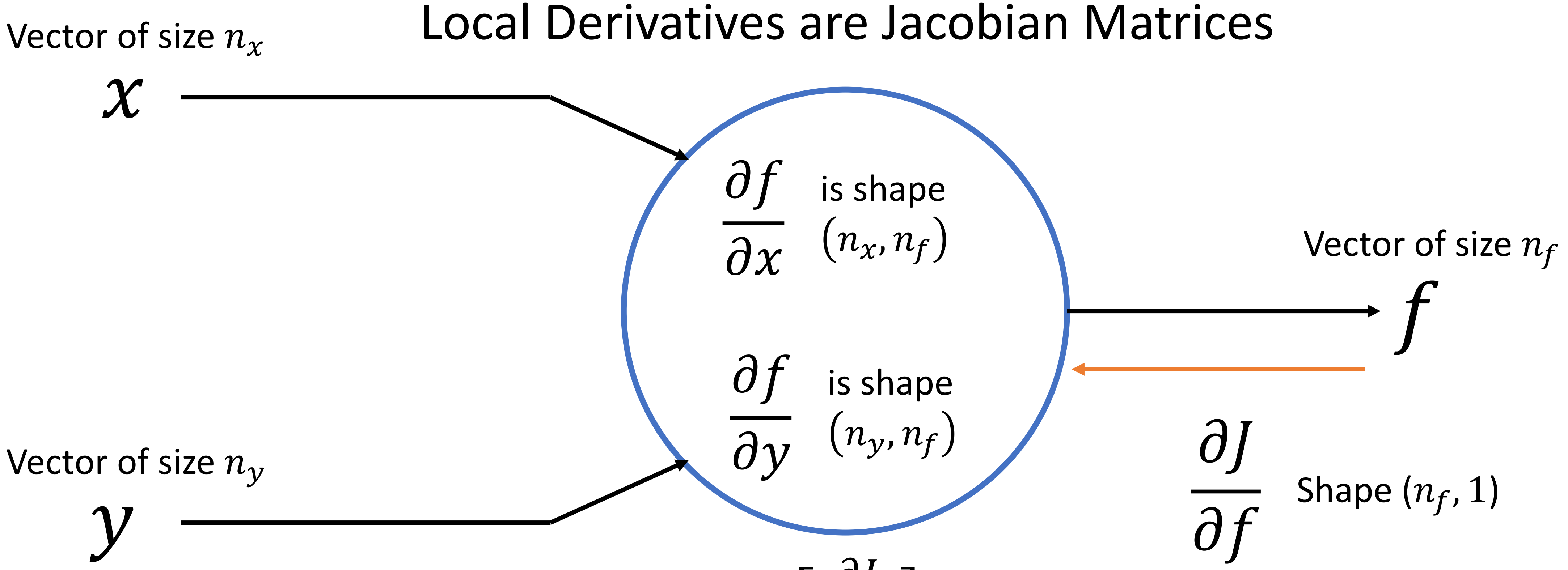
Local Derivatives are Jacobian Matrices



Vector of size n_x
x
$\frac{\partial f}{\partial x}$ is shape (n_x, n_f)
$\boxed{\frac{\partial f}{\partial y}}$ is shape (n_y, n_f)
Vector of size n_f
f
Vector of size n_y
y

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial y_1} & \frac{\partial f_2}{\partial y_1} & \cdots & \frac{\partial f_{n_f}}{\partial y_1} \\ \frac{\partial f_1}{\partial y_2} & \frac{\partial f_2}{\partial y_2} & & \frac{\partial f_{n_f}}{\partial y_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_1}{\partial y_{n_y}} & \frac{\partial f_2}{\partial y_{n_y}} & \cdots & \frac{\partial f_{n_f}}{\partial y_{n_y}} \end{bmatrix}$$

Local Derivatives are Jacobian Matrices



Vector of size n_x

x

$\frac{\partial f}{\partial x}$ is shape (n_x, n_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, n_f)

Vector of size n_y

y

Vector of size n_f

f

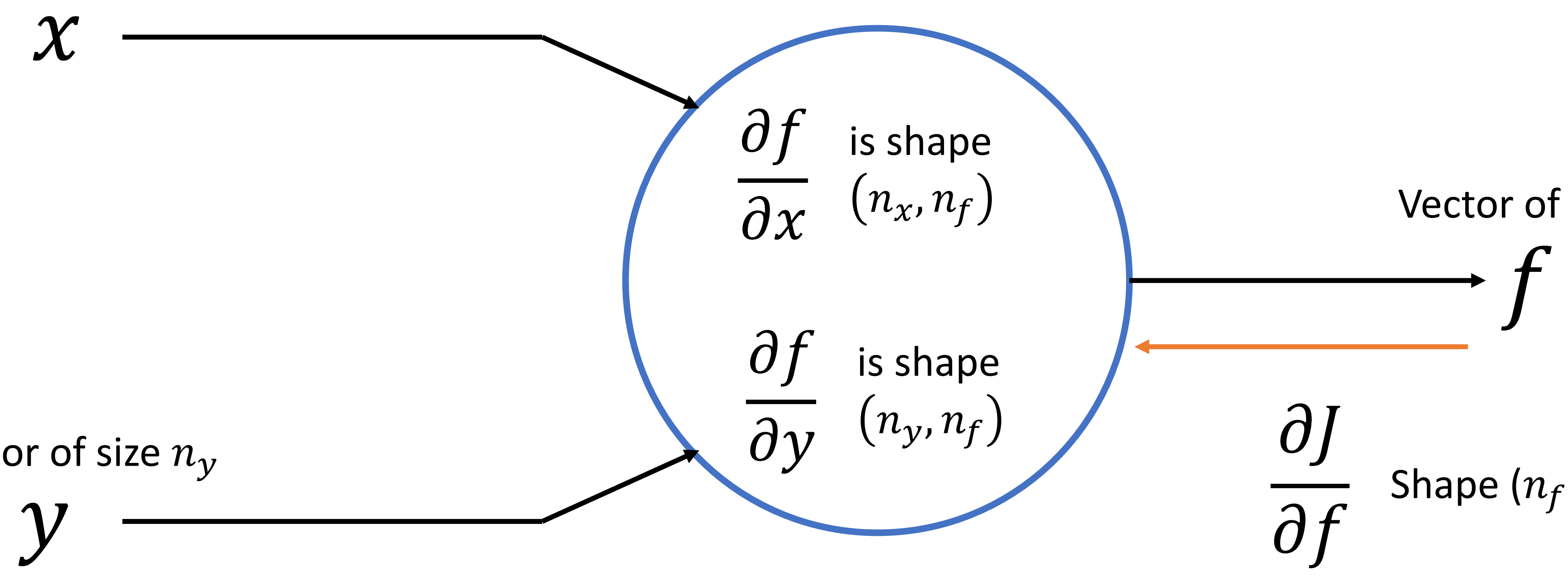
$\frac{\partial J}{\partial f}$ Shape $(n_f, 1)$

$$\frac{\partial J}{\partial f} = \begin{bmatrix} \frac{\partial J}{\partial f_1} \\ \frac{\partial J}{\partial f_2} \\ \vdots \\ \frac{\partial J}{\partial f_{n_f}} \end{bmatrix}$$

Upstream gradient is with respect to Cost J (a **scalar**)

i.e. How does each output f_i affect the Cost

Local Derivatives are Jacobian Matrices



Vector of size n_x

x

Vector of size n_y

y

$\frac{\partial f}{\partial x}$ is shape (n_x, n_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, n_f)

Vector of size n_f

f

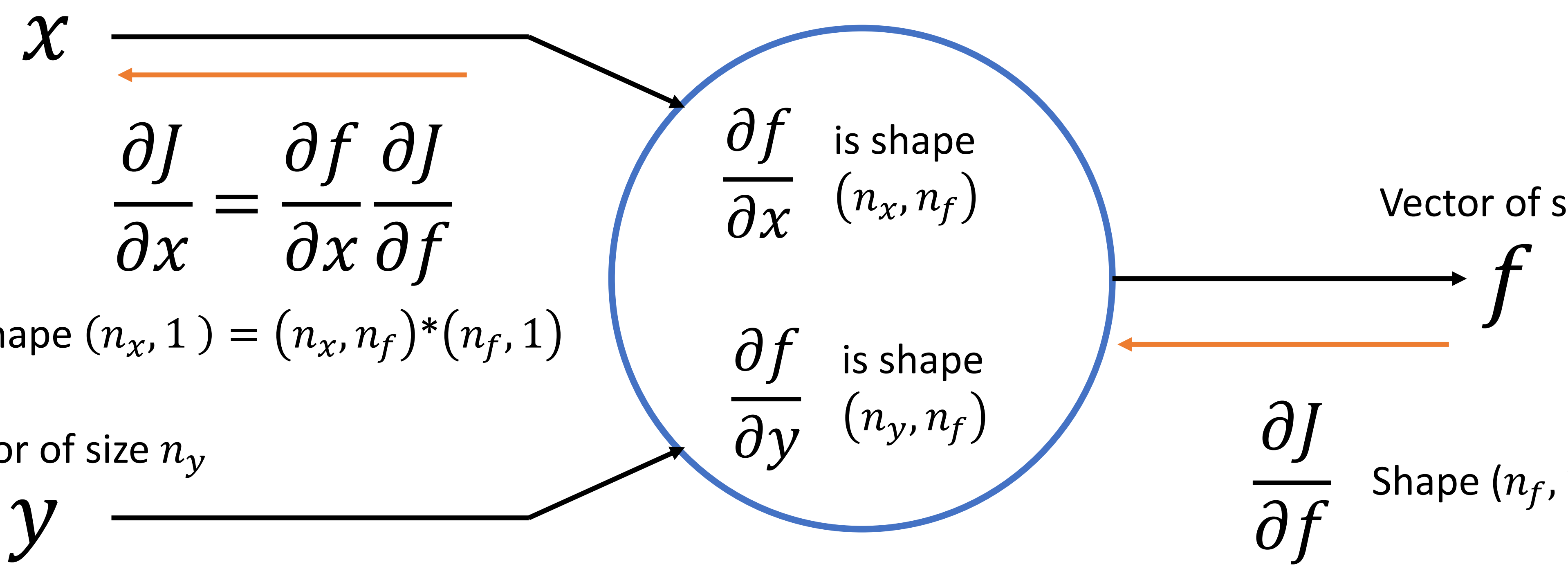
$\frac{\partial J}{\partial f}$ Shape $(n_f, 1)$

Upstream gradient is with respect to Cost J (a **scalar**)

i.e. How does each output f_i affect the Cost

Apply chain rule like before!

Local Derivatives are Jacobian Matrices



Vector of size n_x

x

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f}$$

Shape $(n_x, 1\,) = (n_x, n_f)$*$(n_f, 1)$

Vector of size n_y

y

$\frac{\partial f}{\partial x}$ is shape (n_x, n_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, n_f)

Vector of size n_f

f

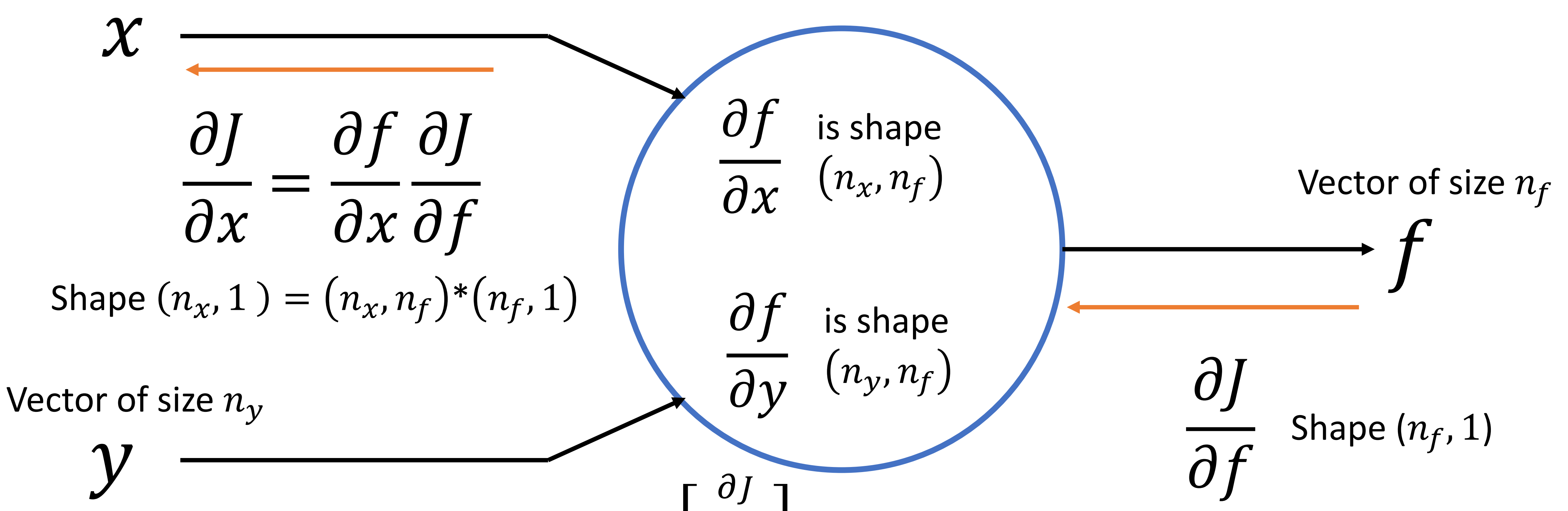
$\frac{\partial J}{\partial f}$ Shape $(n_f, 1)$

Upstream gradient is with respect to Cost J (a **scalar**)

i.e. How does each output f_i affect the Cost

Chain Rule application is Matrix-Vector Multiply

Local Derivatives are Jacobian Matrices



Vector of size n_x

x

Vector of size n_y

y

$\frac{\partial f}{\partial x}$ is shape (n_x, n_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, n_f)

Vector of size n_f

f

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f}$$

Shape $(n_x, 1\,) = (n_x, n_f)^{*}(n_f, 1)$

$\frac{\partial J}{\partial f}$ Shape $(n_f, 1)$

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}, \frac{\partial f_1}{\partial x_2}, & \cdots & \frac{\partial f_1}{\partial x_{n_x}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{n_f}}{\partial x_1}, \frac{\partial f_{n_f}}{\partial x_2} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \begin{bmatrix} \frac{\partial J}{\partial f_1} \\ \frac{\partial J}{\partial f_2} \\ \vdots \\ \frac{\partial J}{\partial f_{n_f}} \end{bmatrix}$$

Upstream gradient is with respect to Cost J (a **scalar**)

i.e. How does each output f_i affect the Cost

Chain Rule application is Matrix-Vector Multiply

Chain Rule – Matrix-Vector Multiply

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f}$$

Chain Rule – Matrix-Vector Multiply

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f} \rightarrow \begin{bmatrix} \frac{\partial J}{\partial x_1} \\ \frac{\partial J}{\partial x_2} \\ \vdots \\ \frac{\partial J}{\partial x_{n_x}} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}, \frac{\partial f_2}{\partial x_1}, & \cdots & \frac{\partial f_{n_f}}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_{n_x}}, \frac{\partial f_2}{\partial x_{n_x}} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \begin{bmatrix} \frac{\partial J}{\partial f_1} \\ \frac{\partial J}{\partial f_2} \\ \vdots \\ \frac{\partial J}{\partial f_{n_f}} \end{bmatrix}$$

Shape $(n_x, 1\ ) = \left(n_x, n_f\right)^*\left(n_f, 1\right)$

Chain Rule – Matrix-Vector Multiply

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f} \rightarrow \begin{bmatrix} \frac{\partial J}{\partial x_1} \\ \frac{\partial J}{\partial x_2} \\ \vdots \\ \frac{\partial J}{\partial x_{n_x}} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}, \frac{\partial f_2}{\partial x_1}, & \cdots & \frac{\partial f_{n_f}}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_{n_x}}, \frac{\partial f_2}{\partial x_{n_x}} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \begin{bmatrix} \frac{\partial J}{\partial f_1} \\ \frac{\partial J}{\partial f_2} \\ \vdots \\ \frac{\partial J}{\partial f_{n_f}} \end{bmatrix}$$

Jacobian

Shape $(n_x, 1\ ) = (n_x, n_f)*(n_f, 1)$

Chain Rule – Matrix-Vector Multiply

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f} \rightarrow \begin{bmatrix} \frac{\partial J}{\partial x_1} \\ \frac{\partial J}{\partial x_2} \\ \vdots \\ \frac{\partial J}{\partial x_{n_x}} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}, \frac{\partial f_2}{\partial x_1}, & \cdots & \frac{\partial f_{n_f}}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_{n_x}}, \frac{\partial f_2}{\partial x_{n_x}} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \begin{bmatrix} \frac{\partial J}{\partial f_1} \\ \frac{\partial J}{\partial f_2} \\ \vdots \\ \frac{\partial J}{\partial f_{n_f}} \end{bmatrix}$$

Jacobian

Shape $(n_x, 1\,) = (n_x, n_f)$*$(n_f, 1)$

Upstream Gradient

Chain Rule – Matrix-Vector Multiply

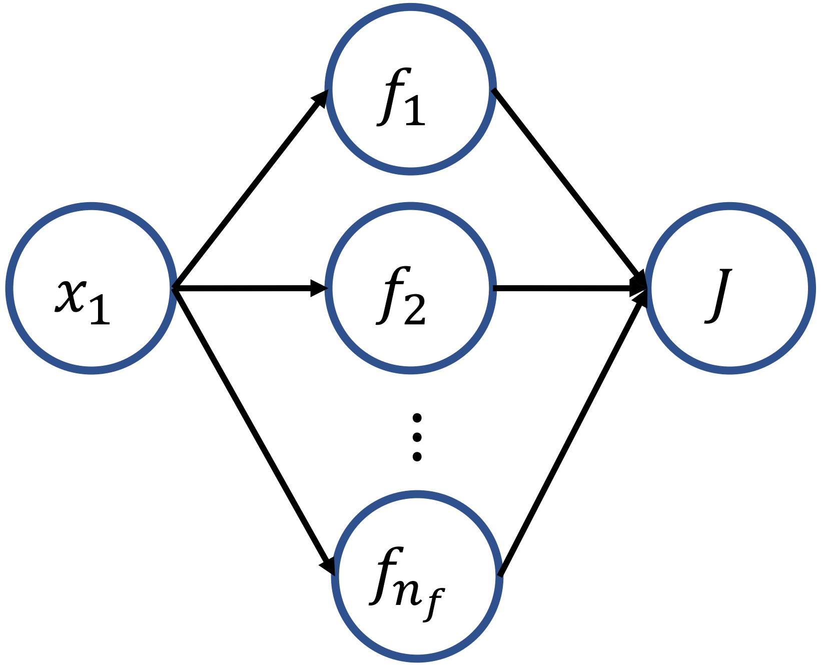
$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f} \rightarrow \begin{bmatrix} \boxed{\dfrac{\partial J}{\partial x_1}} \\ \dfrac{\partial J}{\partial x_2} \\ \vdots \\ \dfrac{\partial J}{\partial x_{n_x}} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1}, \dfrac{\partial f_2}{\partial x_1}, & \cdots & \dfrac{\partial f_{n_f}}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_1}{\partial x_{n_x}}, \dfrac{\partial f_2}{\partial x_{n_x}} & \cdots & \dfrac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \boxed{\begin{bmatrix} \dfrac{\partial J}{\partial f_1} \\ \dfrac{\partial J}{\partial f_2} \\ \vdots \\ \dfrac{\partial J}{\partial f_{n_f}} \end{bmatrix}}$$

Shape $(n_x, 1\ ) = (n_x, n_f)*(n_f, 1)$

$$\frac{\partial J}{\partial x_1} = \frac{\partial f_1}{\partial x_1}\frac{\partial J}{\partial f_1} + \frac{\partial f_2}{\partial x_1}\frac{\partial J}{\partial f_2} + \cdots + \frac{\partial f_{n_f}}{\partial x_1}\frac{\partial J}{\partial f_{n_f}}$$

Chain Rule – Matrix-Vector Multiply

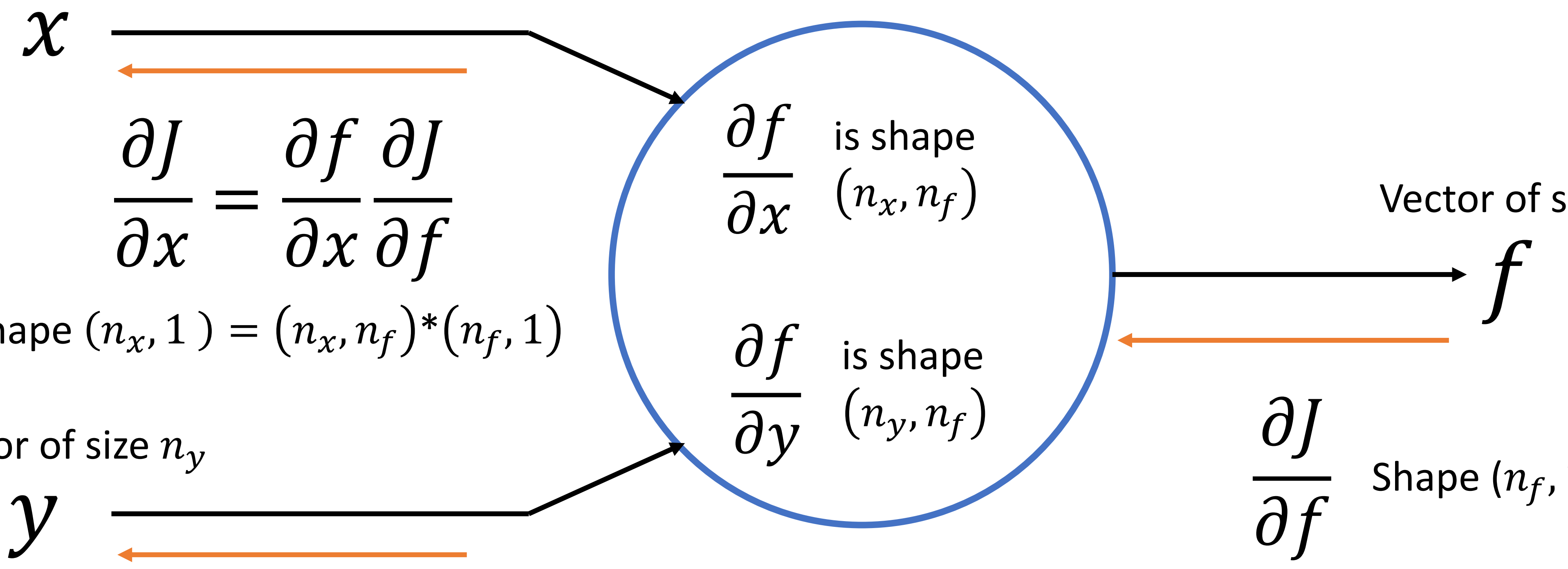
$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f} \rightarrow \begin{bmatrix} \boxed{\frac{\partial J}{\partial x_1}} \\ \frac{\partial J}{\partial x_2} \\ \vdots \\ \frac{\partial J}{\partial x_{n_x}} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}, \frac{\partial f_2}{\partial x_1}, & \cdots & \frac{\partial f_{n_f}}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_{n_x}}, \frac{\partial f_2}{\partial x_{n_x}} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \boxed{\begin{bmatrix} \frac{\partial J}{\partial f_1} \\ \frac{\partial J}{\partial f_2} \\ \vdots \\ \frac{\partial J}{\partial f_{n_f}} \end{bmatrix}}$$



Shape $(n_x, 1) = (n_x, n_f) * (n_f, 1)$

$$\frac{\partial J}{\partial x_1} = \frac{\partial f_1}{\partial x_1}\frac{\partial J}{\partial f_1} + \frac{\partial f_2}{\partial x_1}\frac{\partial J}{\partial f_2} + \cdots + \frac{\partial f_{n_f}}{\partial x_1}\frac{\partial J}{\partial f_{n_f}}$$

Local Derivatives are Jacobian Matrices



Vector of size n_x

x

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f}$$

Shape $(n_x, 1\ ) = (n_x, n_f)$*$(n_f, 1)$

Vector of size n_y

y

$$\frac{\partial J}{\partial y} = \frac{\partial f}{\partial y}\frac{\partial J}{\partial f}$$

Shape $(n_y, 1\ ) = (n_y, n_f)$*$(n_f, 1)$

$\frac{\partial f}{\partial x}$ is shape (n_x, n_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, n_f)

Vector of size n_f

f

$\frac{\partial J}{\partial f}$ Shape $(n_f, 1)$

Upstream gradient is with respect to Cost J (a **scalar**)

i.e. How does each output f_i affect the Cost

Chain Rule application is Matrix-Vector Multiply

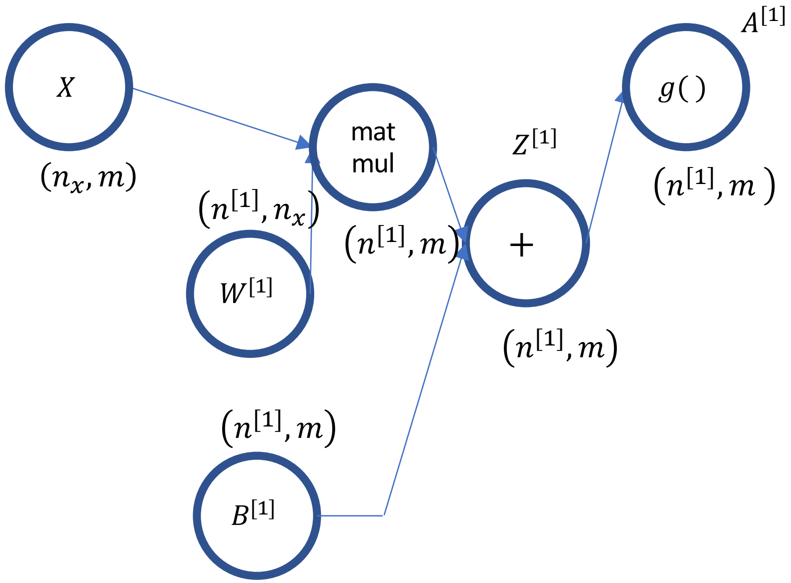
Example:
Activation Function on Layer

Example – Activation Function on Layer

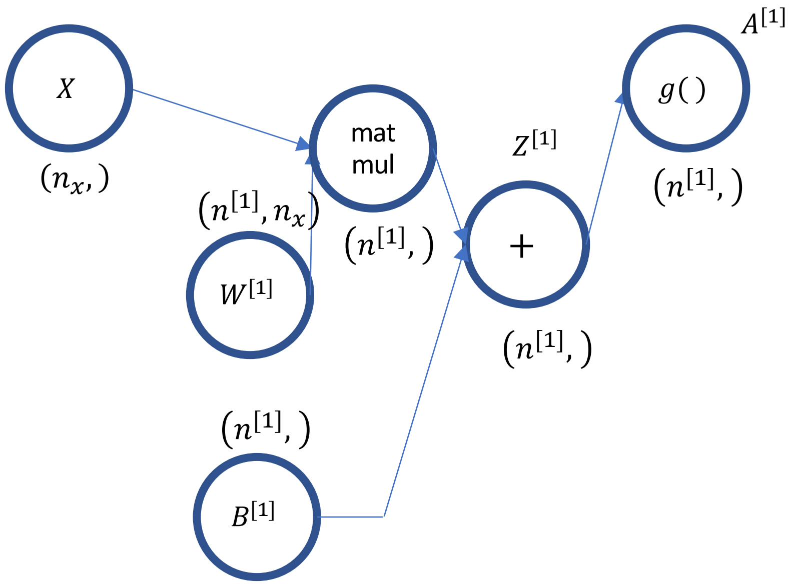
- A common vector in, vector out, computation is applying the activation on all units in a layer (for one sample).
- Vectorized computation of $a_i^{[l]} = g\left(z_i^{[l]}\right)$ where $g(\quad)$ is the activation function such as ReLU, tanh, etc.
- Specifically, we want to compute the activation for all units in the layer with one vectorized operation.

```
# For example in NumPy
Z2 = np.matmul(W2, A1) + B2
A2 = np.tanh(Z2)
```

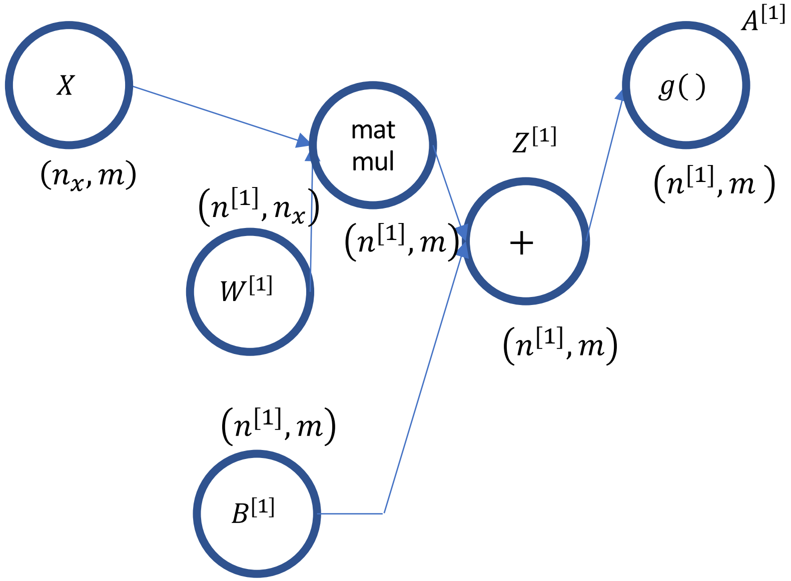
Vectorized for multiple samples



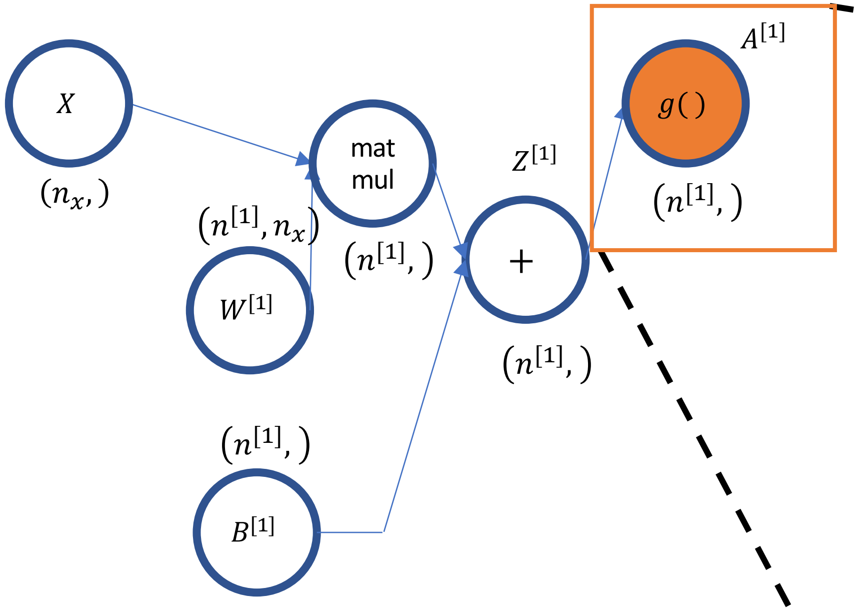
Vectorized for one sample

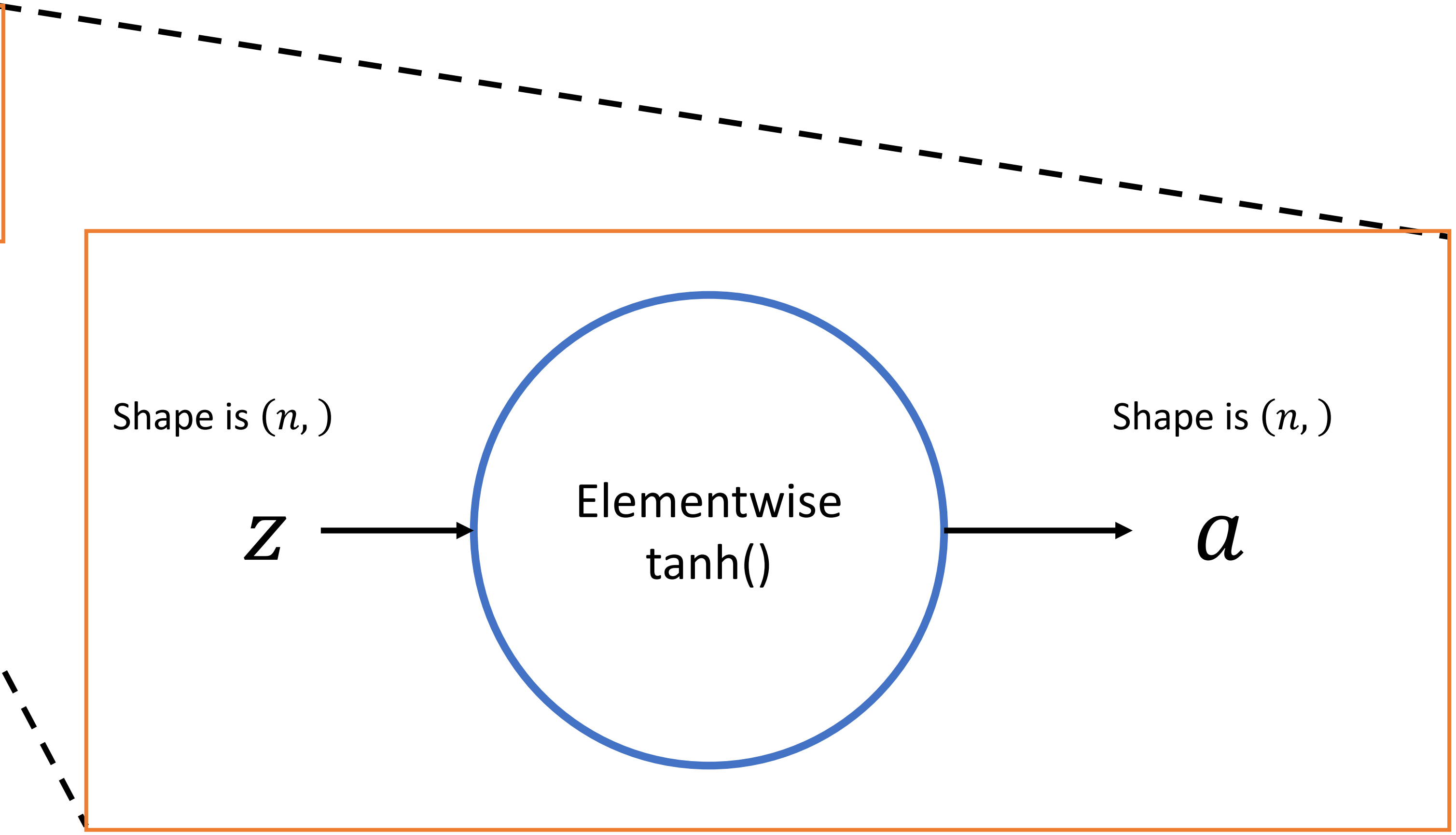


Vectorized for multiple samples

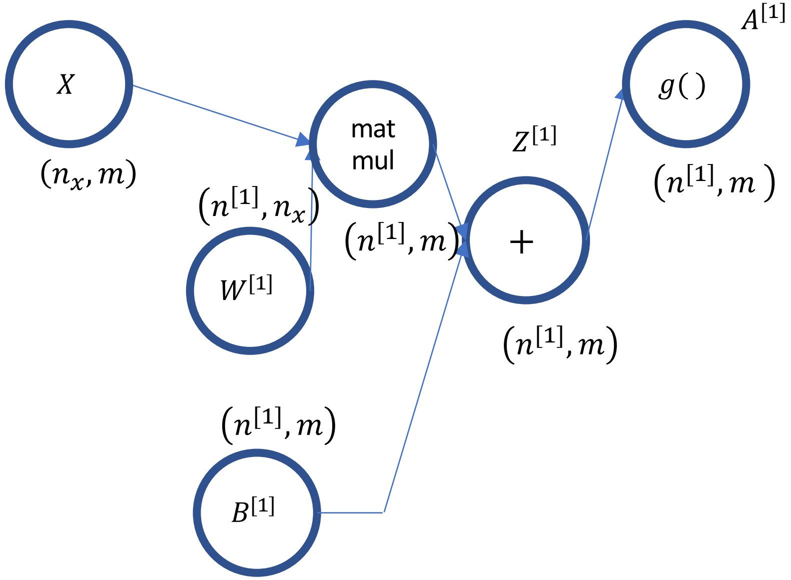


Vectorized for one sample



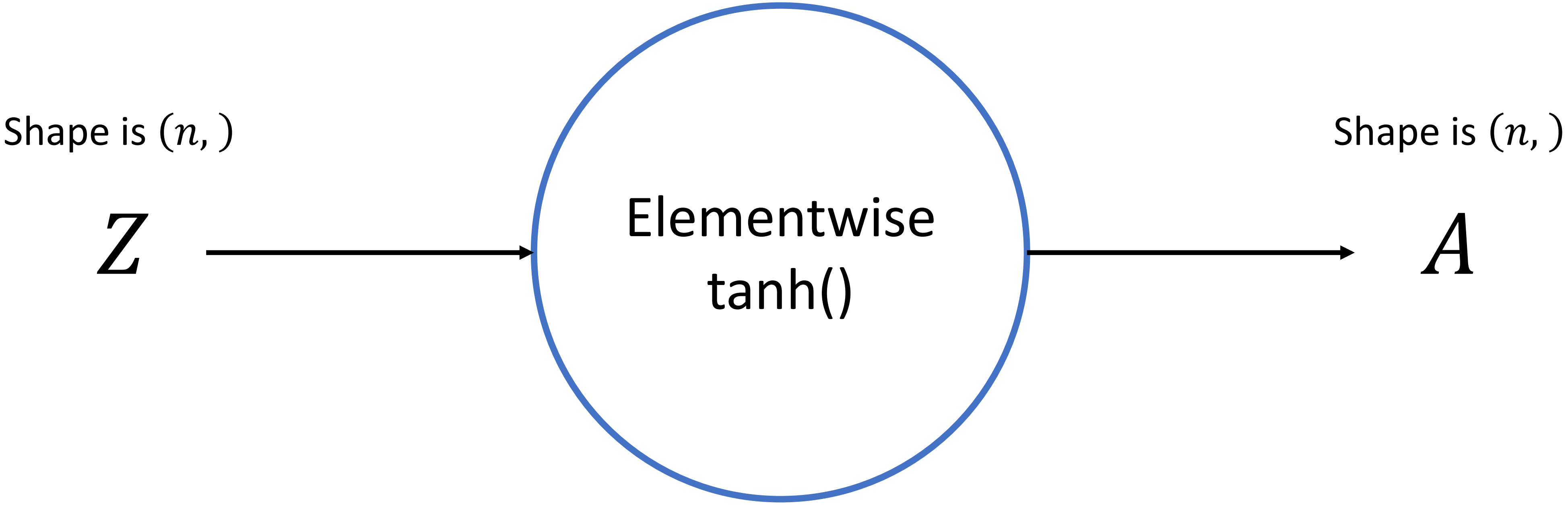


Vectorized for multiple samples



Example: Tanh Activation on Layer for one sample

Example: Tanh Activation Function on Layer for one sample



Shape is $(n,)$

Z

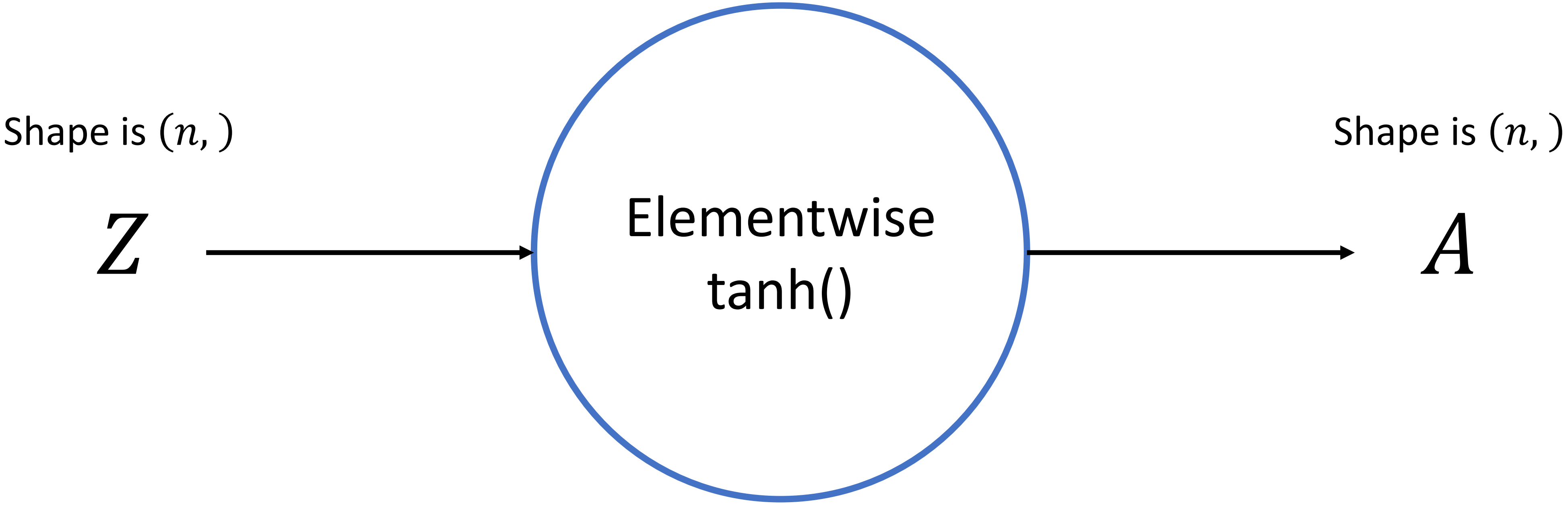
Elementwise tanh()

Shape is $(n,)$

A

$$A = \tanh(Z) = tanh\left(\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_n \end{bmatrix}\right) = \begin{bmatrix} tanh(z_1) \\ tanh(z_2) \\ tanh(z_3) \\ \vdots \\ tanh(z_n) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$

Example: Tanh Activation Function on Layer for one sample



Shape is $(n,)$

Z

Elementwise tanh()

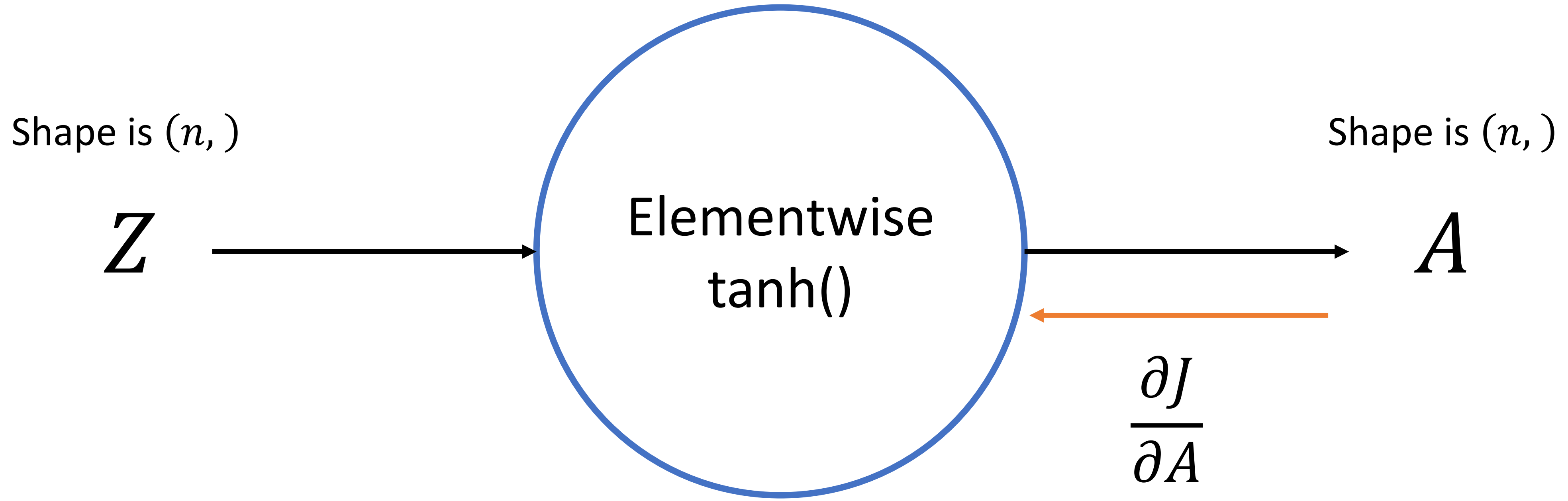
Shape is $(n,)$

A

Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_n}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_n}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_n} & \frac{\partial a_2}{\partial z_n} & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

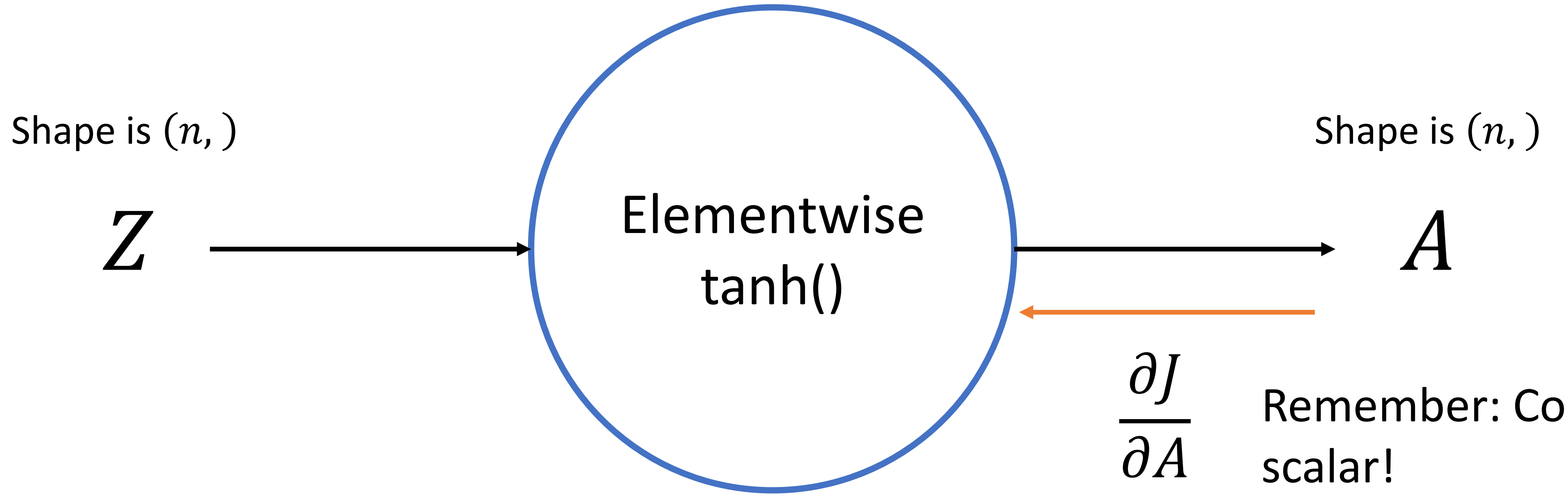
Example: Tanh Activation Function on Layer for one sample



Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_n}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_n}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_n} & \frac{\partial a_2}{\partial z_n} & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

Example: Tanh Activation Function on Layer for one sample



Remember: Cost is scalar!

This is a vector of shape $(n,)$ that tells us how a change in each element a_i of vector A will affect the Cost.

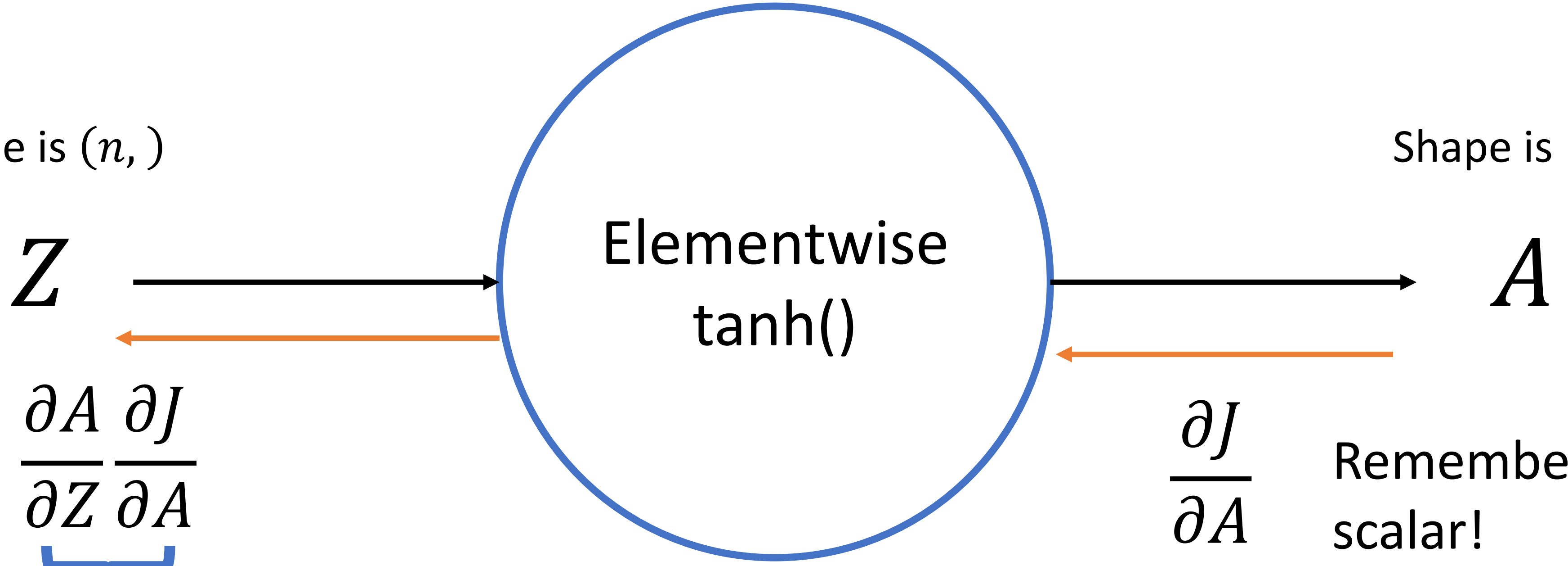
Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_n}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_n}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_n} & \frac{\partial a_2}{\partial z_n} & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

Example: Tanh Activation Function on Layer for one sample

Shape is $(n,)$

Shape is $(n,)$



$$\frac{\partial J}{\partial Z} = \underbrace{\frac{\partial A}{\partial Z}\frac{\partial J}{\partial A}}_{\text{Matrix-Vector Multiply}}$$

$$\frac{\partial J}{\partial A}$$

Remember: Cost is scalar!

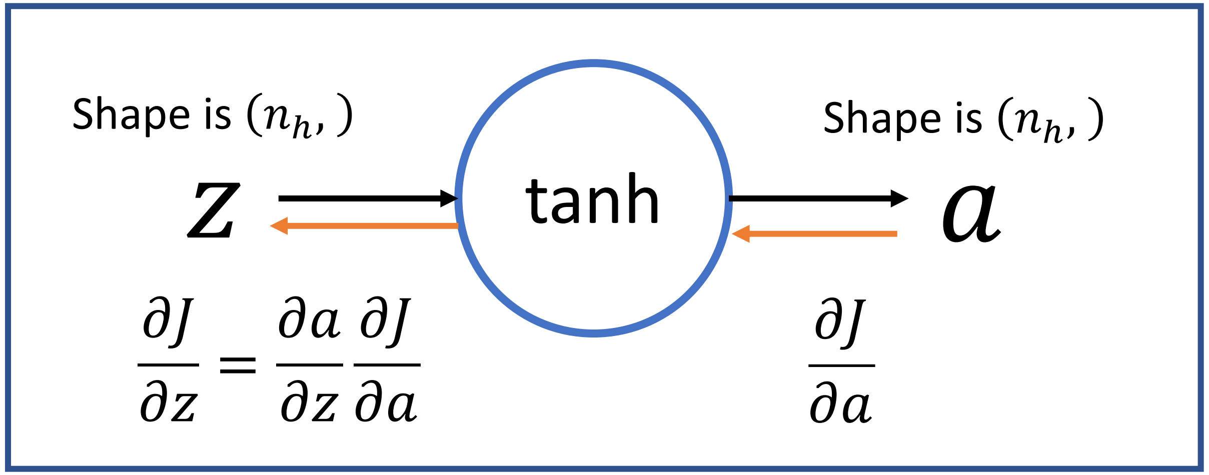
This is a vector of shape $(n,)$ that tells us how a change in each element a_i of vector A will affect the Cost.

Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_n}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_n}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_n} & \frac{\partial a_2}{\partial z_n} & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

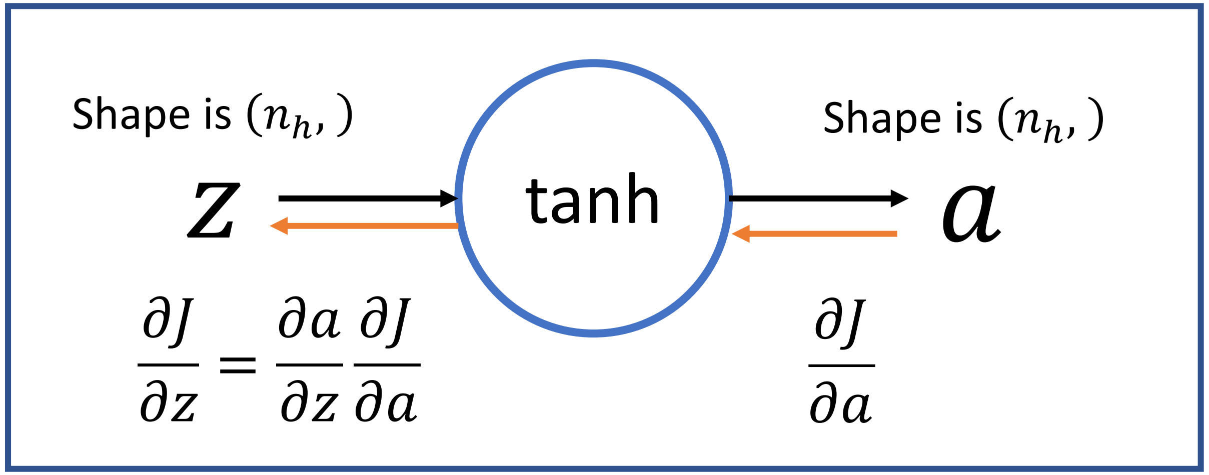
Example: Tanh Activation Function on Layer for one sample

$$\frac{\partial a}{\partial z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_{n_h}}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_{n_h}}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_{n_h}} & \frac{\partial a_2}{\partial z_{n_h}} & \cdots & \frac{\partial a_{n_h}}{\partial z_{n_h}} \end{bmatrix}$$


Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

Example: Tanh Activation Function on Layer for one sample

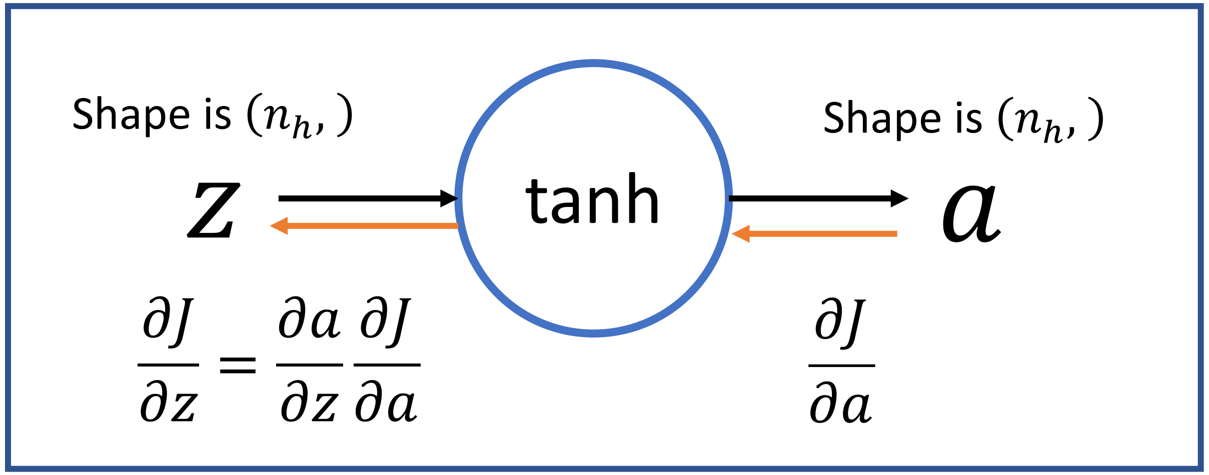
$$\frac{\partial a}{\partial z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_{n_h}}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_{n_h}}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_{n_h}} & \frac{\partial a_2}{\partial z_{n_h}} & \cdots & \frac{\partial a_{n_h}}{\partial z_{n_h}} \end{bmatrix}$$


Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

- Jacobian Matrices can get impractically large.

Example: Tanh Activation Function on Layer for one sample

$$\frac{\partial a}{\partial z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_{n_h}}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_{n_h}}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_{n_h}} & \frac{\partial a_2}{\partial z_{n_h}} & \cdots & \frac{\partial a_{n_h}}{\partial z_{n_h}} \end{bmatrix}$$


Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

- Jacobian Matrices can get impractically large.
- Any simplifications in this case of an **element-wise** operation?

Remember:

$a_1 = \tanh(z_1)$

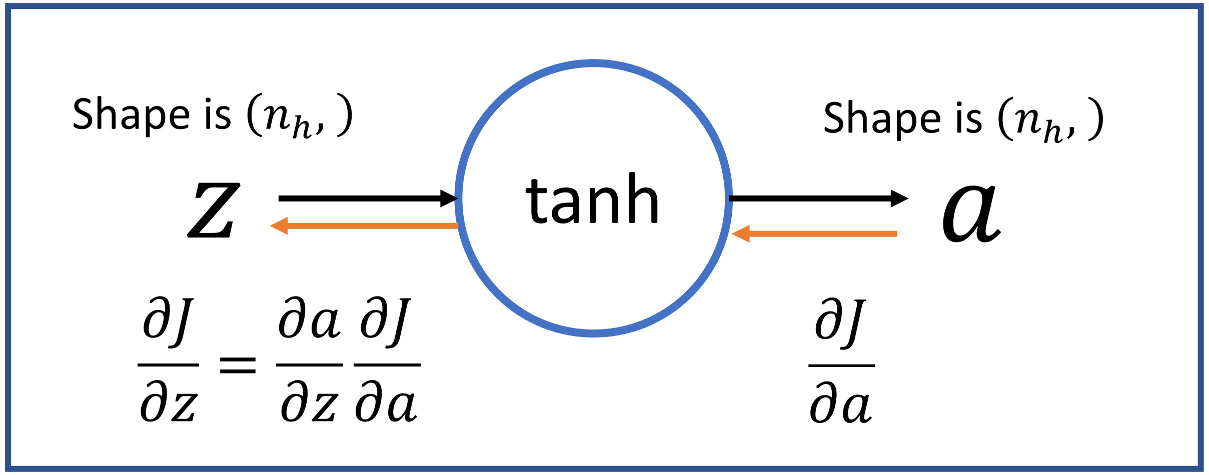
$a_2 = \tanh(z_2)$

$\vdots$

$a_{n_h} = \tanh(z_{n_h})$

Example: Tanh Activation Function on Layer for one sample

$$\frac{\partial a}{\partial z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & 0 & \cdots & 0 \\ 0 & \frac{\partial a_2}{\partial z_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{\partial a_{n_h}}{\partial z_{n_h}} \end{bmatrix}$$


Shape is $(n_h,)$
Shape is $(n_h,)$
z
tanh
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

Remember:

$$a_1 = \tanh(z_1)$$
$$a_2 = \tanh(z_2)$$
$$\vdots$$
$$a_{n_h} = \tanh(z_{n_h})$$

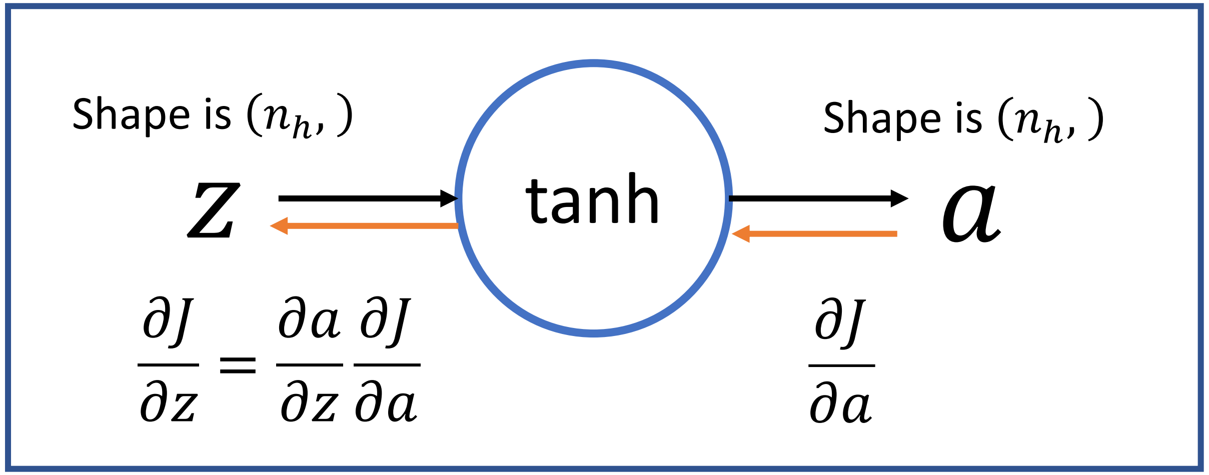
- Jacobian Matrices can get impractically large.
- Any simplifications in this case of an
- element-wise operation?
- Jacobian is a **diagonal matrix**!
- This is true for all elementwise vector operations!

Jacobian Matrix for Element-wise Vector Operations

- Jacobian is diagonal (hence sparse) for element-wise vector operations
- Turns out, (most) vector operations used in neural networks have sparse Jacobian matrices
- Do not need to construct the full Jacobian matrix and never have to compute its full matrix-vector multiply with the upstream gradients
- Vectorized backprop is all about taking advantage of this, and getting around full Jacobian construction and multiplication

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$



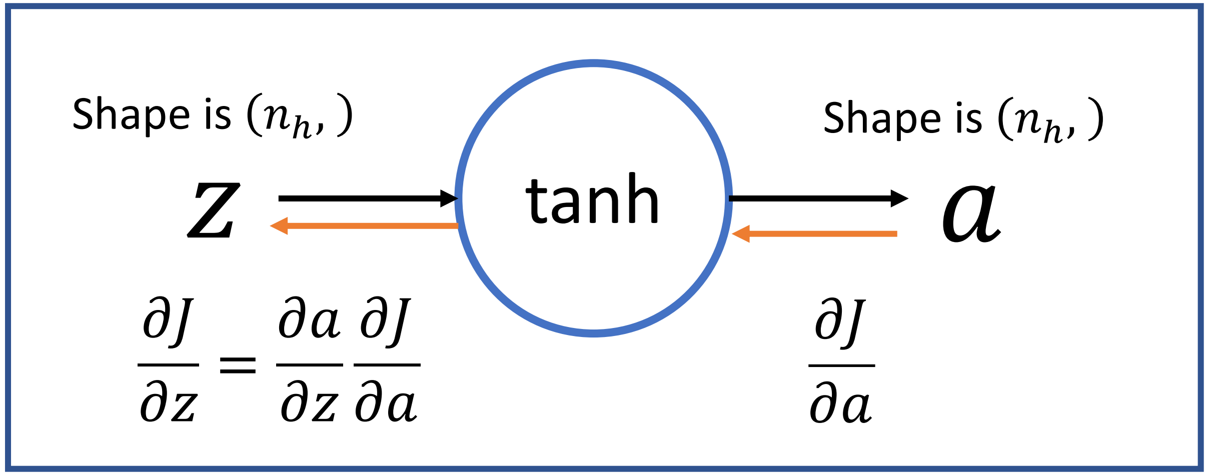
Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2$$

Compute this with element-wise vector operations



Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

Example: Tanh Activation Function on Layer for one sample

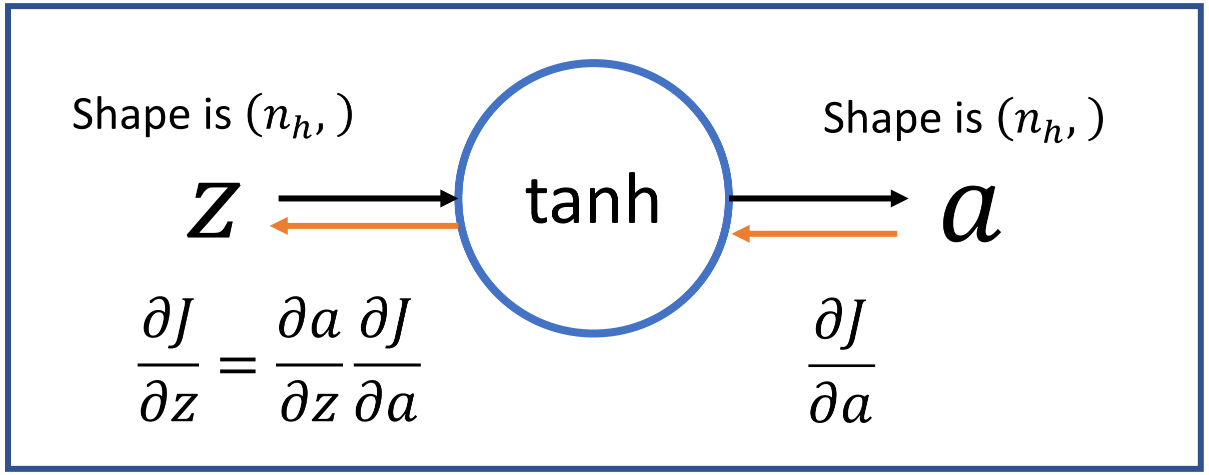
$$a_i = \tanh(z_i)$$

$$\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2$$

Compute this with element-wise vector operations

$$\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}$$

Compute this with
element-wise vector multipication

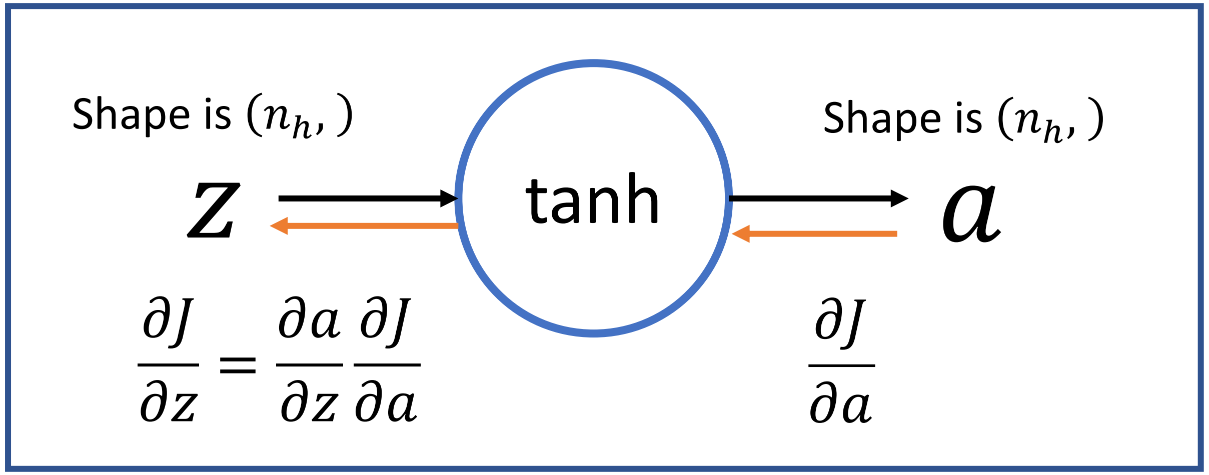

Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2$$

Compute this with element-wise vector operations



Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

$$\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}$$

Compute this with **element-wise** vector multipication

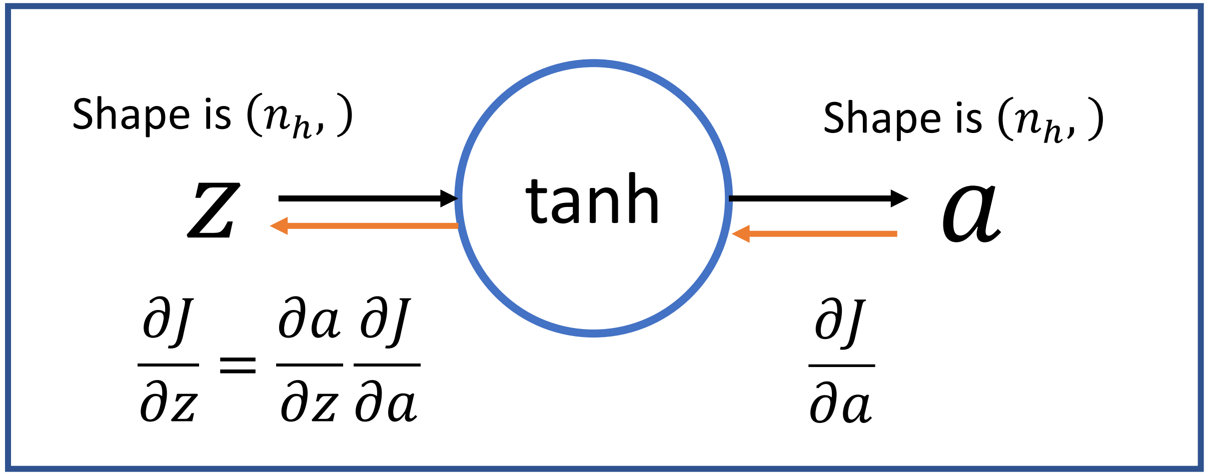
Full Jacobian is NEVER computed
We don’t do a full matrix-vector multiply!

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2$$

Compute this with **element-wise** vector operations



Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial J}{\partial a}$
$\frac{\partial J}{\partial a}$

$$\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}$$

Compute this with
element-wise vector multipication

For example, in numpy (for one sample):

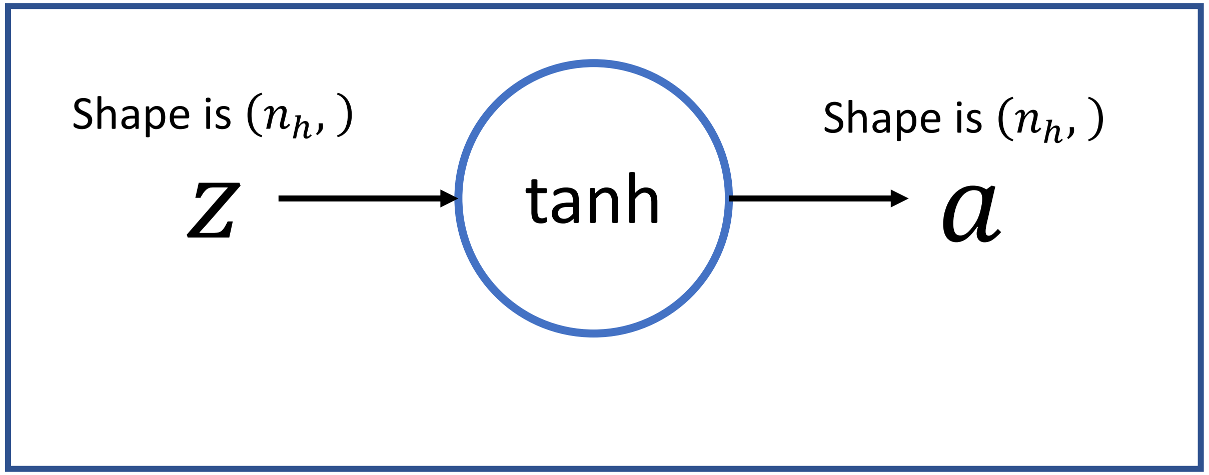
```
# Assuming a and dJ_da are numpy vectors
dJ_dz = np.multiply(1-np.square(a), dJ_da)
```

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2$$

$$\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}$$



Forward Propagation

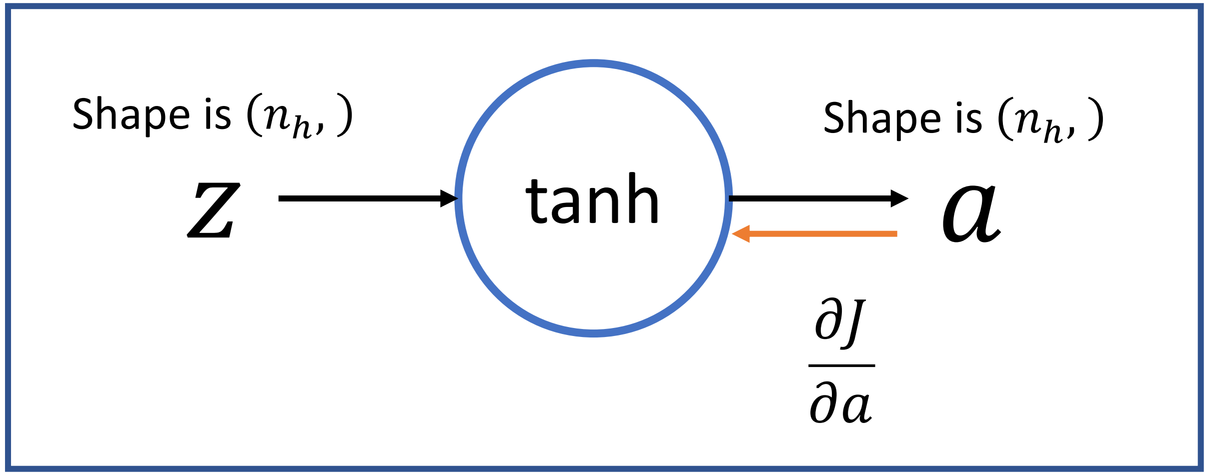
$$z = \begin{bmatrix} 0.8 \\ 1 \\ -0.5 \\ -0.1 \end{bmatrix} \qquad a = \begin{bmatrix} \tanh(0.8) \\ \tanh(1) \\ \tanh(-0.5) \\ \tanh(-0.1) \end{bmatrix} = \begin{bmatrix} 0.66 \\ 0.76 \\ -0.46 \\ -0.10 \end{bmatrix}$$

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2$$

$$\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}$$



Forward Propagation

$$z = \begin{bmatrix} 0.8 \\ 1 \\ -0.5 \\ -0.1 \end{bmatrix} \qquad a = \begin{bmatrix} \tanh(0.8) \\ \tanh(1) \\ \tanh(-0.5) \\ \tanh(-0.1) \end{bmatrix} = \begin{bmatrix} 0.66 \\ 0.76 \\ -0.46 \\ -0.10 \end{bmatrix}$$

Back Propagation

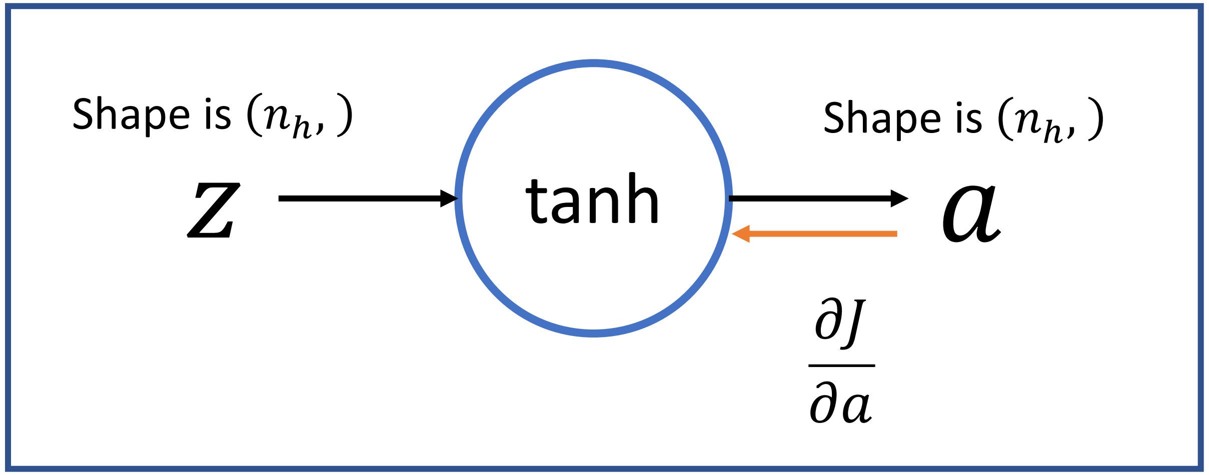
$$\frac{\partial J}{\partial a} = \begin{bmatrix} -0.2 \\ 0.5 \\ -0.3 \\ -0.6 \end{bmatrix}$$

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\boxed{\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2}$$

$$\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}$$



Shape is $(n_h,)$

z

tanh

Shape is $(n_h,)$

a

$\frac{\partial J}{\partial a}$

Forward Propagation

$$z = \begin{bmatrix} 0.8 \\ 1 \\ -0.5 \\ -0.1 \end{bmatrix} \qquad a = \begin{bmatrix} \tanh(0.8) \\ \tanh(1) \\ \tanh(-0.5) \\ \tanh(-0.1) \end{bmatrix} = \begin{bmatrix} 0.66 \\ 0.76 \\ -0.46 \\ -0.10 \end{bmatrix}$$

Back Propagation

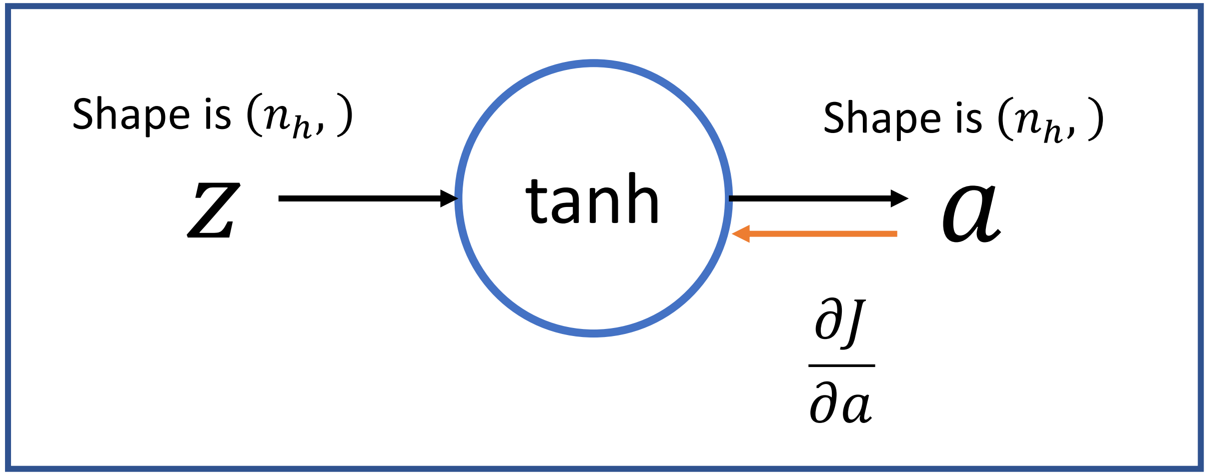
$$\frac{\partial J}{\partial a} = \begin{bmatrix} -0.2 \\ 0.5 \\ -0.3 \\ -0.6 \end{bmatrix} \qquad \frac{\partial a}{\partial z} = \begin{bmatrix} 1 - 0.66^2 \\ 1 - 0.76^2 \\ 1 + 0.46^2 \\ 1 + 0.10^2 \end{bmatrix} = \begin{bmatrix} 0.56 \\ 0.42 \\ 0.79 \\ 0.99 \end{bmatrix}$$

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\boxed{\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2}$$

$$\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}$$



Forward Propagation

$$z = \begin{bmatrix} 0.8 \\ 1 \\ -0.5 \\ -0.1 \end{bmatrix} \qquad a = \begin{bmatrix} \tanh(0.8) \\ \tanh(1) \\ \tanh(-0.5) \\ \tanh(-0.1) \end{bmatrix} = \begin{bmatrix} 0.66 \\ 0.76 \\ -0.46 \\ -0.10 \end{bmatrix}$$

$$\left(\frac{\partial a}{\partial z}\right)_{full} = \begin{bmatrix} 0.56 & 0 & 0 & 0 \\ 0 & 0.42 & 0 & 0 \\ 0 & 0 & 0.79 & 0 \\ 0 & 0 & 0 & 0.99 \end{bmatrix}$$

Back Propagation

$$\frac{\partial J}{\partial a} = \begin{bmatrix} -0.2 \\ 0.5 \\ -0.3 \\ -0.6 \end{bmatrix} \qquad \frac{\partial a}{\partial z} = \begin{bmatrix} 1 - 0.66^2 \\ 1 - 0.76^2 \\ 1 + 0.46^2 \\ 1 + 0.10^2 \end{bmatrix} = \begin{bmatrix} 0.56 \\ 0.42 \\ 0.79 \\ 0.99 \end{bmatrix}$$

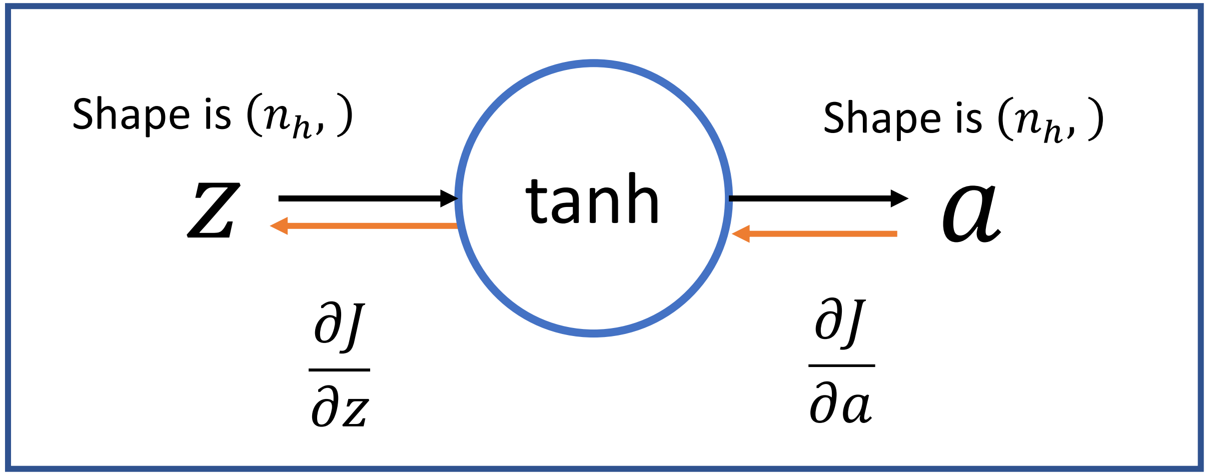
Not the full Jacobian! Just the diagonal!

Example: Tanh Activation Function on Layer for one sample

$$a_i = \tanh(z_i)$$

$$\frac{\partial a_i}{\partial z_i} = 1 - \tanh^2(z_i) = 1 - a_i^2$$

$$\boxed{\frac{\partial J}{\partial z_i} = \frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i}}$$


Shape is $(n_h,)$
z
tanh
Shape is $(n_h,)$
a
$\frac{\partial J}{\partial z}$
$\frac{\partial J}{\partial a}$

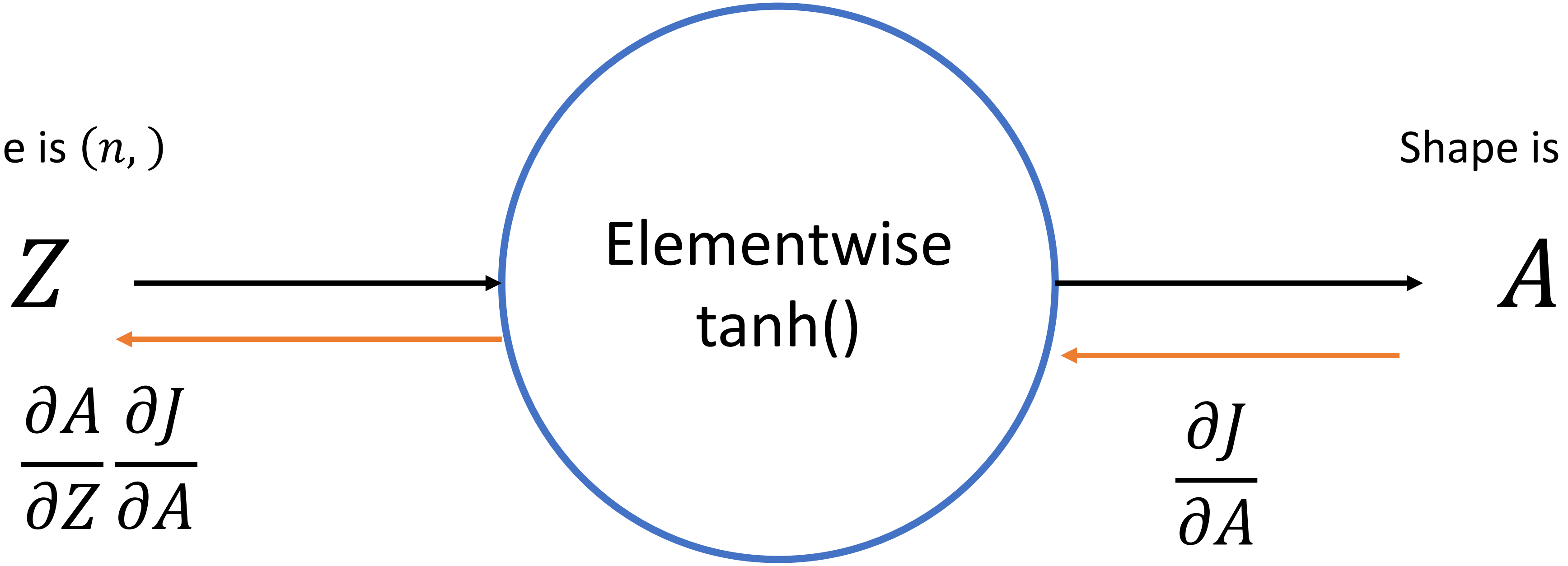
Forward Propagation

$$z = \begin{bmatrix} 0.8 \\ 1 \\ -0.5 \\ -0.1 \end{bmatrix} \qquad a = \begin{bmatrix} \tanh(0.8) \\ \tanh(1) \\ \tanh(-0.5) \\ \tanh(-0.1) \end{bmatrix} = \begin{bmatrix} 0.66 \\ 0.76 \\ -0.46 \\ -0.10 \end{bmatrix}$$

Back Propagation

$$\frac{\partial J}{\partial a} = \begin{bmatrix} -0.2 \\ 0.5 \\ -0.3 \\ -0.6 \end{bmatrix} \qquad \frac{\partial a}{\partial z} = \begin{bmatrix} 1 - 0.66^2 \\ 1 - 0.76^2 \\ 1 + 0.46^2 \\ 1 + 0.10^2 \end{bmatrix} = \begin{bmatrix} 0.56 \\ 0.42 \\ 0.79 \\ 0.99 \end{bmatrix} \qquad \frac{\partial J}{\partial z} = \begin{bmatrix} 0.56 * -0.2 \\ 0.42 * 0.5 \\ 0.79 * -0.3 \\ 0.99 * -0.6 \end{bmatrix} = \begin{bmatrix} -0.11 \\ 0.21 \\ -0.24 \\ -0.59 \end{bmatrix}$$

Example: Tanh Activation Function on Layer for one sample



Shape is $(n,)$

Shape is $(n,)$

$$\frac{\partial J}{\partial Z} = \frac{\partial A}{\partial Z}\frac{\partial J}{\partial A}$$

$$\frac{\partial J}{\partial A}$$

Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & 0 & \cdots & 0 \\ 0 & \frac{\partial a_2}{\partial z_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

Remember:

a_1= $\tanh(z_1)$

a_2= $\tanh(z_2)$

$\vdots$

a_n= $\tanh(z_n)$

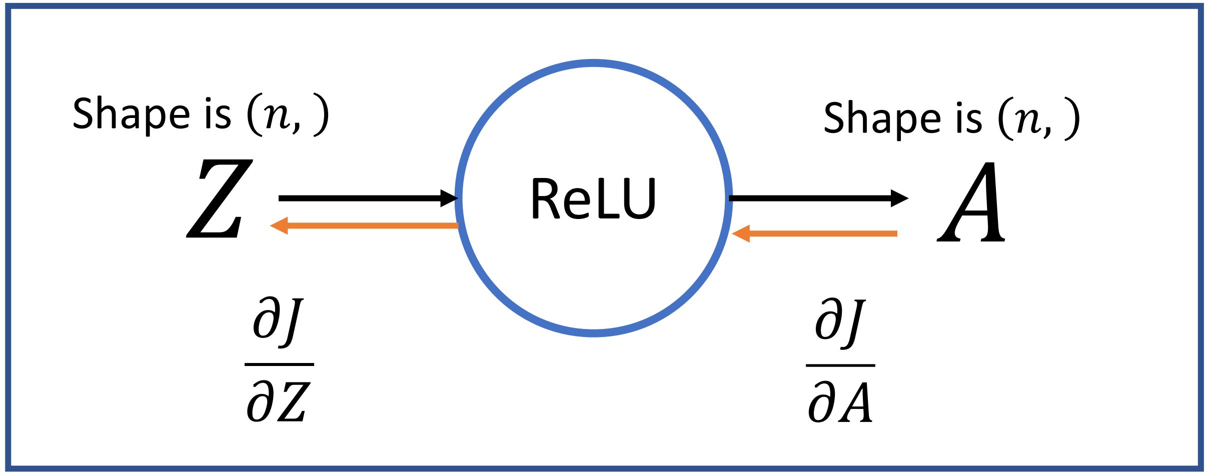
Summary

- Elementwise operation
- Therefore only non-zero values are along diagonal
- We don't need to construct the full Jacobian

Example ReLU

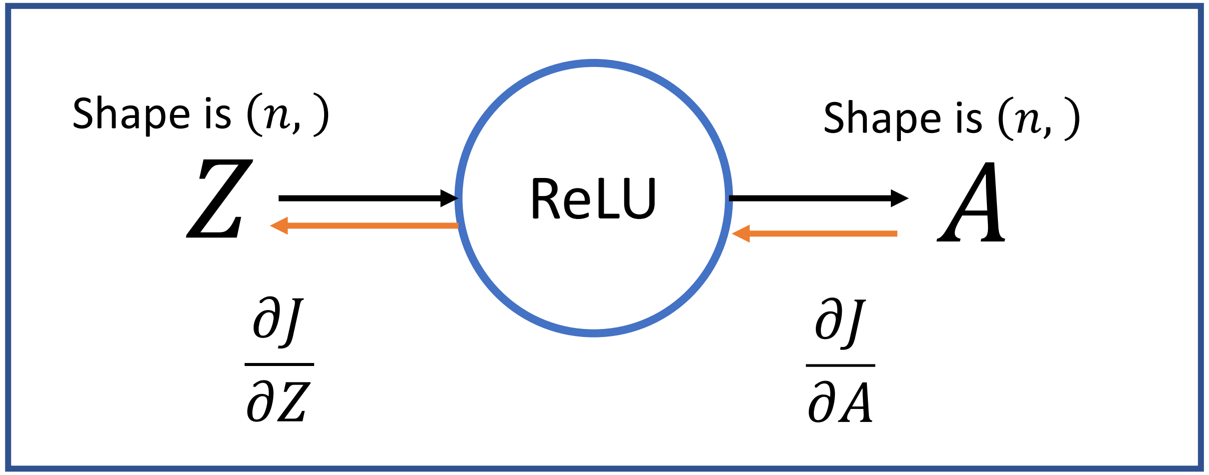
Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$

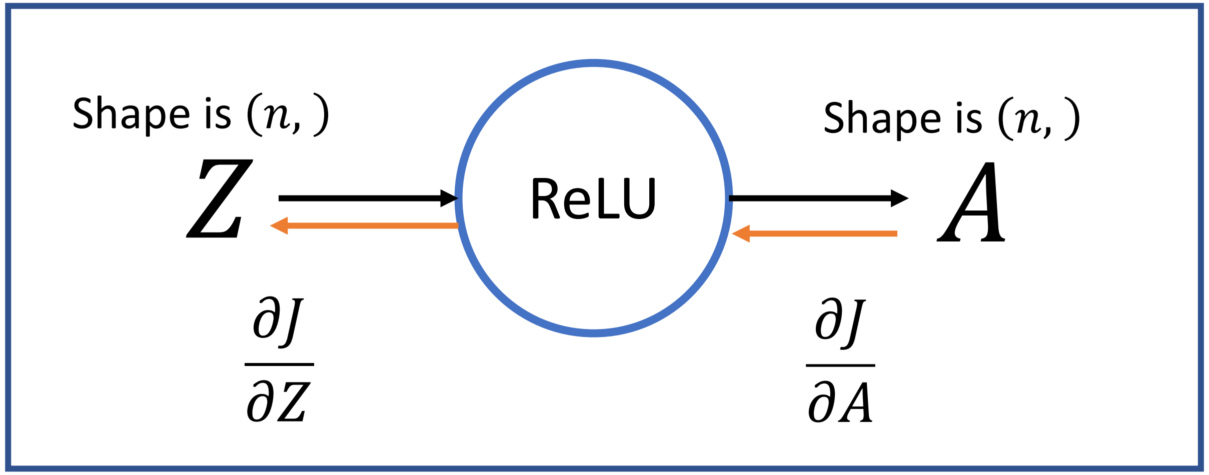

Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_n}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_n}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_n} & \frac{\partial a_2}{\partial z_n} & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

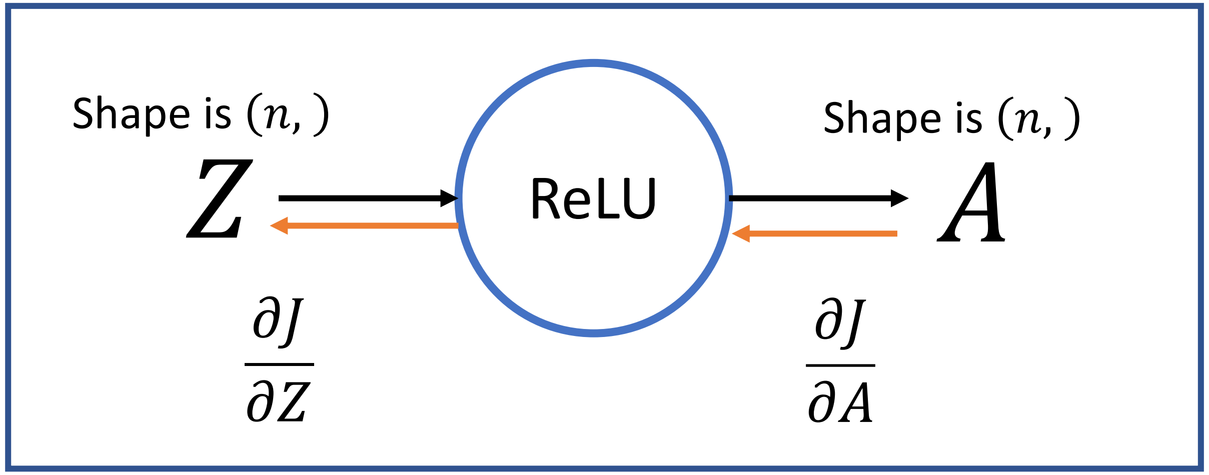
Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & \frac{\partial a_2}{\partial z_1} & \cdots & \frac{\partial a_n}{\partial z_1} \\ \frac{\partial a_1}{\partial z_2} & \frac{\partial a_2}{\partial z_2} & & \frac{\partial a_n}{\partial z_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_1}{\partial z_n} & \frac{\partial a_2}{\partial z_n} & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

$$\frac{\partial J}{\partial Z} = \frac{\partial A}{\partial Z}\frac{\partial J}{\partial A}$$

Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$



Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Local Gradients / Jacobian Matrix

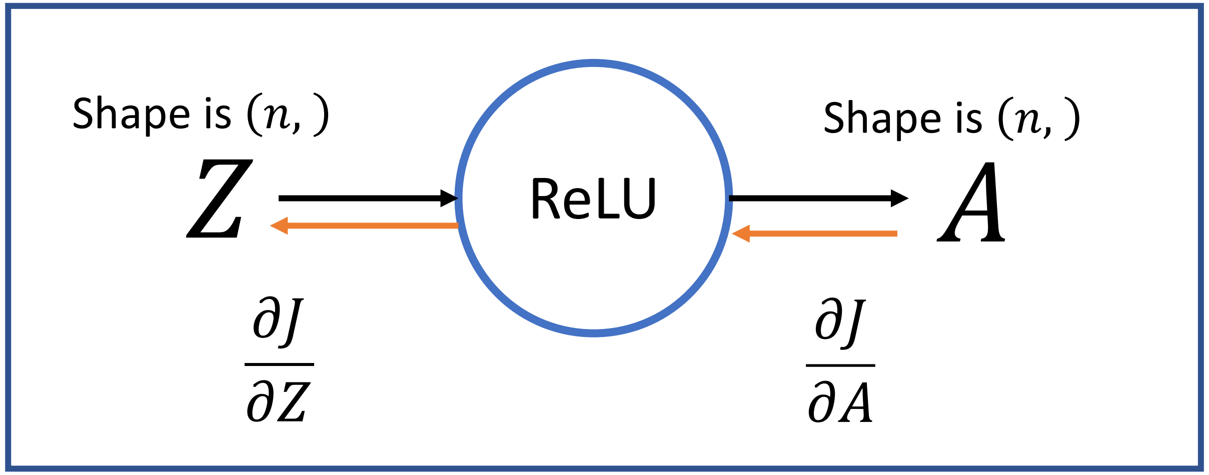
$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & 0 & \cdots & 0 \\ 0 & \frac{\partial a_2}{\partial z_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

Since this is an elementwise vector operation, We know that only the diagonal of the Jacobian will be nonzero

$$\frac{\partial J}{\partial Z} = \frac{\partial A}{\partial Z}\frac{\partial J}{\partial A}$$

Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Local Gradients / Jacobian Matrix

$$\frac{\partial A}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} & 0 & \dots & 0 \\ 0 & \frac{\partial a_2}{\partial z_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & \frac{\partial a_n}{\partial z_n} \end{bmatrix}$$

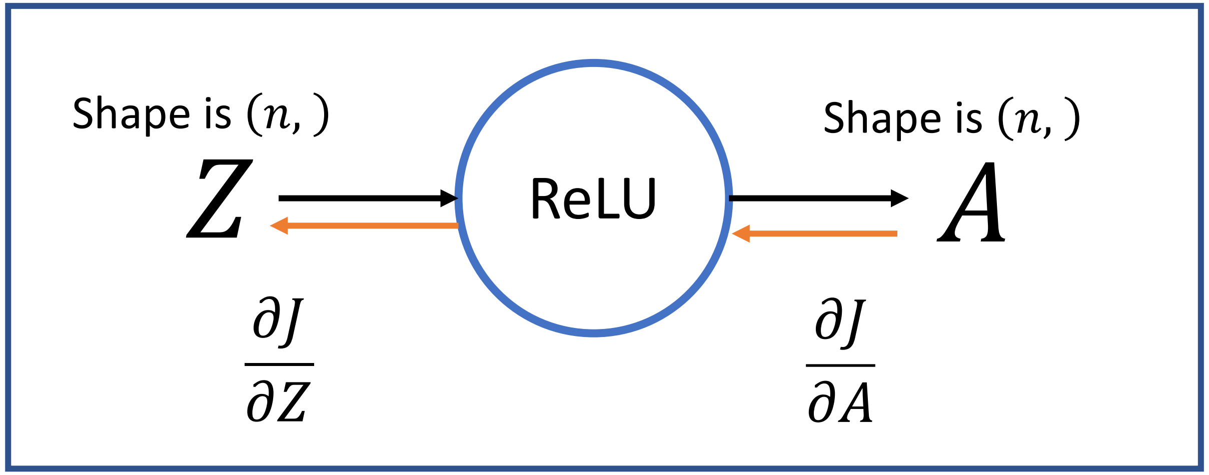
$$\frac{\partial J}{\partial Z} = \frac{\partial A}{\partial Z} \frac{\partial J}{\partial A}$$

Simplifies to this

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} * \frac{\partial J}{\partial a_1} \\ \frac{\partial a_2}{\partial z_2} * \frac{\partial J}{\partial a_2} \\ \vdots \\ \frac{\partial a_n}{\partial z_n} * \frac{\partial J}{\partial a_n} \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample

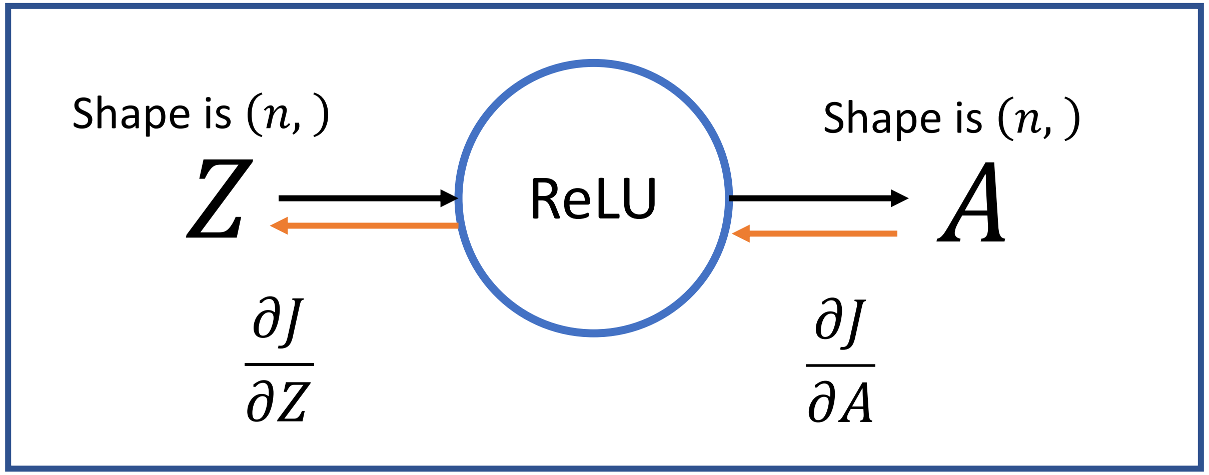
$$A = \text{relu}(Z) = \begin{bmatrix} \text{relu}(z_1) \\ \text{relu}(z_2) \\ \text{relu}(z_3) \\ \vdots \\ \text{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} * \frac{\partial J}{\partial a_1} \\ \frac{\partial a_2}{\partial z_2} * \frac{\partial J}{\partial a_2} \\ \vdots \\ \frac{\partial a_n}{\partial z_n} * \frac{\partial J}{\partial a_n} \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$



Shape is $(n,)$

Z

ReLU

Shape is $(n,)$

A

$\frac{\partial J}{\partial Z}$

$\frac{\partial J}{\partial A}$

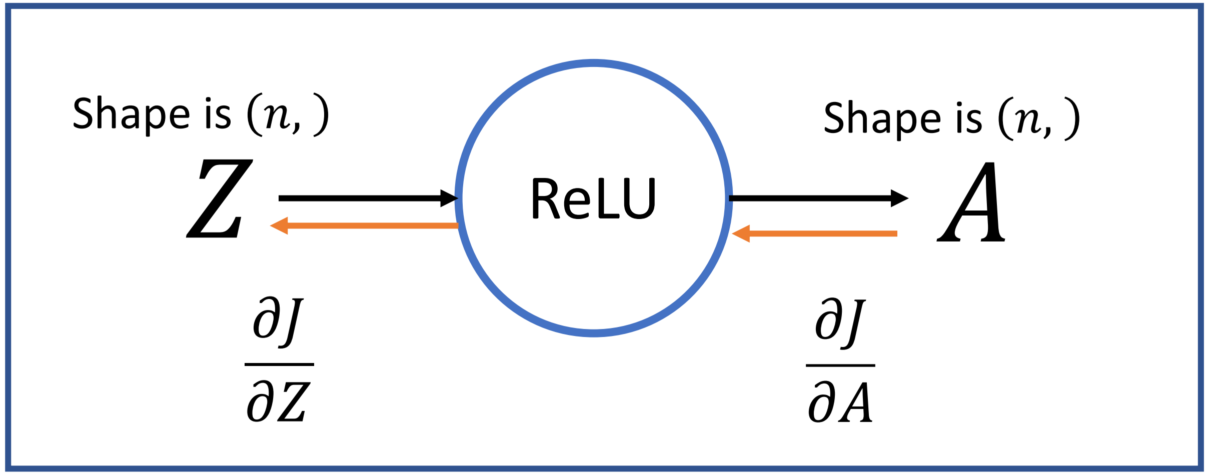
$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial a_i}{\partial z_i} = \begin{cases} 1, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} \ast \frac{\partial J}{\partial a_1} \\ \frac{\partial a_2}{\partial z_2} \ast \frac{\partial J}{\partial a_2} \\ \vdots \\ \frac{\partial a_n}{\partial z_n} \ast \frac{\partial J}{\partial a_n} \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$



Shape is $(n,)$

Z

ReLU

Shape is $(n,)$

A

$\frac{\partial J}{\partial Z}$

$\frac{\partial J}{\partial A}$

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

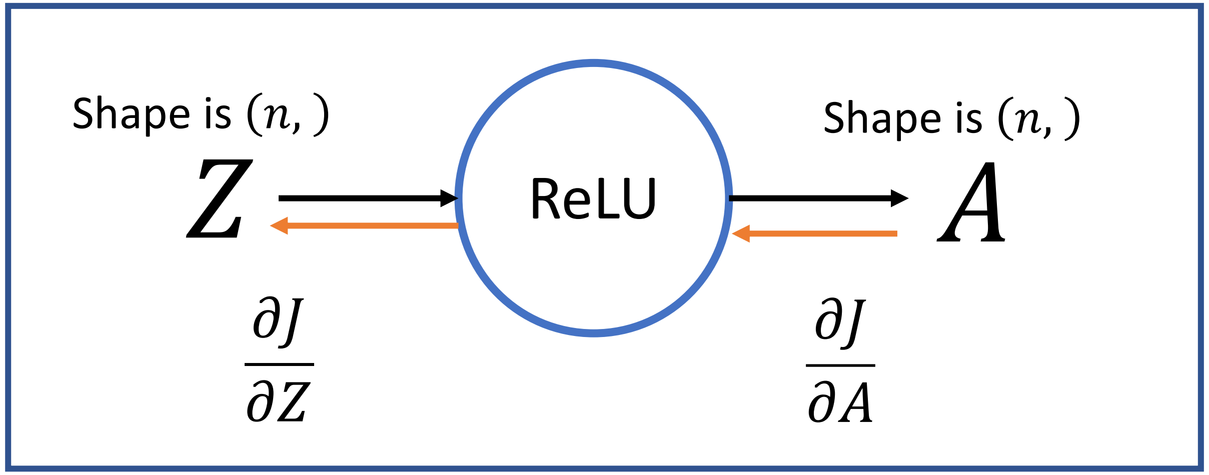
$$\frac{\partial a_i}{\partial z_i} = \begin{cases} 1, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} * \frac{\partial J}{\partial a_1} \\ \frac{\partial a_2}{\partial z_2} * \frac{\partial J}{\partial a_2} \\ \vdots \\ \frac{\partial a_n}{\partial z_n} * \frac{\partial J}{\partial a_n} \end{bmatrix}$$

$$\frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i} = \begin{cases} 1 * \frac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0 * \frac{\partial J}{\partial a_i}, & z_i < 0 \end{cases}$$

Example: ReLU Activation Function on Layer for one sample

$$A = \mathrm{relu}(Z) = \begin{bmatrix} \mathrm{relu}(z_1) \\ \mathrm{relu}(z_2) \\ \mathrm{relu}(z_3) \\ \vdots \\ \mathrm{relu}(z_n) \end{bmatrix} = \begin{bmatrix} \max(z_1, 0) \\ \max(z_2, 0) \\ \max(z_3, 0) \\ \vdots \\ \max(z_n, 0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}$$



Shape is $(n,)$

Shape is $(n,)$

Z

ReLU

A

$\frac{\partial J}{\partial Z}$

$\frac{\partial J}{\partial A}$

Simply copy over upstream gradient or set to 0

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial a_i}{\partial z_i} = \begin{cases} 1, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \frac{\partial a_1}{\partial z_1} * \frac{\partial J}{\partial a_1} \\ \frac{\partial a_2}{\partial z_2} * \frac{\partial J}{\partial a_2} \\ \vdots \\ \frac{\partial a_n}{\partial z_n} * \frac{\partial J}{\partial a_n} \end{bmatrix}$$

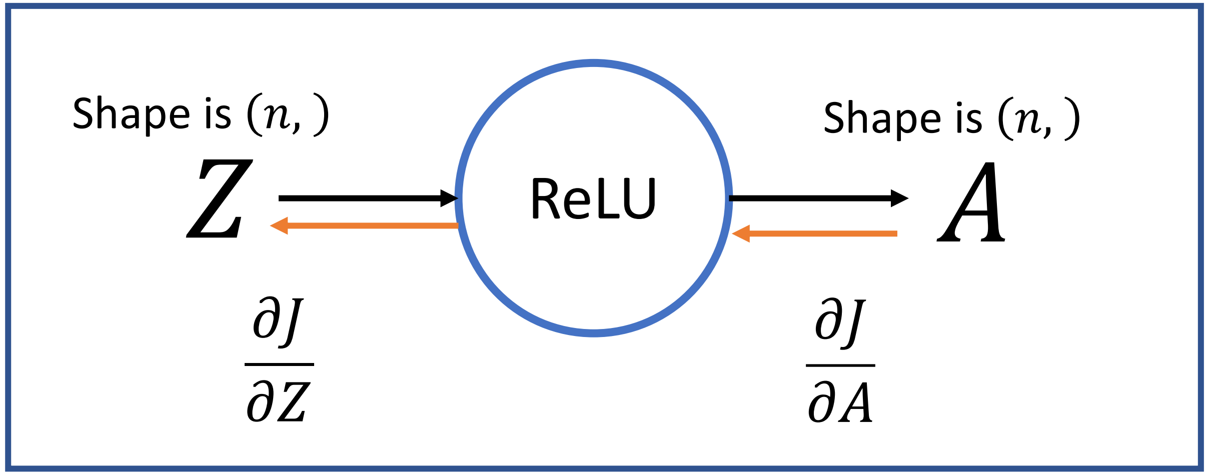
$$\frac{\partial a_i}{\partial z_i} * \frac{\partial J}{\partial a_i} = \begin{cases} \frac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

Extremely efficient **implicit** Jacobian matrix-vector multiply!

Example: ReLU Activation Function on Layer for one sample with Numbers

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

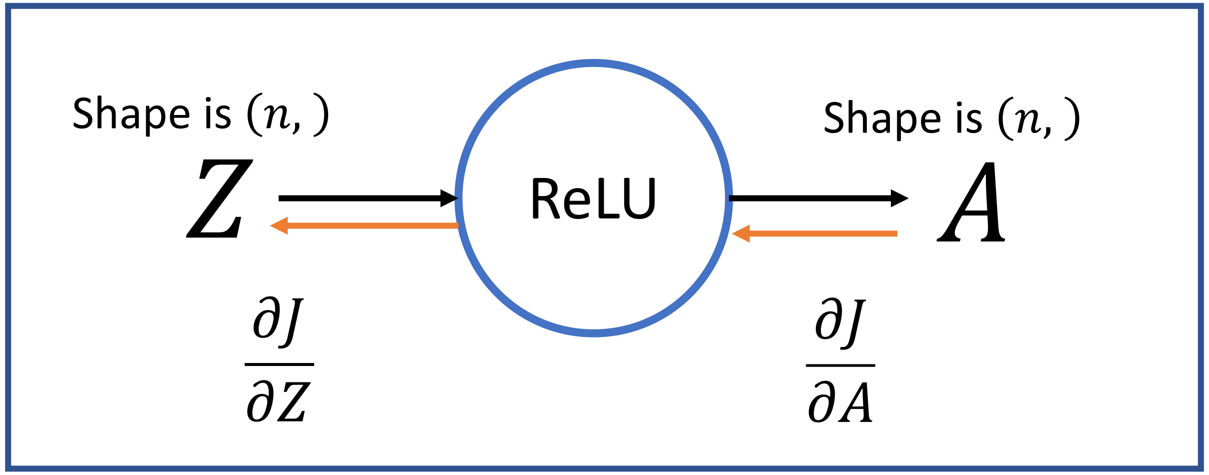
$$\frac{\partial J}{\partial z_i} = \begin{cases} \dfrac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Example: ReLU Activation Function on Layer for one sample with Numbers

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial z_i} = \begin{cases} \dfrac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$



Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

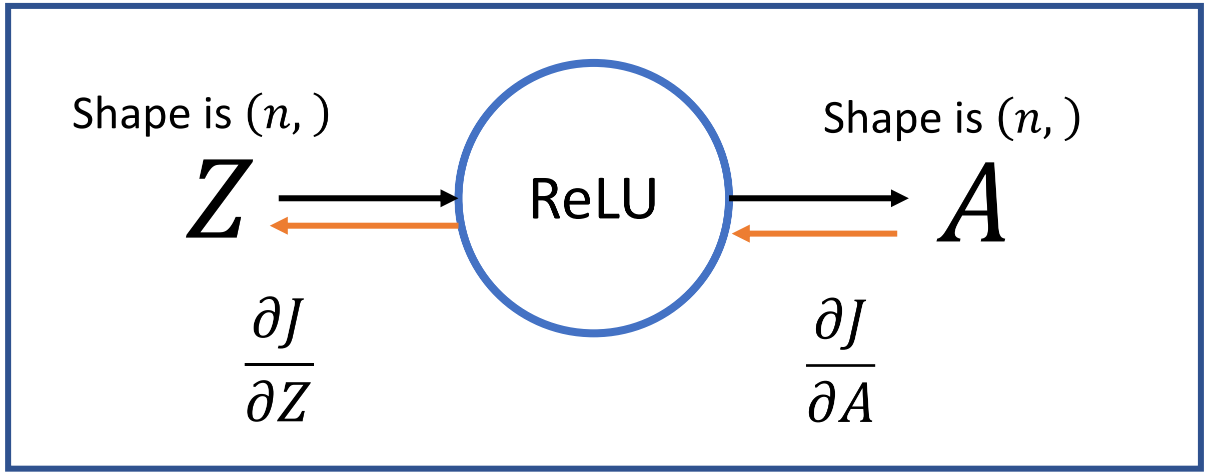
Forward Propagation

$$Z = \begin{bmatrix} 0.8 \\ 1.2 \\ -0.5 \\ 0.1 \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample with Numbers

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial z_i} = \begin{cases} \dfrac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

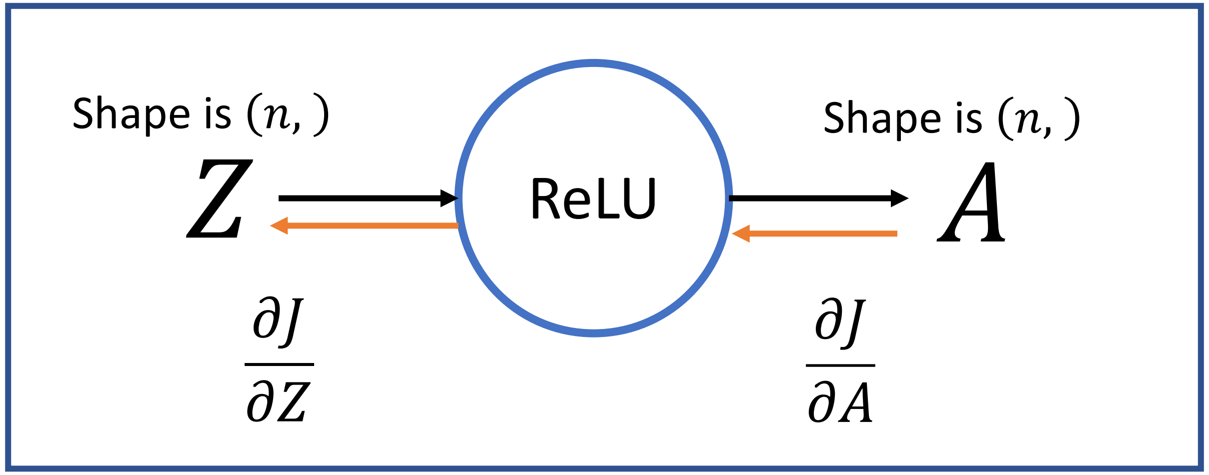
Forward Propagation

$$Z = \begin{bmatrix} 0.8 \\ 1.2 \\ -0.5 \\ 0.1 \end{bmatrix} \qquad A = \begin{bmatrix} 0.8 \\ 1.2 \\ 0 \\ 0.1 \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample with Numbers

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial z_i} = \begin{cases} \dfrac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Forward Propagation

$$Z = \begin{bmatrix} 0.8 \\ 1.2 \\ -0.5 \\ 0.1 \end{bmatrix} \qquad A = \begin{bmatrix} 0.8 \\ 1.2 \\ 0 \\ 0.1 \end{bmatrix}$$

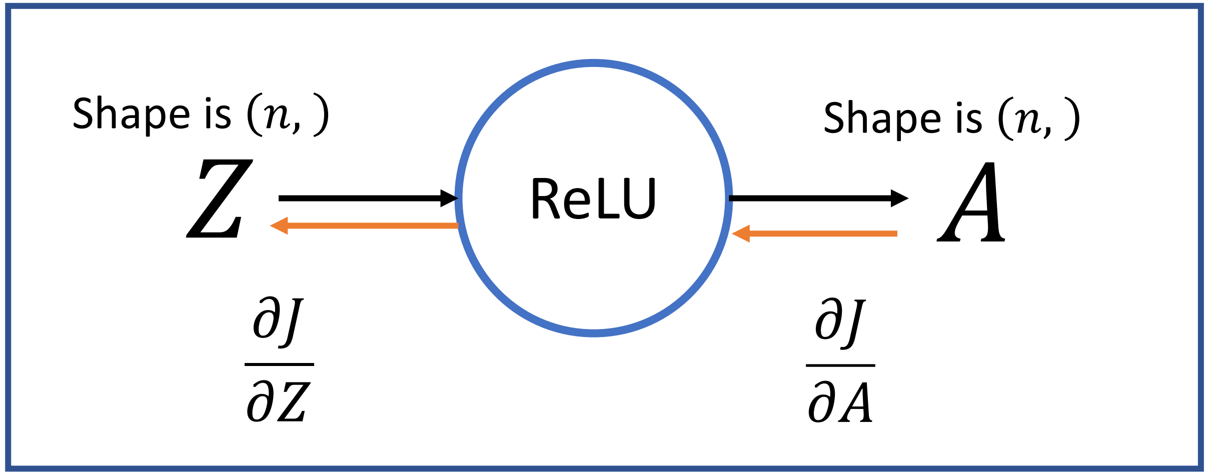
Back Propagation

$$\frac{\partial J}{\partial A} = \begin{bmatrix} -0.2 \\ 0.5 \\ -0.3 \\ -0.6 \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample with Numbers

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial z_i} = \begin{cases} \frac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$


Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Forward Propagation

$$Z = \begin{bmatrix} 0.8 \\ 1.2 \\ -0.5 \\ 0.1 \end{bmatrix} \qquad A = \begin{bmatrix} 0.8 \\ 1.2 \\ 0 \\ 0.1 \end{bmatrix}$$

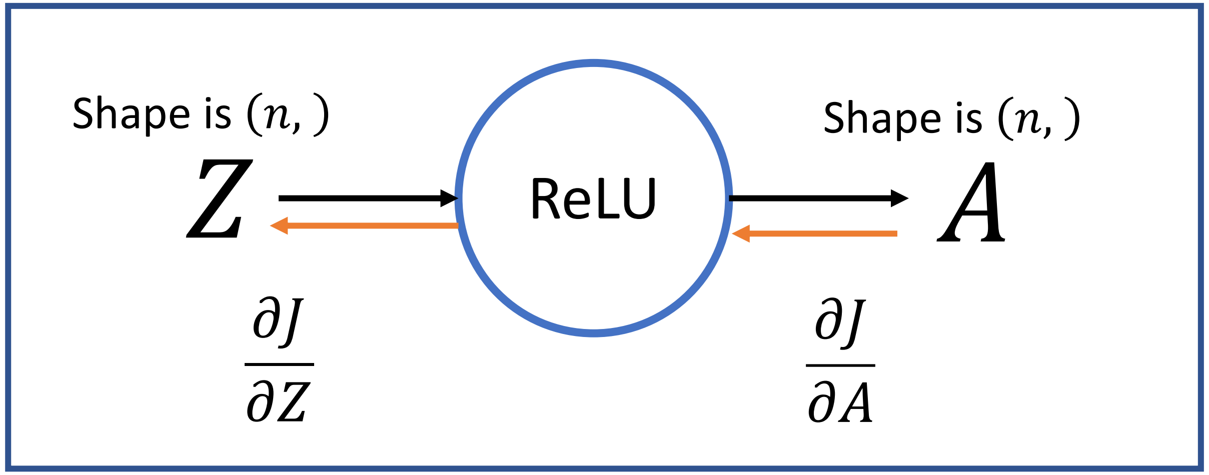
Back Propagation

$$\frac{\partial J}{\partial A} = \begin{bmatrix} -0.2 \\ 0.5 \\ -0.3 \\ -0.6 \end{bmatrix} \qquad \frac{\partial J}{\partial Z} = \begin{bmatrix} -0.2 \\ 0.5 \\ 0 \\ -0.6 \end{bmatrix}$$

Example: ReLU Activation Function on Layer for one sample with Numbers

$$a_i = \max(0, z_i) = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$

$$\frac{\partial J}{\partial z_i} = \begin{cases} \frac{\partial J}{\partial a_i}, & z_i \geq 0 \\ 0, & z_i < 0 \end{cases}$$



Shape is $(n,)$
Z
ReLU
Shape is $(n,)$
A
$\frac{\partial J}{\partial Z}$
$\frac{\partial J}{\partial A}$

Forward Propagation

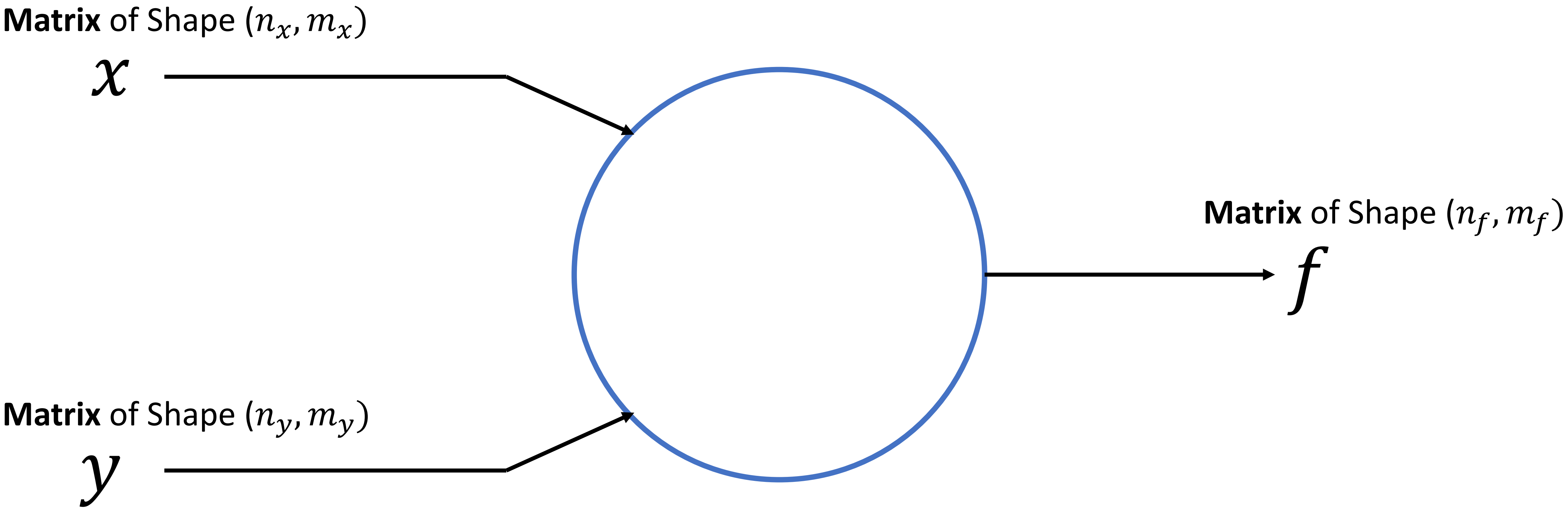
$$Z = \begin{bmatrix} 0.8 \\ 1.2 \\ -0.5 \\ 0.1 \end{bmatrix} \qquad A = \begin{bmatrix} 0.8 \\ 1.2 \\ 0 \\ 0.1 \end{bmatrix}$$

Back Propagation

Full Jacobian not needed!

$$\frac{\partial J}{\partial A} = \begin{bmatrix} -0.2 \\ 0.5 \\ -0.3 \\ -0.6 \end{bmatrix} \qquad \frac{\partial J}{\partial Z} = \begin{bmatrix} -0.2 \\ 0.5 \\ 0 \\ -0.6 \end{bmatrix} \qquad \frac{\partial A}{\partial Z} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Backpropagation with Matrices



Matrix of Shape (n_x, m_x)

x

Matrix of Shape (n_y, m_y)

y

Matrix of Shape (n_f, m_f)

f

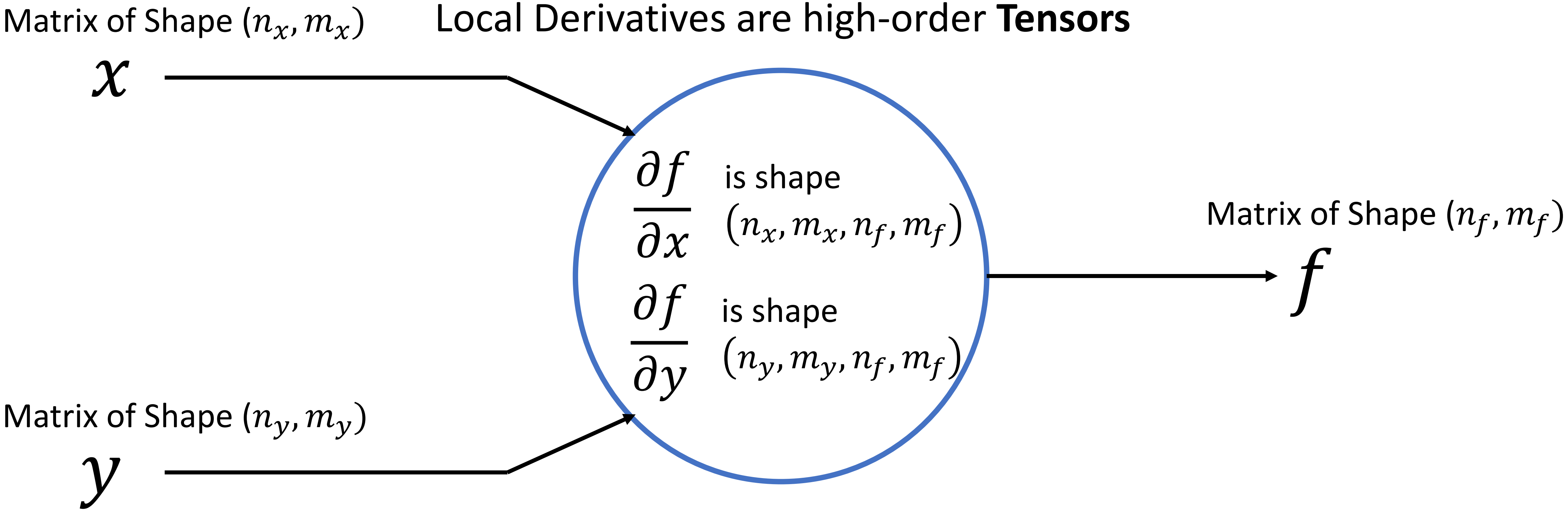
Tensors

- For our purposes, Tensors are multidimensional arrays.

Examples with special names:

- A scalar is a 0d Tensor
- A vector is a 1d Tensor
- A matrix is a 2d tensor

Local Derivatives are high-order **Tensors**



Matrix of Shape (n_x, m_x)

x

$\frac{\partial f}{\partial x}$ is shape (n_x, m_x, n_f, m_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, m_y, n_f, m_f)

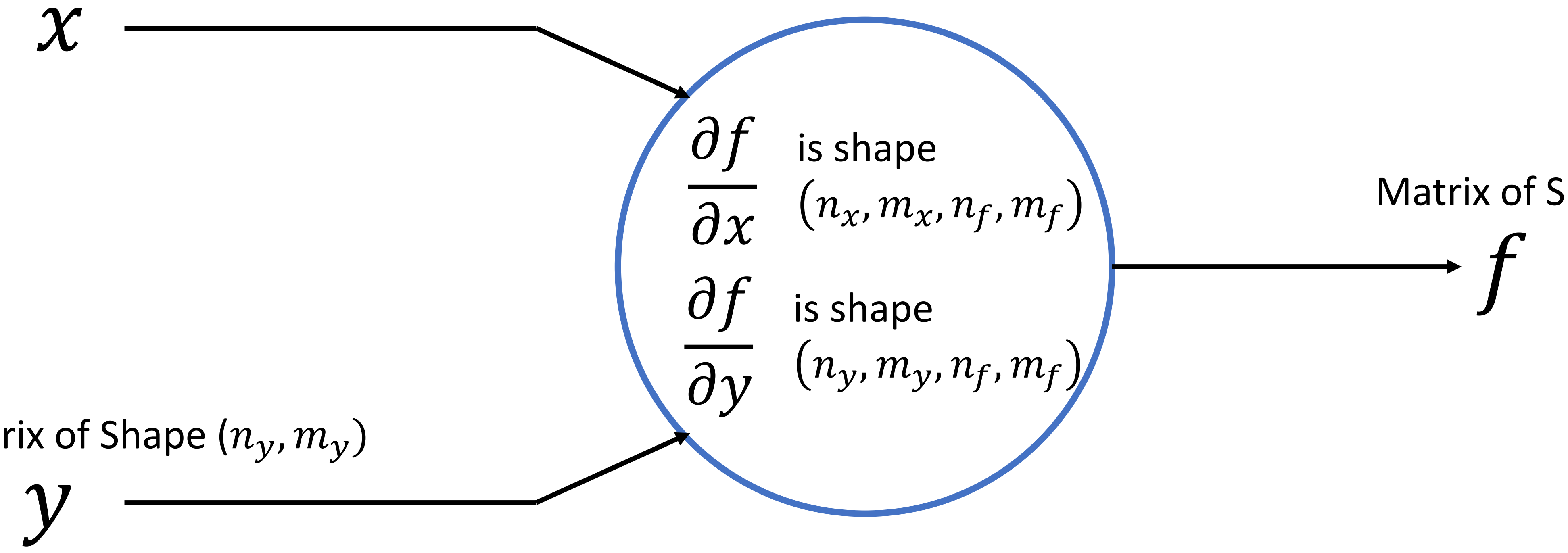
Matrix of Shape (n_f, m_f)

f

Matrix of Shape (n_y, m_y)

y

Local Derivatives are high-order **Tensors**



Matrix of Shape (n_x, m_x)

x

$\frac{\partial f}{\partial x}$ is shape (n_x, m_x, n_f, m_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, m_y, n_f, m_f)

Matrix of Shape (n_f, m_f)

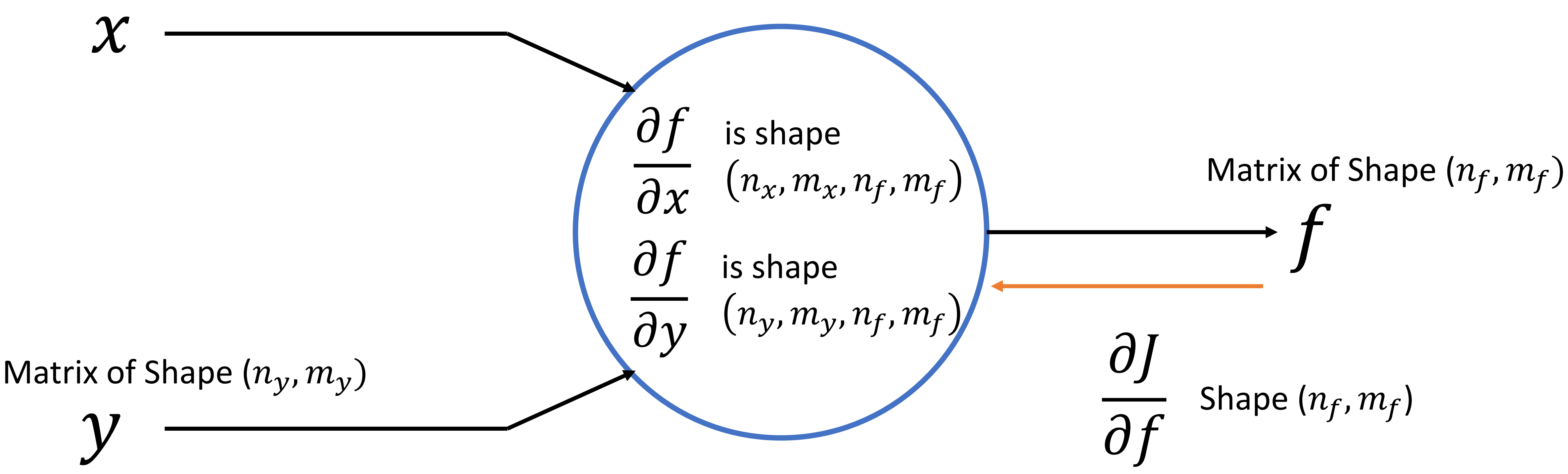
f

Matrix of Shape (n_y, m_y)

y

$\frac{\partial f}{\partial x}$ tells you how each of the (n_x, m_x) inputs affects each of the (n_f, m_f) outputs

Local Derivatives are high-order **Tensors**



Matrix of Shape (n_x, m_x)

x

Matrix of Shape (n_y, m_y)

y

$\frac{\partial f}{\partial x}$ is shape (n_x, m_x, n_f, m_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, m_y, n_f, m_f)

Matrix of Shape (n_f, m_f)

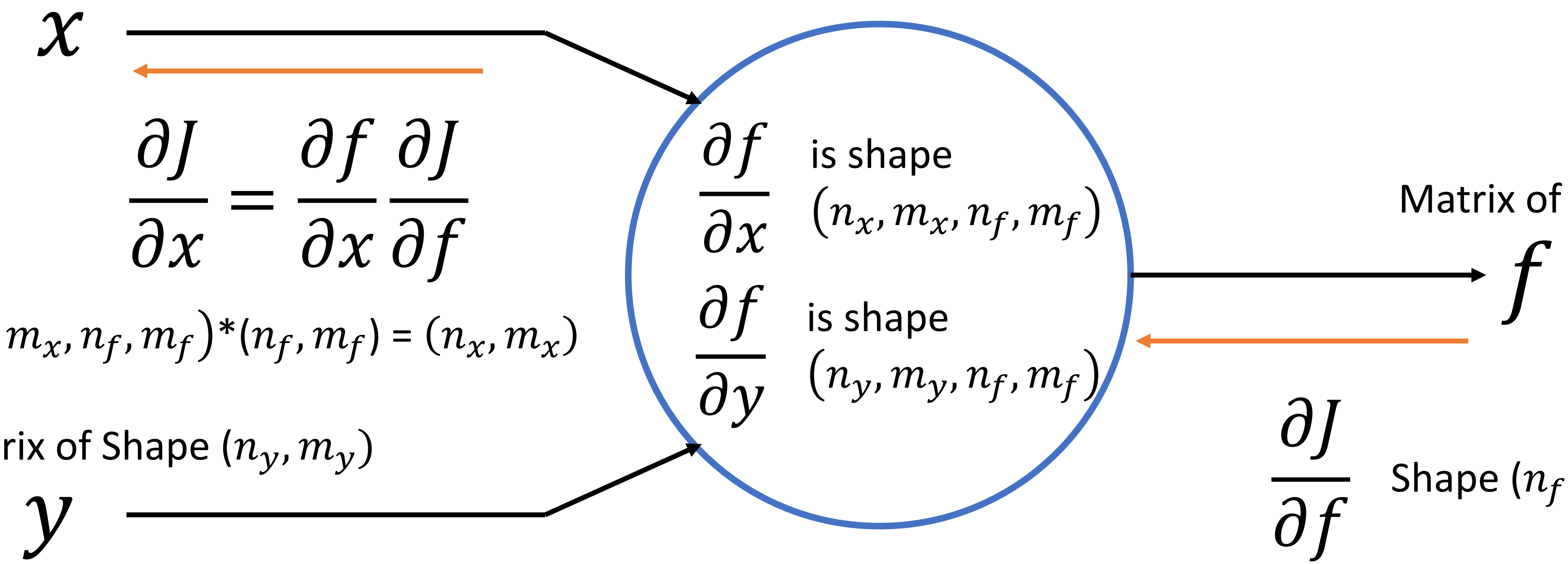
f

$\frac{\partial J}{\partial f}$ Shape (n_f, m_f)

Upstream gradient is with respect to Cost J (a scalar)

i.e. How does each of the (n_f, m_f) outputs affect the Cost

Local Derivatives are high-order **Tensors**



Matrix of Shape (n_x, m_x)

x

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f}$$

(n_x, m_x, n_f, m_f)*(n_f, m_f) = (n_x, m_x)

Matrix of Shape (n_y, m_y)

y

$\frac{\partial f}{\partial x}$ is shape (n_x, m_x, n_f, m_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, m_y, n_f, m_f)

Matrix of Shape (n_f, m_f)

f

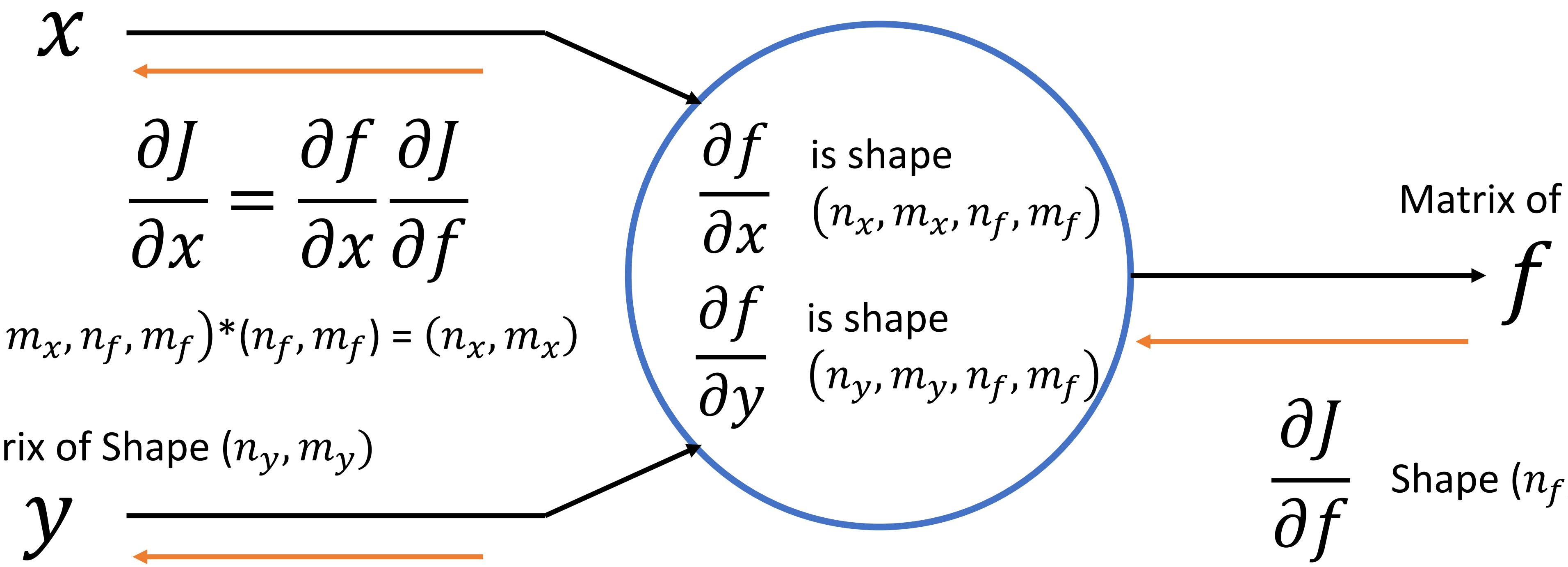
$\frac{\partial J}{\partial f}$ Shape (n_f, m_f)

Upstream gradient is with respect to Cost J (a scalar)

i.e. How does each of the (n_f, m_f) outputs affect the Cost

Chain Rule application is **Tensor-Matrix Multiply**

Local Derivatives are high-order **Tensors**



Matrix of Shape (n_x, m_x)

$$\frac{\partial J}{\partial x} = \frac{\partial f}{\partial x}\frac{\partial J}{\partial f}$$

(n_x, m_x, n_f, m_f)*(n_f, m_f) = (n_x, m_x)

$\frac{\partial f}{\partial x}$ is shape (n_x, m_x, n_f, m_f)

$\frac{\partial f}{\partial y}$ is shape (n_y, m_y, n_f, m_f)

Matrix of Shape (n_f, m_f)

$\frac{\partial J}{\partial f}$ Shape (n_f, m_f)

Matrix of Shape (n_y, m_y)

$$\frac{\partial J}{\partial y} = \frac{\partial f}{\partial y}\frac{\partial J}{\partial f}$$

(n_y, m_y, n_f, m_f)*(n_f, m_f) = (n_y, m_y)

Upstream gradient is with respect to Cost J (a scalar)

i.e. How does each of the (n_f, m_f) outputs affect the Cost

Chain Rule application is **Tensor-Matrix Multiply**

Example: Matrix Multiplcation

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

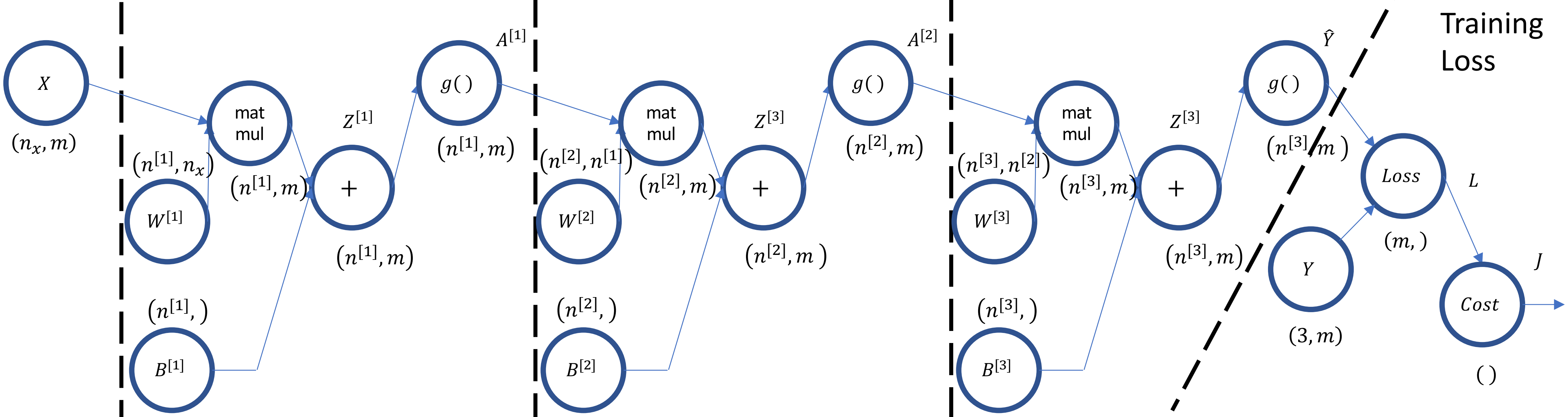
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

$n^{[1]} = 5$

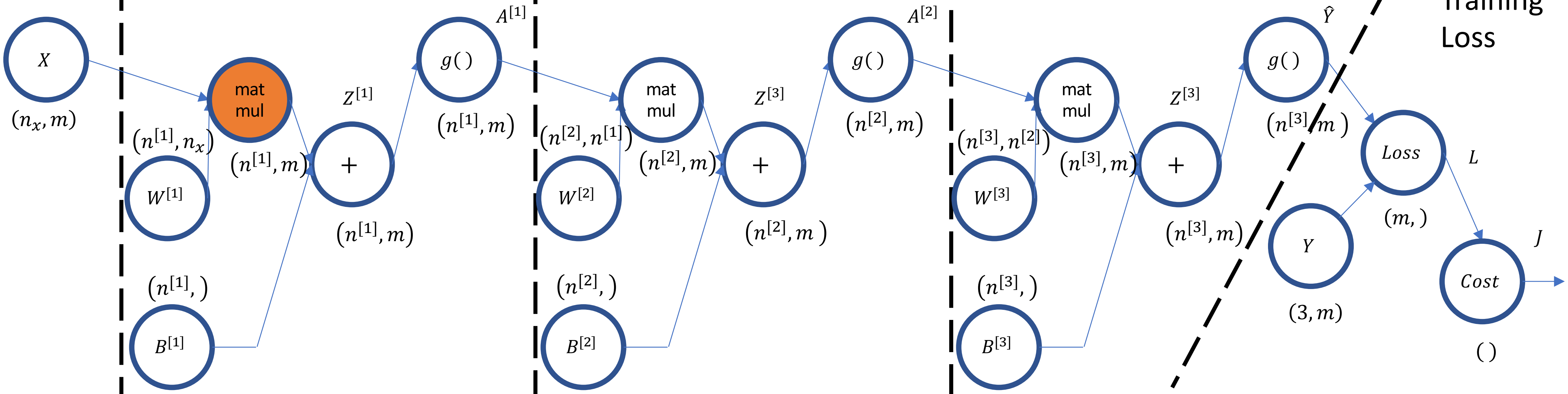
Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$

Training Loss



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

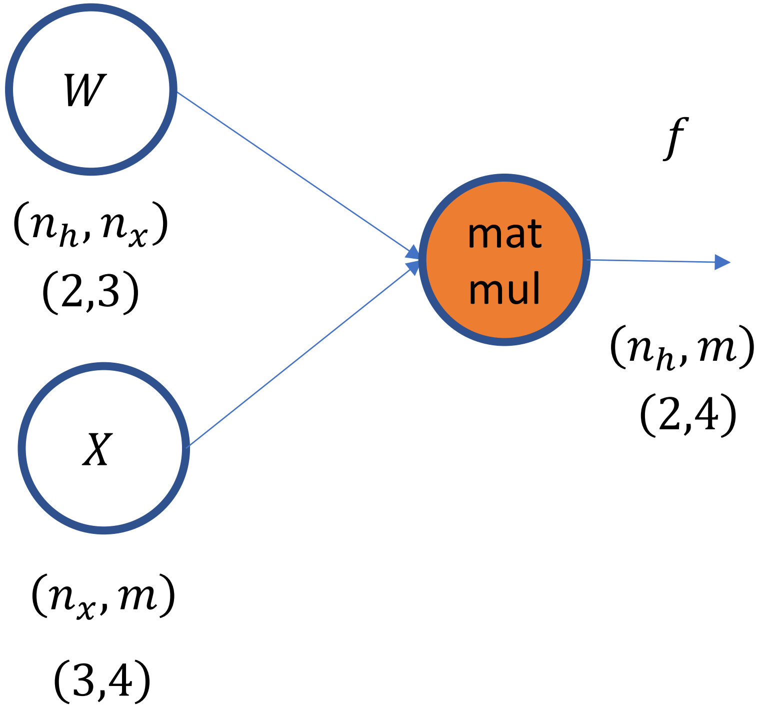
$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

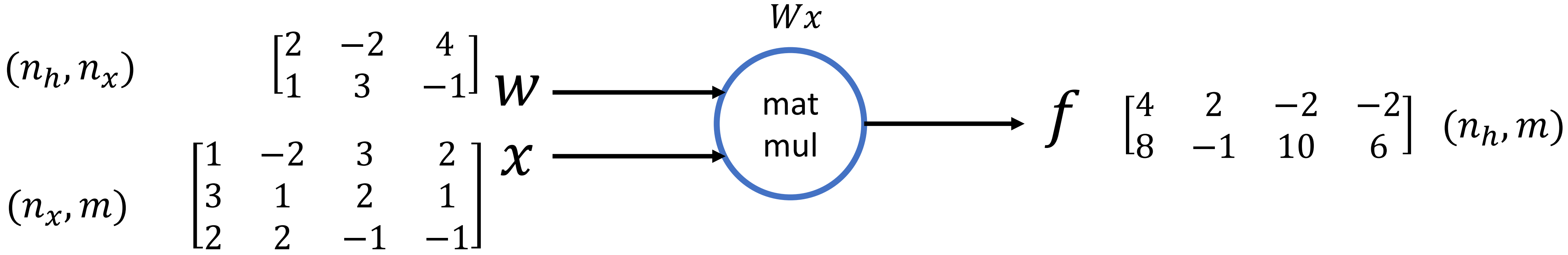
$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

Example: Matrix Multiplcation



Example:

$$n_x = 3$$
$$n_h = 2$$
$$m = 4$$



(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix}$ W

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix}$ x

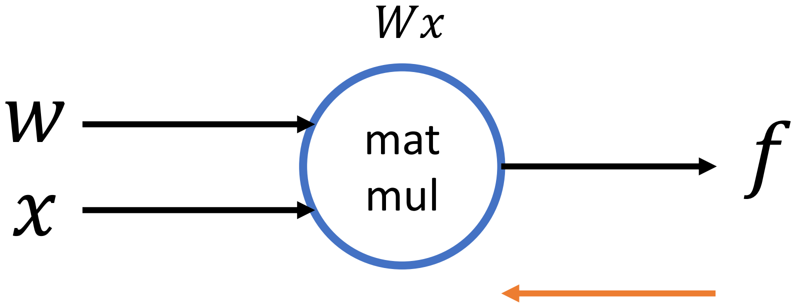
Wx

mat
mul

f $\begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix}$ (n_h, m)

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$

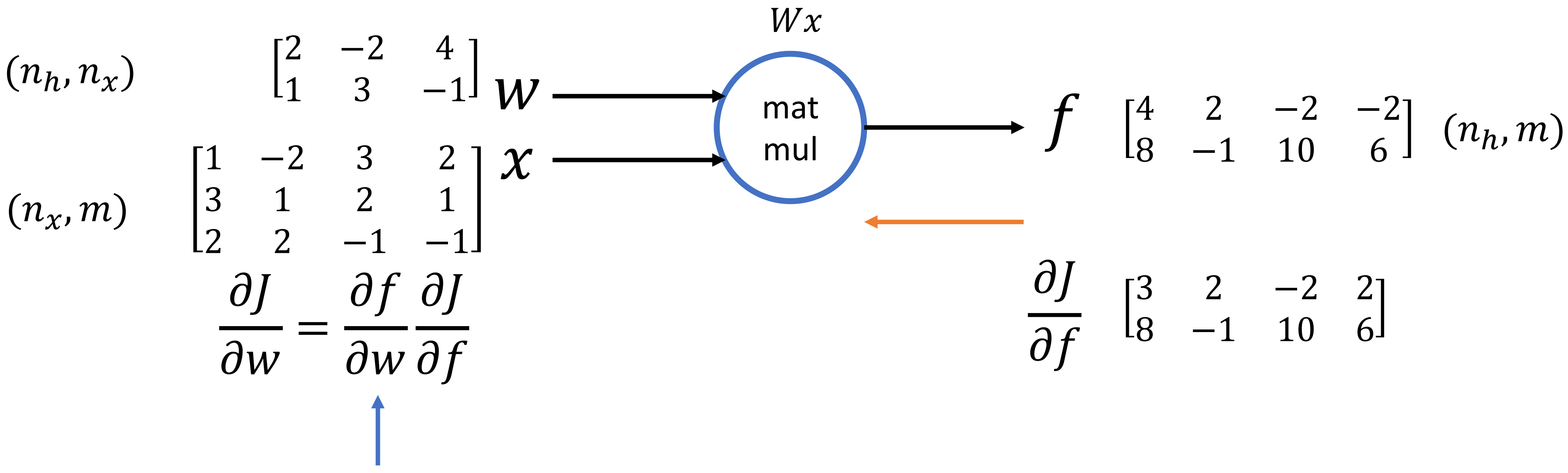
(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$



$\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \begin{bmatrix} 3 & 2 & -2 & 2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} = \frac{\partial f}{\partial w} \frac{\partial J}{\partial f}$$



Wx

(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix}$ w

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix}$ x

f $\begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix}$ (n_h, m)

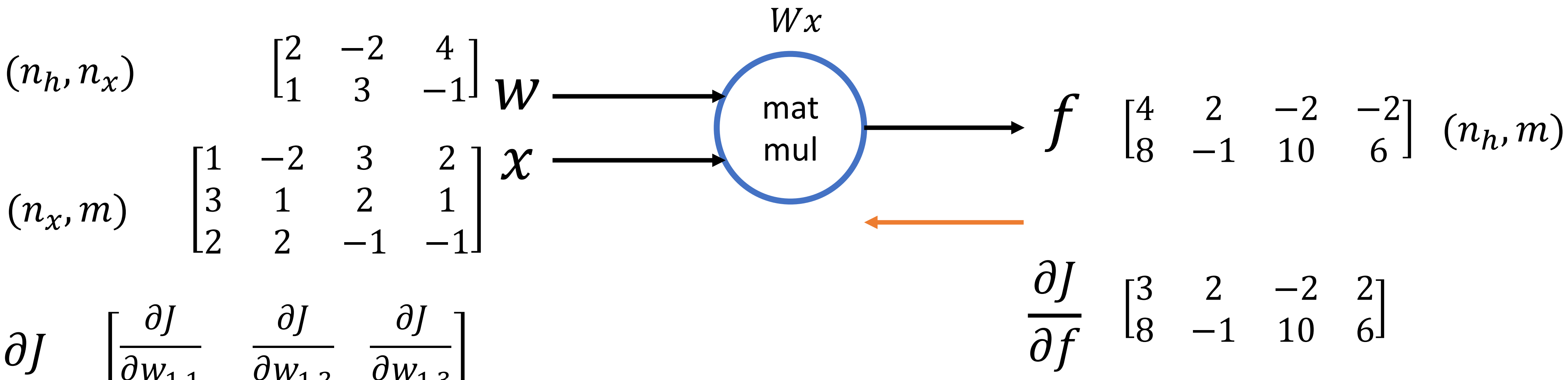
$\frac{\partial J}{\partial f}$ $\begin{bmatrix}3 & 2 & -2 & 2\\ 8 & -1 & 10 & 6\end{bmatrix}$

$$\frac{\partial J}{\partial w} = \frac{\partial f}{\partial w}\frac{\partial J}{\partial f}$$

- Jacobian is shape (n_h, n_x, n_h, m)

This can get big. For example:

- $m = 128$
- $n_h = n_x = 2048$
- Full Jacobian is 2048*2048*2048*128*4bytes = 1TeraByte!



(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix}$ W

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix}$ x

f $\begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix}$ (n_h, m)

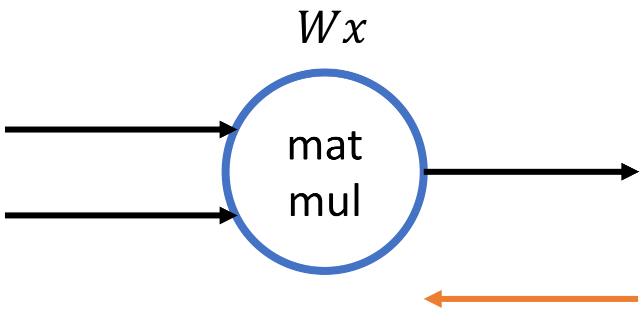
$\dfrac{\partial J}{\partial f}$ $\begin{bmatrix}3 & 2 & -2 & 2\\ 8 & -1 & 10 & 6\end{bmatrix}$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix}\dfrac{\partial J}{\partial w_{1,1}} & \dfrac{\partial J}{\partial w_{1,2}} & \dfrac{\partial J}{\partial w_{1,3}}\\ \dfrac{\partial J}{\partial w_{2,1}} & \dfrac{\partial J}{\partial w_{2,2}} & \dfrac{\partial J}{\partial w_{2,3}}\end{bmatrix}$$

- Let’s look at the Jacobian and see if we can avoid forming it
- Let’s start by computing gradients $\frac{\partial J}{\partial w}$
- $\frac{\partial J}{\partial w}$ will have the same shape as w
- Let’s look at each element separately
- Each element tells us how much the one weight $w_{i,j}$ affects J

(n_h, n_x) $$\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} W$$

(n_x, m) $$\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$$



$$f \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} (n_h, m)$$

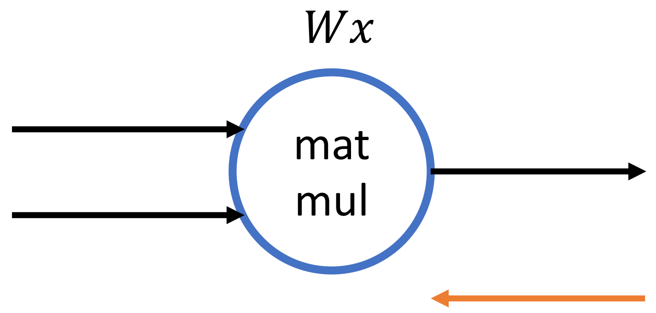
$$\frac{\partial J}{\partial f} \begin{bmatrix} 3 & 2 & -2 & 2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \begin{bmatrix} \boxed{\frac{\partial J}{\partial w_{1,1}}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \begin{bmatrix} 3 & 2 & -2 & 2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$$

$$\frac{\partial J}{\partial W} \begin{bmatrix} \boxed{\frac{\partial J}{\partial w_{1,1}}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

How does the one weight $w_{1,1}$ affect J?

Slice of the Jacobian.
How does the one weight $w_{1,1}$ affect f?
Derivative of a matrix by a scalar

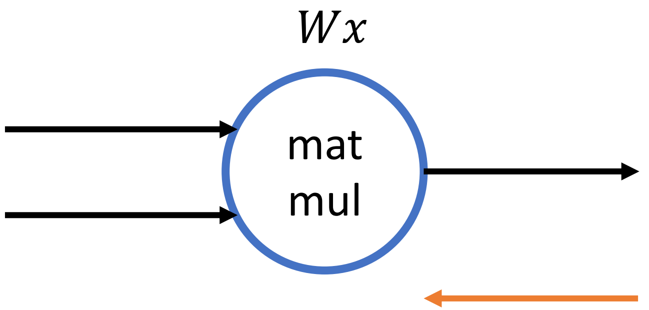
Math – Derivative of a Matrix by a Scalar

- Consider a matrix output function f with
  - Scalar input x
  - Matrix output F with shape (n, m)
- How does a small change the input, x, affect each output?
- Derivative is same shape as the output matrix

$$\frac{\partial F}{\partial x} = \begin{bmatrix} \frac{\partial f_{1,1}}{\partial x}, \frac{\partial f_{1,2}}{\partial x}, & \cdots & \frac{\partial f_{1,m}}{\partial x} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{n,1}}{\partial x}, \frac{\partial f_{n,2}}{\partial x} & \cdots & \frac{\partial f_{n,m}}{\partial x} \end{bmatrix}$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

$$\frac{\partial J}{\partial W} \begin{bmatrix} \boxed{\frac{\partial J}{\partial w_{1,1}}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

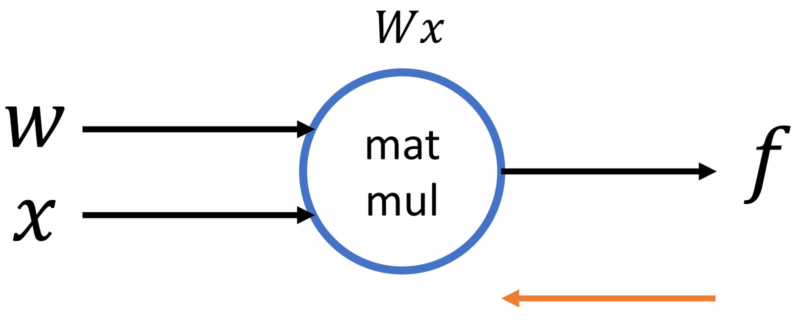
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

How does the one weight $w_{1,1}$ affect J?

Slice of the Jacobian.
How does the one weight $w_{1,1}$ affect f?
Derivative of a matrix by a scalar

(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\1 & 3 & -1\end{bmatrix}$ W

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\3 & 1 & 2 & 1\\2 & 2 & -1 & -1\end{bmatrix}$ x



f $\begin{bmatrix}4 & 2 & -2 & -2\\8 & -1 & 10 & 6\end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \begin{bmatrix}3 & 1 & -1 & 4\\-2 & 2 & 5 & 1\end{bmatrix}$$

$$\frac{\partial J}{\partial W} \begin{bmatrix}\frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}}\\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}}\end{bmatrix}$$

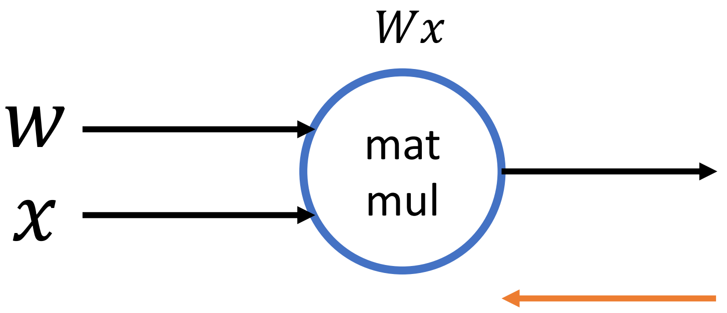
$$\frac{\partial J}{\partial w_{1,1}} = \boxed{\frac{\partial f}{\partial w_{1,1}}} \cdot \frac{\partial J}{\partial f}$$

$$\boxed{\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix}\frac{\partial f_{1,1}}{\partial w_{1,1}} & \frac{\partial f_{1,2}}{\partial w_{1,1}} & \frac{\partial f_{1,3}}{\partial w_{1,1}} & \frac{\partial f_{1,4}}{\partial w_{1,1}}\\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}}\end{bmatrix}}$$

Derivative of matrix by scalar. So this will be a matrix with same shape as f
Let's compute the values

$$(n_h, n_x) \quad \begin{bmatrix} \boxed{\begin{matrix} 2 & -2 & 4 \end{matrix}} \\ \begin{matrix} 1 & 3 & -1 \end{matrix} \end{bmatrix} \; W$$

$$(n_x, m) \quad \left[\boxed{\begin{matrix} 1 \\ 3 \\ 2 \end{matrix}} \; \begin{matrix} -2 & 3 & 2 \\ 1 & 2 & 1 \\ 2 & -1 & -1 \end{matrix} \right] \; x$$



$$f \quad \begin{bmatrix} \boxed{4} & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial W} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

First we can write the equation for $f_{1,1}$

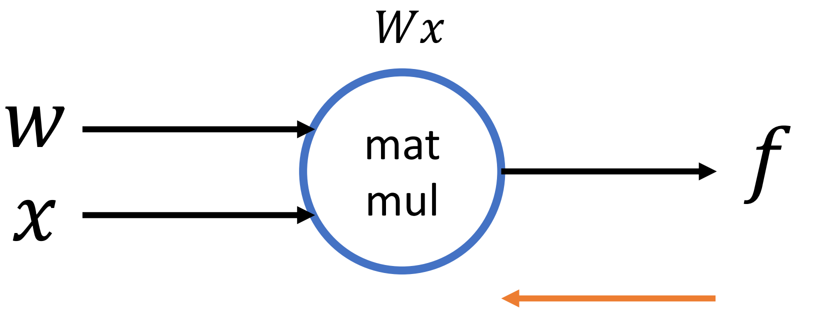
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f} \qquad \frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} \boxed{\frac{\partial f_{1,1}}{\partial w_{1,1}}} & \frac{\partial f_{1,2}}{\partial w_{1,1}} & \frac{\partial f_{1,3}}{\partial w_{1,1}} & \frac{\partial f_{1,4}}{\partial w_{1,1}} \\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{1,1} = w_{1,1}x_{1,1} + w_{1,2}x_{2,1} + w_{1,3}x_{3,1}$$

$$\frac{\partial f_{1,1}}{\partial w_{1,1}} = x_{1,1} = 1$$

$$(n_h, n_x) \quad W = \begin{bmatrix} \boxed{2} & \boxed{-2} & \boxed{4} \\ 1 & 3 & -1 \end{bmatrix}$$

$$(n_x, m) \quad x = \begin{bmatrix} \boxed{1} & -2 & 3 & 2 \\ \boxed{3} & 1 & 2 & 1 \\ \boxed{2} & 2 & -1 & -1 \end{bmatrix}$$



$$f = \begin{bmatrix} \boxed{4} & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

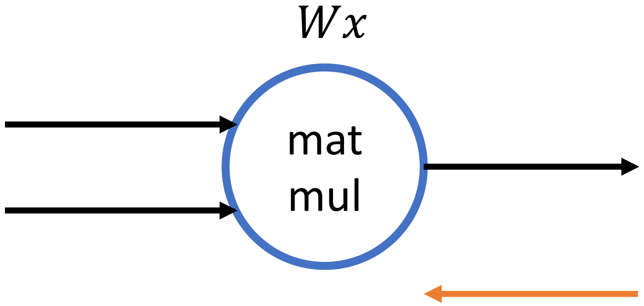
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f} \qquad \frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} \boxed{1} & \frac{\partial f_{1,2}}{\partial w_{1,1}} & \frac{\partial f_{1,3}}{\partial w_{1,1}} & \frac{\partial f_{1,4}}{\partial w_{1,1}} \\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{1,1} = w_{1,1}x_{1,1} + w_{1,2}x_{2,1} + w_{1,3}x_{3,1}$$

$$\frac{\partial f_{1,1}}{\partial w_{1,1}} = x_{1,1} = 1$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & \boxed{2} & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

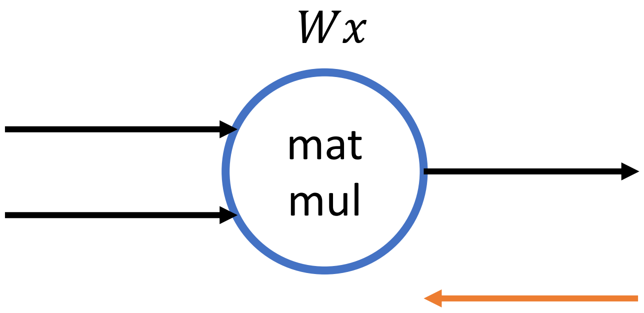
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f} \qquad \frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & \boxed{\frac{\partial f_{1,2}}{\partial w_{1,1}}} & \frac{\partial f_{1,3}}{\partial w_{1,1}} & \frac{\partial f_{1,4}}{\partial w_{1,1}} \\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{1,2} = w_{1,1}x_{1,2} + w_{1,2}x_{2,2} + w_{1,3}x_{3,2}$$

$$\frac{\partial f_{1,2}}{\partial w_{1,1}} = x_{1,2} = -2$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & \boxed{2} & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\dfrac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

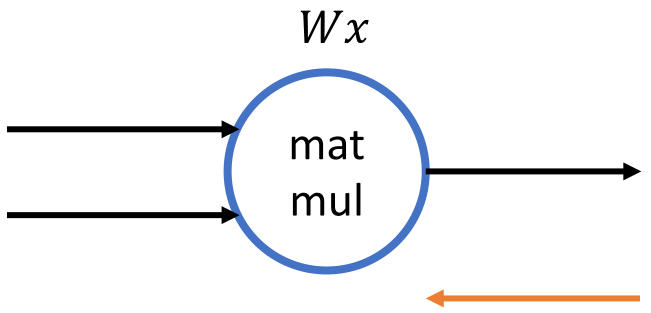
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f} \qquad \frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & \boxed{-2} & \frac{\partial f_{1,3}}{\partial w_{1,1}} & \frac{\partial f_{1,4}}{\partial w_{1,1}} \\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{1,2} = w_{1,1}x_{1,2} + w_{1,2}x_{2,2} + w_{1,3}x_{3,2}$$

$$\frac{\partial f_{1,2}}{\partial w_{1,1}} = x_{1,2} = -2$$

$$(n_h, n_x) \quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} \; W$$

$$(n_x, m) \quad \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} \; x$$



$$f \quad \begin{bmatrix} 4 & 2 & \boxed{-2} & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

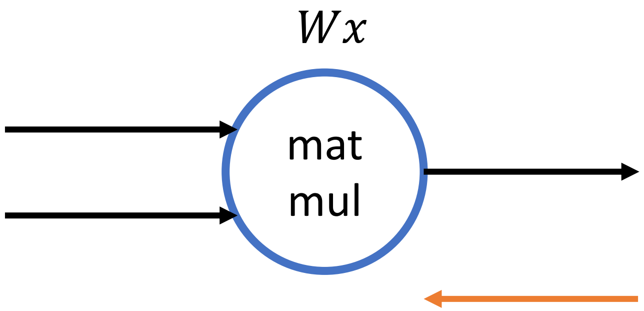
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f} \qquad \frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & \boxed{\frac{\partial f_{1,3}}{\partial w_{1,1}}} & \frac{\partial f_{1,4}}{\partial w_{1,1}} \\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{1,3} = w_{1,1}x_{1,3} + w_{1,2}x_{2,3} + w_{1,3}x_{3,3}$$

$$\frac{\partial f_{1,3}}{\partial w_{1,1}} = x_{1,3} = 3$$

(n_h, n_x) $\quad \begin{bmatrix} \boxed{2 \quad -2 \quad 4} \\ 1 \quad\;\; 3 \quad -1 \end{bmatrix}$ W

(n_x, m) $\quad \begin{bmatrix} 1 & -2 & \boxed{3} & 2 \\ 3 & 1 & \boxed{2} & 1 \\ 2 & 2 & \boxed{-1} & -1 \end{bmatrix}$ x



f $\quad \begin{bmatrix} 4 & 2 & \boxed{-2} & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

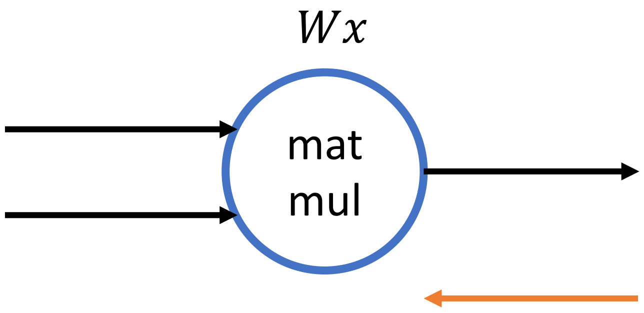
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f} \qquad \frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & \boxed{3} & \frac{\partial f_{1,4}}{\partial w_{1,1}} \\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{1,3} = w_{1,1}x_{1,3} + w_{1,2}x_{2,3} + w_{1,3}x_{3,3}$$

$$\frac{\partial f_{1,3}}{\partial w_{1,1}} = x_{1,3} = 3$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ w

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & \boxed{-2} \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

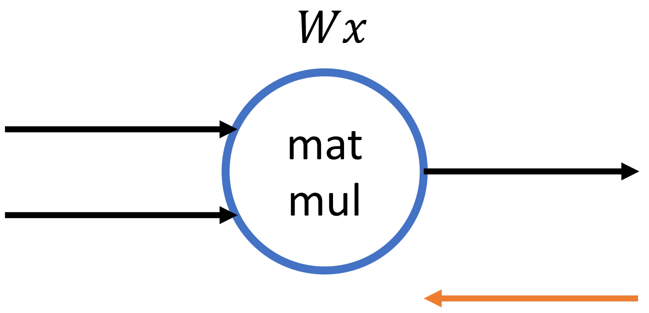
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & \boxed{2} \\ \frac{\partial f_{2,1}}{\partial w_{1,1}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{1,4} = w_{1,1}x_{1,4} + w_{1,2}x_{2,4} + w_{1,3}x_{3,4}$$

$$\frac{\partial f_{1,4}}{\partial w_{1,1}} = x_{1,4} = 2$$

(n_h, n_x) $$W = \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$$

(n_x, m) $$x = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$



$$f = \begin{bmatrix} 4 & 2 & -2 & -2 \\ \boxed{8} & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial W} \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

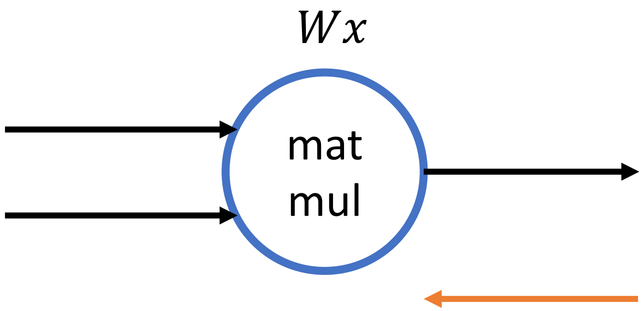
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ \boxed{\frac{\partial f_{2,1}}{\partial w_{1,1}}} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{2,1} = w_{2,1}x_{1,1} + w_{2,2}x_{2,1} + w_{2,3}x_{3,1}$$

$$\frac{\partial f_{2,1}}{\partial w_{1,1}} = 0$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ \boxed{1 \quad 3 \quad -1} \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} \boxed{\begin{matrix}1\\3\\2\end{matrix}} & \begin{matrix}-2\\1\\2\end{matrix} & \begin{matrix}3\\2\\-1\end{matrix} & \begin{matrix}2\\1\\-1\end{matrix} \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ \boxed{8} & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial W} \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

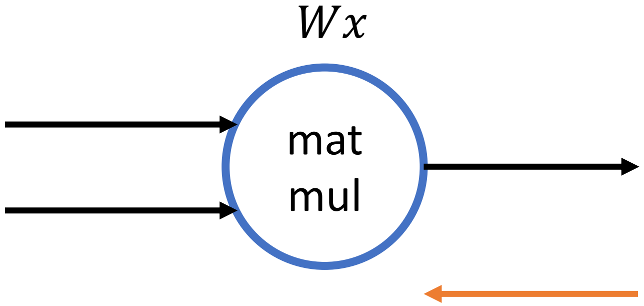
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ \boxed{0} & \frac{\partial f_{2,2}}{\partial w_{1,1}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{2,1} = w_{2,1}x_{1,1} + w_{2,2}x_{2,1} + w_{2,3}x_{3,1}$$

$$\frac{\partial f_{2,1}}{\partial w_{1,1}} = 0$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & \boxed{-1} & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

$$\frac{\partial J}{\partial w} \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

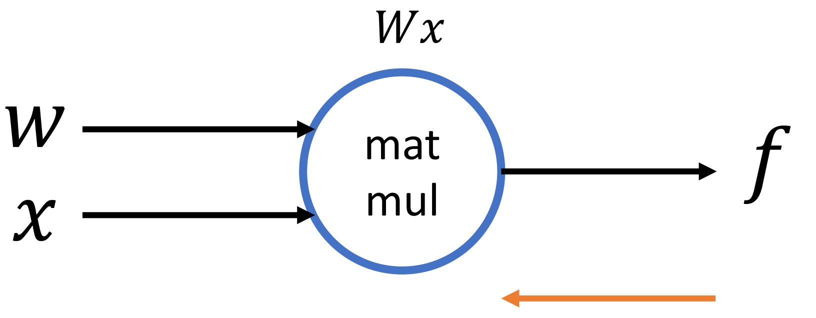
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & \boxed{\frac{\partial f_{2,2}}{\partial w_{1,1}}} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{2,2} = w_{2,1}x_{1,2} + w_{2,2}x_{2,2} + w_{2,3}x_{3,2}$$

$$\frac{\partial f_{2,2}}{\partial w_{1,1}} = 0$$

(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix}$

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix}$



$\begin{bmatrix}4 & 2 & -2 & -2\\ 8 & \boxed{-1} & 10 & 6\end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \begin{bmatrix}3 & 1 & -1 & 4\\ -2 & 2 & 5 & 1\end{bmatrix}$$

$$\frac{\partial J}{\partial W} \begin{bmatrix}\frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}}\\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}}\end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

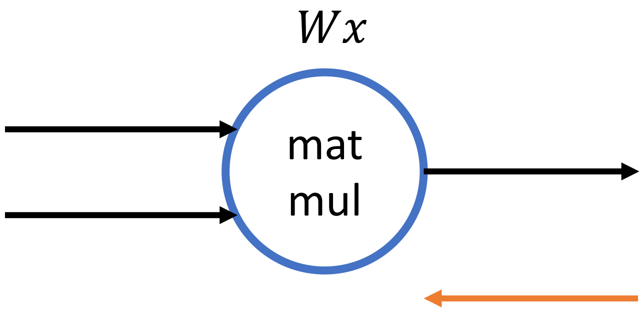
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix}1 & -2 & 3 & 2\\ 0 & \boxed{0} & \frac{\partial f_{2,3}}{\partial w_{1,1}} & \frac{\partial f_{2,4}}{\partial w_{1,1}}\end{bmatrix}$$

$$f_{2,2} = w_{2,1}x_{1,2} + w_{2,2}x_{2,2} + w_{2,3}x_{3,2}$$

$$\frac{\partial f_{2,2}}{\partial w_{1,1}} = 0$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & \boxed{10} & 6 \end{bmatrix}$ (n_h, m)

$\dfrac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

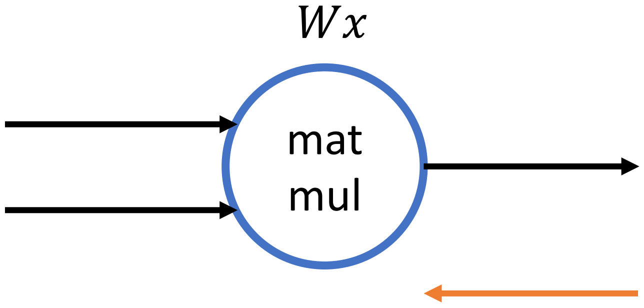
$$\frac{\partial J}{\partial W} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & \boxed{\frac{\partial f_{2,3}}{\partial w_{1,1}}} & \frac{\partial f_{2,4}}{\partial w_{1,1}} \end{bmatrix}$$

$$f_{2,3} = w_{2,1}x_{1,3} + w_{2,2}x_{2,3} + w_{2,3}x_{3,3}$$

$$\frac{\partial f_{2,3}}{\partial w_{1,1}} = 0$$



(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix}$ W

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix}$ x

f $\begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & \boxed{10} & 6\end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix}3 & 1 & -1 & 4\\ -2 & 2 & 5 & 1\end{bmatrix}$$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix}\frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}}\\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}}\end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix}1 & -2 & 3 & 2\\ 0 & 0 & \boxed{0} & \frac{\partial f_{2,4}}{\partial w_{1,1}}\end{bmatrix}$$

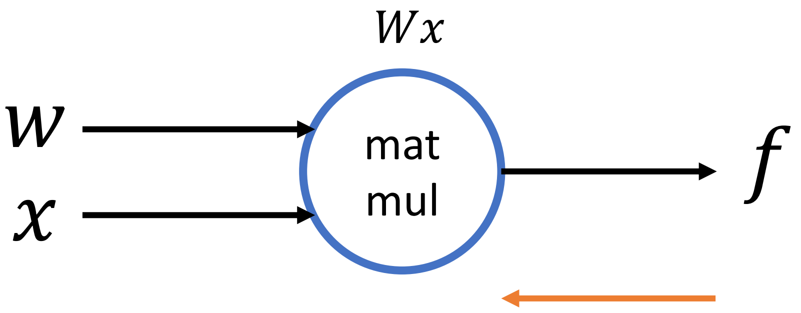
$$f_{2,3} = w_{2,1}x_{1,3} + w_{2,2}x_{2,3} + w_{2,3}x_{3,3}$$

$$\frac{\partial f_{2,3}}{\partial w_{1,1}} = 0$$

Brad Quinton, Scott Chin

$$(n_h, n_x) \quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} W$$

$$(n_x, m) \quad \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$$



$$f \quad \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & \boxed{6} \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

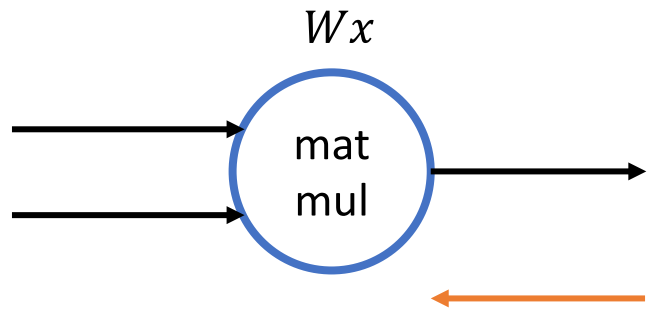
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & \boxed{\frac{\partial f_{2,4}}{\partial w_{1,1}}} \end{bmatrix}$$

$$f_{2,4} = w_{2,1}x_{1,4} + w_{2,2}x_{2,4} + w_{2,3}x_{3,4}$$

$$\frac{\partial f_{2,4}}{\partial w_{1,1}} = 0$$

$$(n_h, n_x) \quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} W$$

$$(n_x, m) \quad \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$$



$$f \quad \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix} \frac{\partial J}{\partial w_{1,1}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

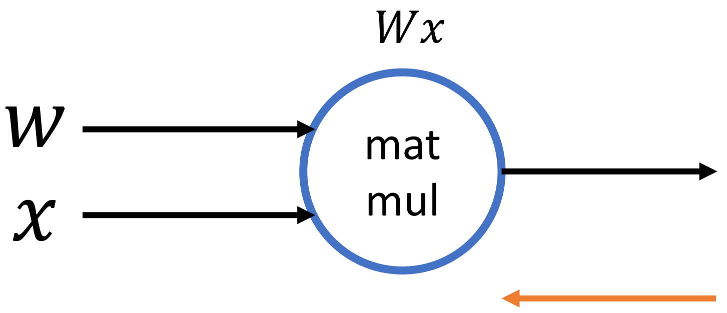
$$\boxed{\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}}$$

Reminder:

This is part of the full Jacobian $\frac{\partial f}{\partial w}$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

$$\frac{\partial J}{\partial W} \begin{bmatrix} \boxed{\frac{\partial J}{\partial w_{1,1}}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

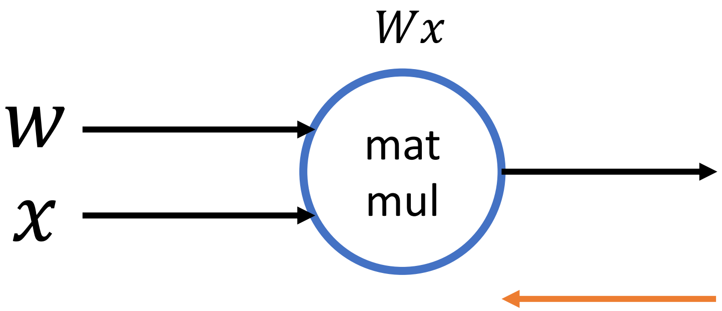
Now we can compute this

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix}$ W

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix}$ x



f $\begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix}$ (n_h, m)

$\dfrac{\partial J}{\partial f}$ $\begin{bmatrix}3 & 1 & -1 & 4\\ -2 & 2 & 5 & 1\end{bmatrix}$

$$\frac{\partial J}{\partial w} \begin{bmatrix} \boxed{\dfrac{\partial J}{\partial w_{1,1}}} & \dfrac{\partial J}{\partial w_{1,2}} & \dfrac{\partial J}{\partial w_{1,3}} \\ \dfrac{\partial J}{\partial w_{2,1}} & \dfrac{\partial J}{\partial w_{2,2}} & \dfrac{\partial J}{\partial w_{2,3}} \end{bmatrix}$$

Remember:
The full operation is a 4D-Tensor Jacobian multiply with a 2D-Tensor. We are only looking at one part of this operation.

$$\boxed{\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}}$$

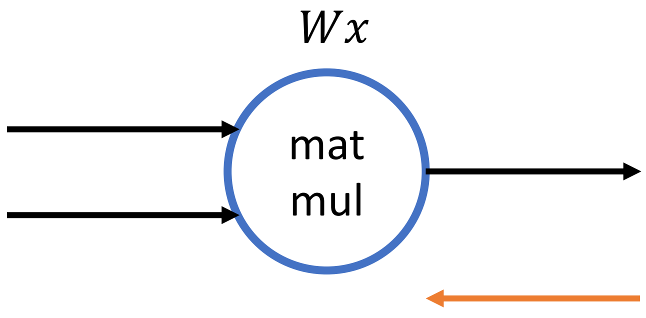
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix}1 & -2 & 3 & 2\\ 0 & 0 & 0 & 0\end{bmatrix}$$

Inner product
(i.e. sum of elementwise multiply)

$$\boxed{\frac{\partial J}{\partial w_{1,1}} = 1 * 3 + (-2) * 1 + 3 * (-1) + 2 * 4 + 0 * (-2) + 0 * (2) + 0 * 5 + 0 * 1 = 6}$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} W$

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$$\frac{\partial J}{\partial f} \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} \begin{bmatrix} \boxed{6} & \frac{\partial L}{\partial w_{1,2}} & \frac{\partial L}{\partial w_{1,3}} \\ \frac{\partial L}{\partial w_{2,1}} & \frac{\partial L}{\partial w_{2,2}} & \frac{\partial L}{\partial w_{2,3}} \end{bmatrix}$$

Weight $w_{1,2}$ has a -3 “impact” on Cost

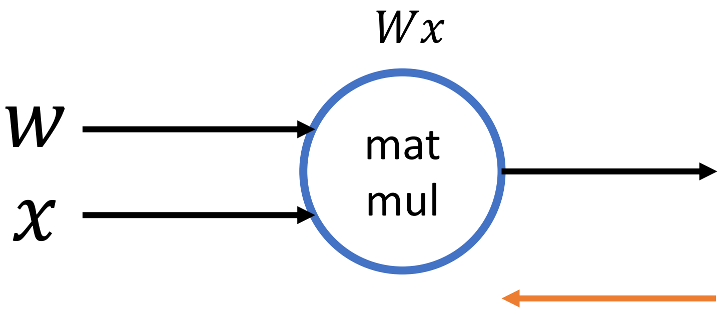
$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = 1 * 3 + (-2) * 1 + 3 * (-1) + 2 * 4 + 0 * (-2) + 0 * (2) + 0 * 5 + 0 * 1 = \boxed{6}$$

$$(n_h, n_x) \quad \begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix} W$$

$$(n_x, m) \quad \begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix} x$$



$$f \quad \begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix} \quad (n_h, m)$$

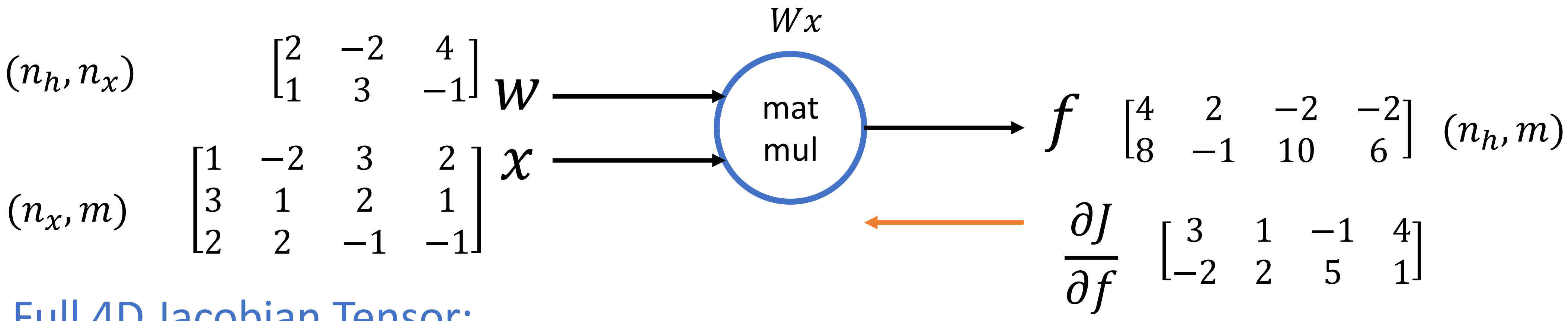
$$\frac{\partial J}{\partial f} \quad \begin{bmatrix}3 & 1 & -1 & 4\\ -2 & 2 & 5 & 1\end{bmatrix}$$

$$\frac{\partial J}{\partial w} \quad \begin{bmatrix}\boxed{6} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}}\\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}}\end{bmatrix}$$

$$= \begin{bmatrix}\boxed{\frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}} & \frac{\partial f}{\partial w_{1,2}} \cdot \frac{\partial J}{\partial f} & \frac{\partial f}{\partial w_{1,3}} \cdot \frac{\partial J}{\partial f}\\ \frac{\partial f}{\partial w_{2,1}} \cdot \frac{\partial J}{\partial f} & \frac{\partial f}{\partial w_{2,2}} \cdot \frac{\partial J}{\partial f} & \frac{\partial f}{\partial w_{2,3}} \cdot \frac{\partial J}{\partial f}\end{bmatrix}$$

Summary so far:

- We looked at how to compute **one** element of the multiplication between the full Jacobian Tensor with the upstream gradient matrix.
- Each element of the result depends on a slice of the full 4D Jacobian Tensor.
- We looked at how to find a slice of the 4D Jacobian Tensor



(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x

f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\dfrac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \dfrac{\partial f}{\partial w_{1,1}} & \dfrac{\partial f}{\partial w_{1,2}} & \dfrac{\partial f}{\partial w_{1,3}} \\ \dfrac{\partial f}{\partial w_{2,1}} & \dfrac{\partial f}{\partial w_{2,2}} & \dfrac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

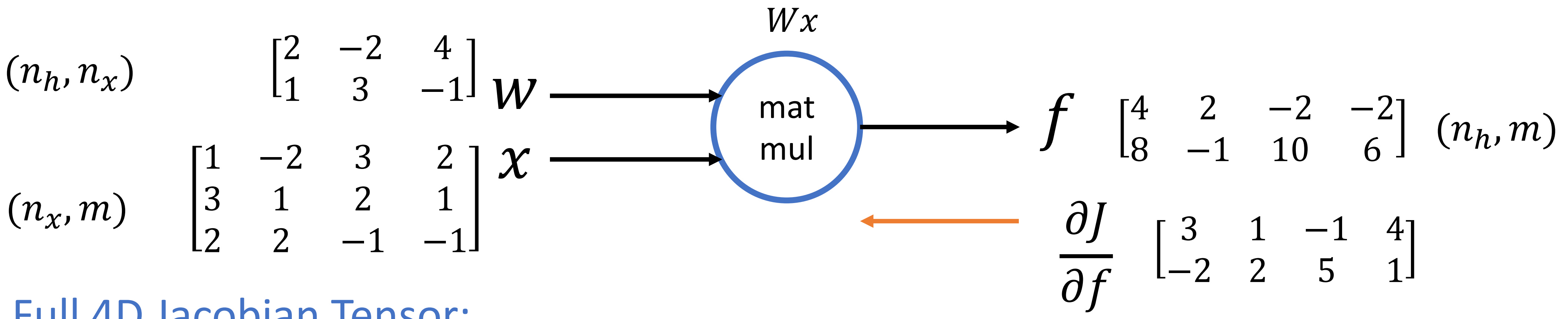
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Shape: (n_h, n_x, n_h, m) can also think of it as $\big((n_h, n_x), (n_h, m)\big)$

For this example: shape is $(2,3,2,4)$ and it’s easy to compute the whole thing

Remember this is not practical for any moderate deep neural network



$$(n_h, n_x) \quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} w$$

$$(n_x, m) \quad \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$$

$$f \quad \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

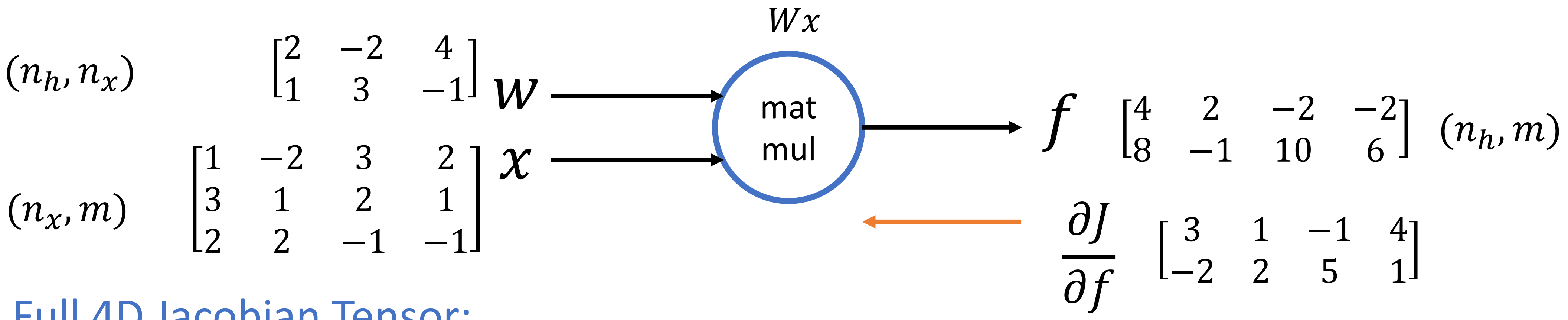
The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Notice any patterns to this Jacobian?



Wx

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ w

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x

f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\dfrac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \dfrac{\partial f}{\partial w_{1,1}} & \dfrac{\partial f}{\partial w_{1,2}} & \dfrac{\partial f}{\partial w_{1,3}} \\ \dfrac{\partial f}{\partial w_{2,1}} & \dfrac{\partial f}{\partial w_{2,2}} & \dfrac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

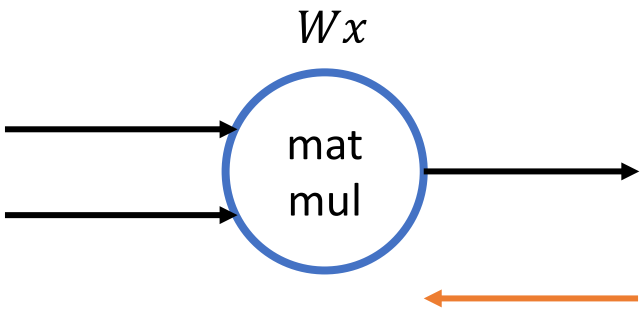
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Each slice of the Jacobian is a copy of a row from the other operand, x, and 0’s otherwise!

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\boxed{\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \boxed{\begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

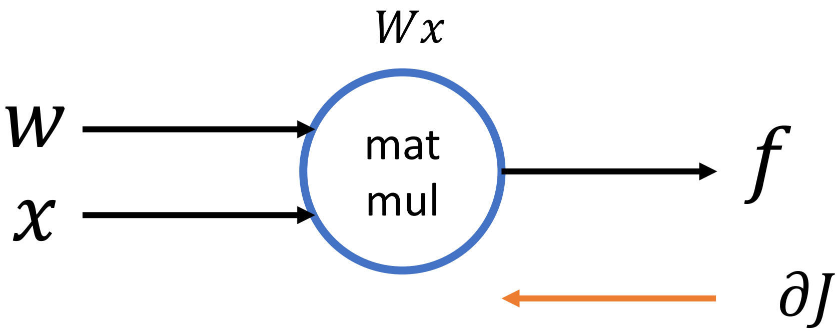
$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient.

$$\frac{\partial J}{\partial w} = \begin{bmatrix} \boxed{\frac{\partial J}{\partial w_{1,1}}} & \frac{\partial J}{\partial w_{1,2}} & \frac{\partial J}{\partial w_{1,3}} \\ \frac{\partial J}{\partial w_{2,1}} & \frac{\partial J}{\partial w_{2,2}} & \frac{\partial J}{\partial w_{2,3}} \end{bmatrix} \qquad \frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\dfrac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

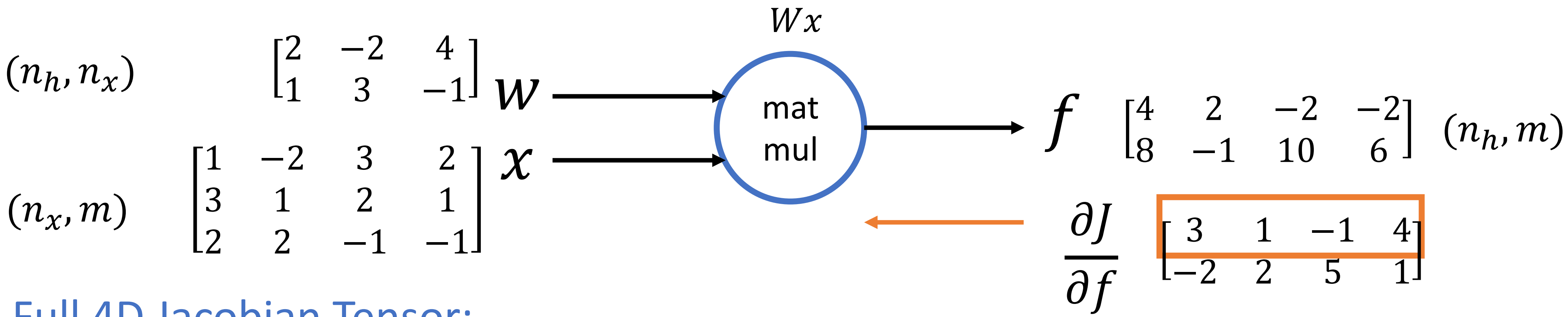
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} \boxed{6} & 12 & 5 \\ 11 & 7 & -6 \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,1}} = \frac{\partial f}{\partial w_{1,1}} \cdot \frac{\partial J}{\partial f}$$



$$(n_h, n_x) \quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} W$$

$$(n_x, m) \quad \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$$

$$f \quad \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} \boxed{\begin{matrix} 3 & 1 & -1 & 4 \end{matrix}} \\ \begin{matrix} -2 & 2 & 5 & 1 \end{matrix} \end{bmatrix}$$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

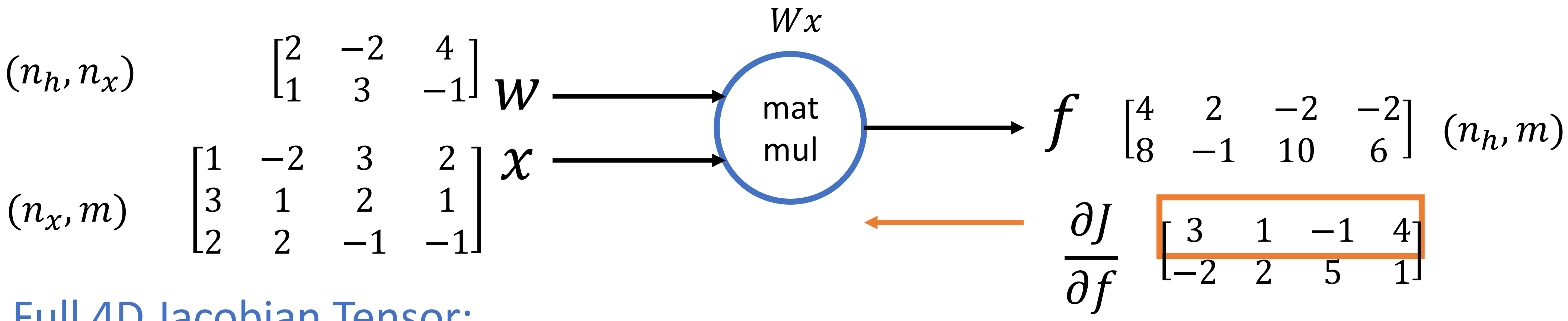
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} \boxed{\begin{matrix} 3 & 1 & 2 & 1 \end{matrix}} \\ \begin{matrix} 0 & 0 & 0 & 0 \end{matrix} \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & \boxed{12} & 5 \\ 11 & 7 & -6 \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,2}} = \frac{\partial f}{\partial w_{1,2}} \cdot \frac{\partial J}{\partial f}$$



(n_h, n_x) $\quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} \; W$

(n_x, m) $\quad \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} \; x$

$f \quad \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} \boxed{3 \quad 1 \quad -1 \quad 4} \\ -2 \quad 2 \quad 5 \quad 1 \end{bmatrix}$$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} \boxed{2 \quad 2 \quad -1 \quad -1} \\ 0 \quad 0 \quad 0 \quad 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \qquad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

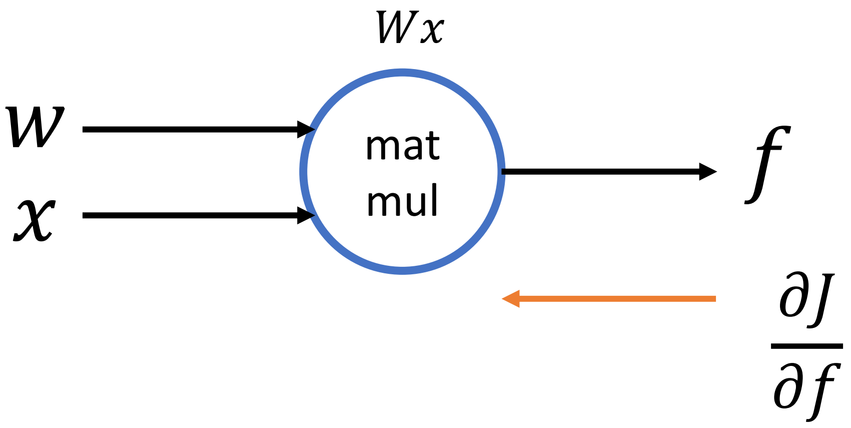
Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & \boxed{5} \\ 11 & 7 & -6 \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{1,3}} = \frac{\partial f}{\partial w_{1,3}} \cdot \frac{\partial J}{\partial f}$$

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$



$\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\left[\begin{array}{c} 3 \quad 1 \quad -1 \quad 4 \\ \boxed{-2 \quad 2 \quad 5 \quad 1} \end{array}\right]$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \left[\begin{array}{c} 0 \quad 0 \quad 0 \quad 0 \\ \boxed{1 \quad -2 \quad 3 \quad 2} \end{array}\right] \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

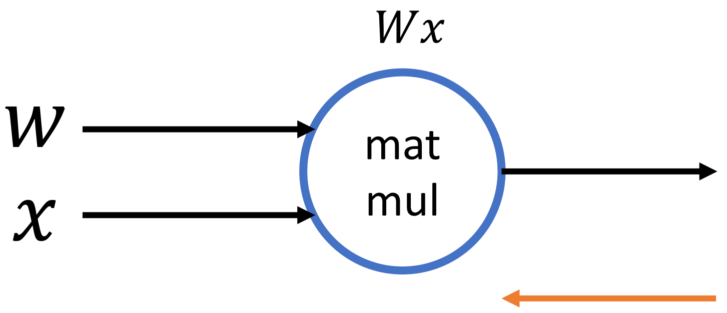
Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & 5 \\ \boxed{11} & 7 & -6 \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{2,1}} = \frac{\partial f}{\partial w_{2,1}} \cdot \frac{\partial J}{\partial f}$$

$$(n_h, n_x) \quad \begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix} W$$

$$(n_x, m) \quad \begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix} x$$



$$f \quad \begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix}3 & 1 & -1 & 4\\ \boxed{-2 \quad 2 \quad 5 \quad 1}\end{bmatrix}$$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix}\frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}}\\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}}\end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix}1 & -2 & 3 & 2\\ 0 & 0 & 0 & 0\end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix}3 & 1 & 2 & 1\\ 0 & 0 & 0 & 0\end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix}2 & 2 & -1 & -1\\ 0 & 0 & 0 & 0\end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix}0 & 0 & 0 & 0\\ 1 & -2 & 3 & 2\end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix}0 & 0 & 0 & 0\\ \boxed{3 \quad 1 \quad 2 \quad 1}\end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix}0 & 0 & 0 & 0\\ 2 & 2 & -1 & -1\end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

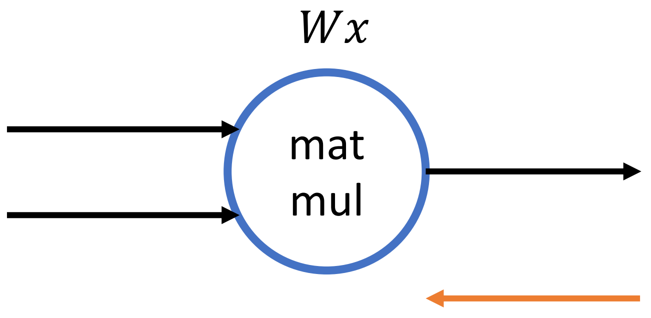
$$\frac{\partial J}{\partial w} = \begin{bmatrix}6 & 12 & 5\\ 11 & \boxed{7} & -6\end{bmatrix}$$

$$\frac{\partial J}{\partial w_{2,2}} = \frac{\partial f}{\partial w_{2,2}} \cdot \frac{\partial J}{\partial f}$$

Brad Quinton, Scott Chin

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

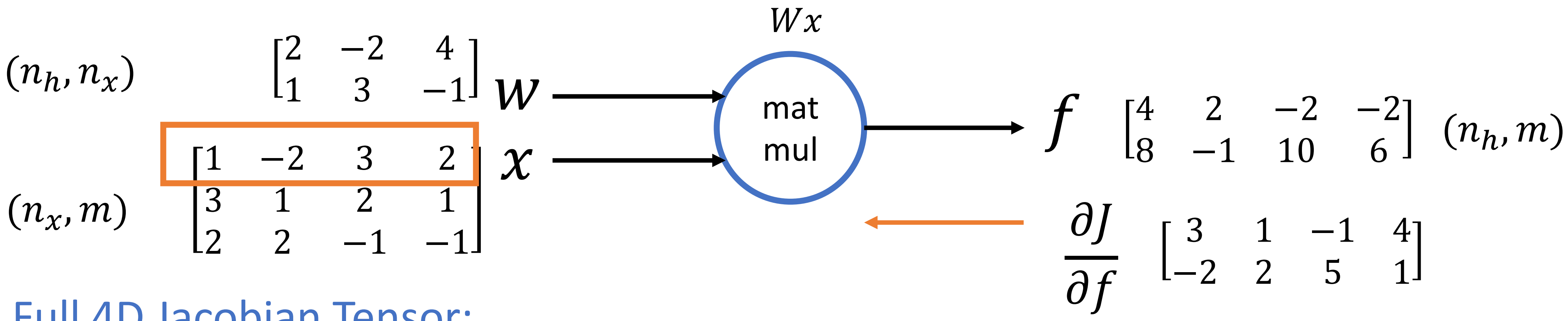
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & 5 \\ 11 & 7 & \boxed{-6} \end{bmatrix}$$

$$\frac{\partial J}{\partial w_{2,3}} = \frac{\partial f}{\partial w_{2,3}} \cdot \frac{\partial J}{\partial f}$$



(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ w

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x

Wx

f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

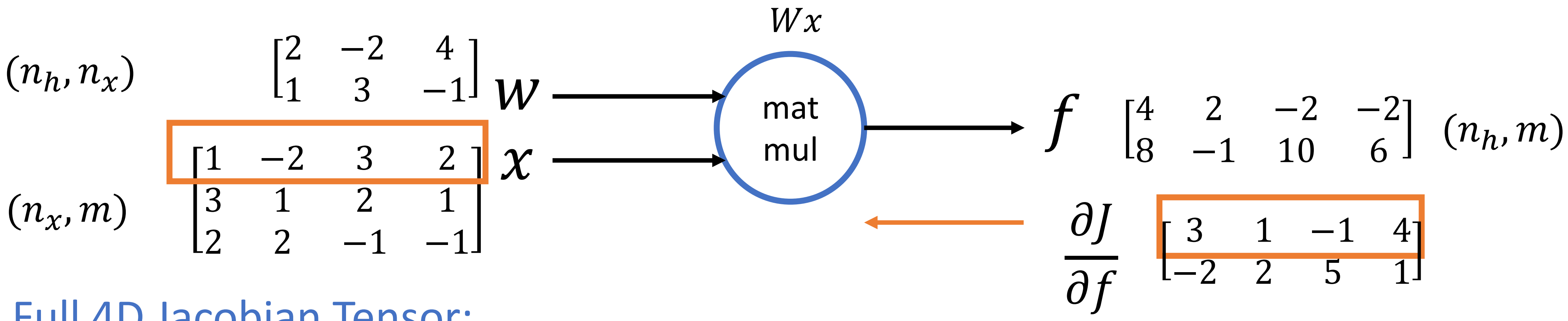
$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & 5 \\ 11 & 7 & -6 \end{bmatrix}$$

Furthermore, recall, Jacobian slices are just copies of rows from x.



(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ w

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x

f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

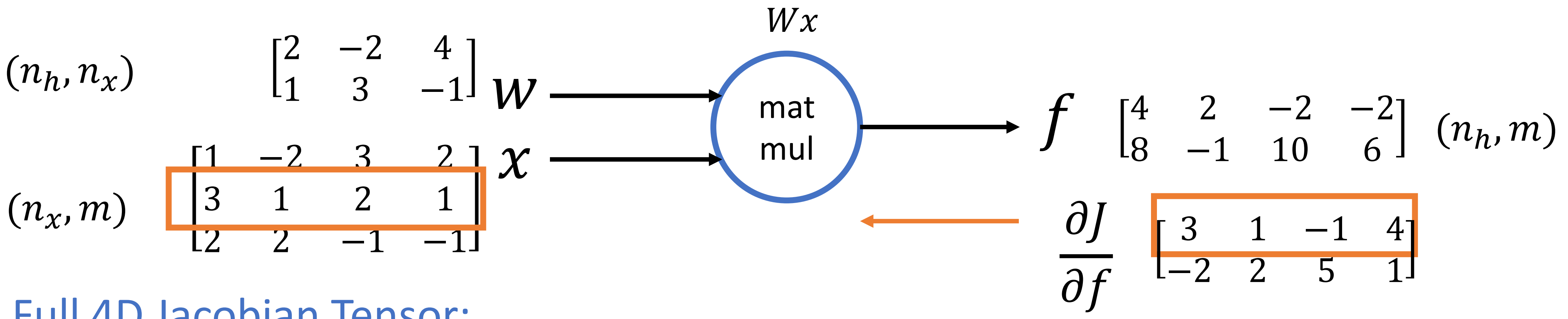
$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Don’t Need Jacobian Tensor at all

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} \boxed{6} & 12 & 5 \\ 11 & 7 & -6 \end{bmatrix}$$

Furthermore, recall, Jacobian slices are just copies of rows from x. Therefore, don’t need Jacobian at all! Just look at x!



$$(n_h, n_x) \quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} W$$

$$(n_x, m) \quad \left[\begin{array}{cccc} 1 & -2 & 3 & 2 \\ \hline 3 & 1 & 2 & 1 \\ \hline 2 & 2 & -1 & -1 \end{array}\right] x$$

$$f \quad \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \left[\begin{array}{cccc} \hline 3 & 1 & -1 & 4 \\ \hline -2 & 2 & 5 & 1 \end{array}\right]$$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Don’t Need Jacobian Tensor at all

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

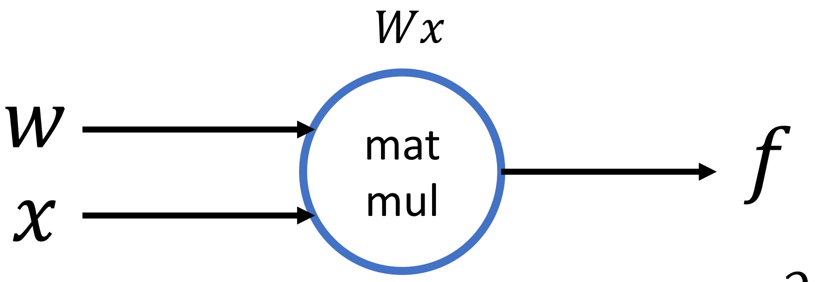
Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & \boxed{12} & 5 \\ 11 & 7 & -6 \end{bmatrix}$$

Furthermore, recall, Jacobian slices are just copies of rows from x. Therefore, don’t need Jacobian at all! Just look at x!

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ \boxed{2 \quad 2 \quad -1 \quad -1} \end{bmatrix}$ x



f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} \boxed{3 \quad 1 \quad -1 \quad 4} \\ -2 \quad 2 \quad 5 \quad 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

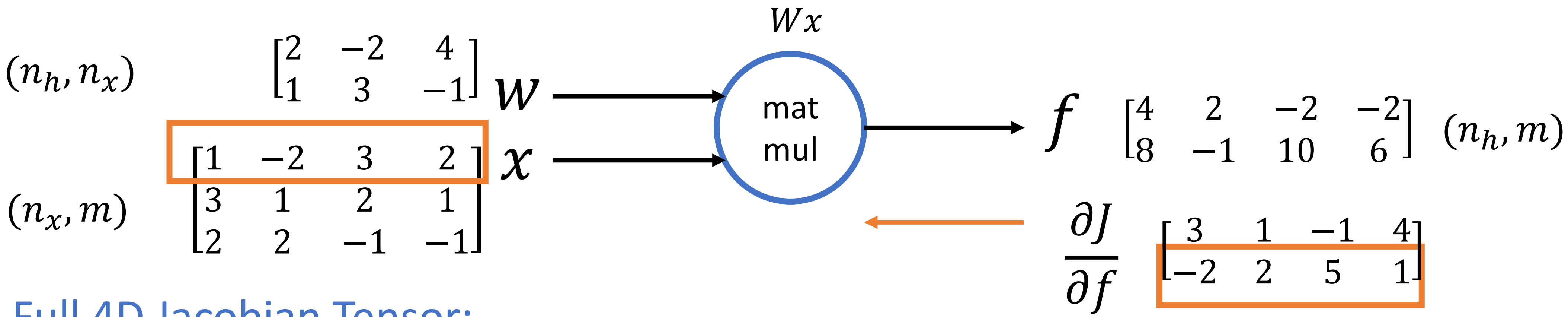
Don’t Need Jacobian Tensor at all

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & \boxed{5} \\ 11 & 7 & -6 \end{bmatrix}$$

Furthermore, recall, Jacobian slices are just copies of rows from x. Therefore, don’t need Jacobian at all! Just look at x!



(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ w

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix}$ x

f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & & \end{bmatrix} \quad \frac{\partial f}{} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ & & & \end{bmatrix} \quad \frac{\partial f}{} \quad \begin{bmatrix} & & -1 & -1 \\ & & 0 & 0 \end{bmatrix}$$

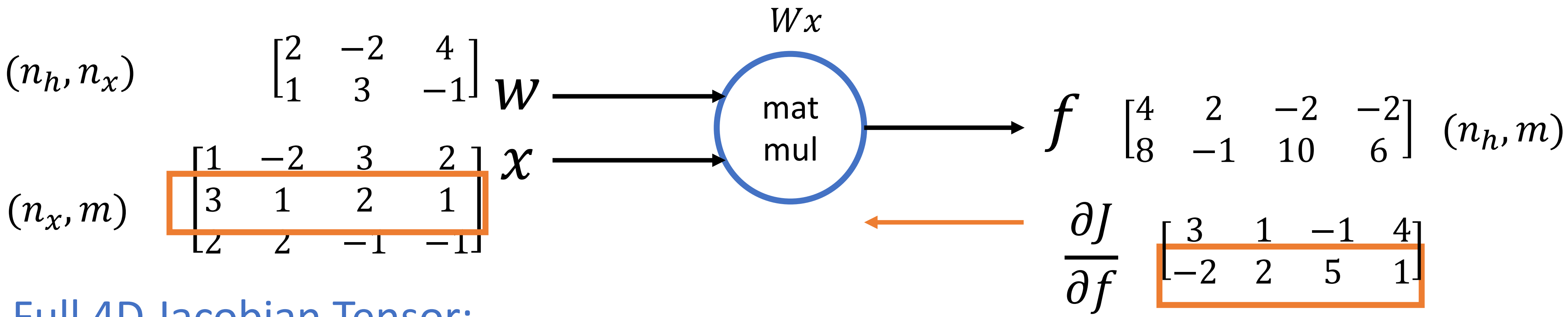
$$\frac{}{\partial w_{2,1}} = \begin{bmatrix} & & & \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Don’t Need Jacobian Tensor at all

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & 5 \\ \boxed{11} & 7 & -6 \end{bmatrix}$$

Furthermore, recall, Jacobian slices are just copies of rows from x. Therefore, don’t need Jacobian at all! Just look at x!



$$W = \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} \quad x = \begin{bmatrix} 1 & -2 & 3 & 2 \\ \boxed{3 \quad 1 \quad 2 \quad 1} \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

$$f = \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad \frac{\partial J}{\partial f} = \begin{bmatrix} 3 & 1 & -1 & 4 \\ \boxed{-2 \quad 2 \quad 5 \quad 1} \end{bmatrix}$$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} 2 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

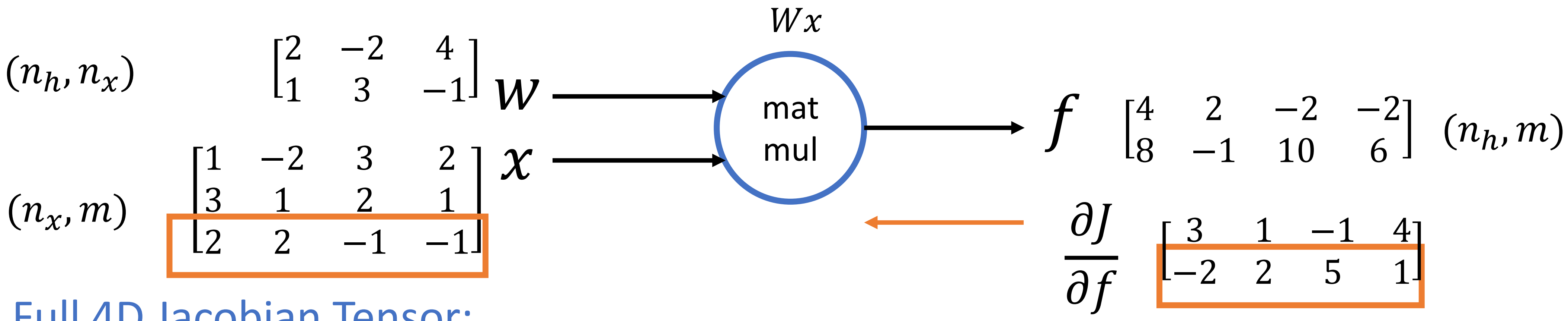
Don’t Need Jacobian Tensor at all

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & 5 \\ 11 & \boxed{7} & -6 \end{bmatrix}$$

Furthermore, recall, Jacobian slices are just copies of rows from x. Therefore, don’t need Jacobian at all! Just look at x!



(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix}$ W

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ \boxed{2 \quad 2 \quad -1 \quad -1} \end{bmatrix}$ x

f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ \boxed{-2 \quad 2 \quad 5 \quad 1} \end{bmatrix}$

The Full 4D Jacobian Tensor:

$$\frac{\partial f}{\partial w} = \begin{bmatrix} \frac{\partial f}{\partial w_{1,1}} & \frac{\partial f}{\partial w_{1,2}} & \frac{\partial f}{\partial w_{1,3}} \\ \frac{\partial f}{\partial w_{2,1}} & \frac{\partial f}{\partial w_{2,2}} & \frac{\partial f}{\partial w_{2,3}} \end{bmatrix}$$

$$\frac{\partial f}{\partial w_{1,1}} = \begin{bmatrix} 1 & -2 & 3 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,2}} = \begin{bmatrix} 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{1,3}} = \begin{bmatrix} -1 & -1 \\ 0 & 0 \end{bmatrix}$$

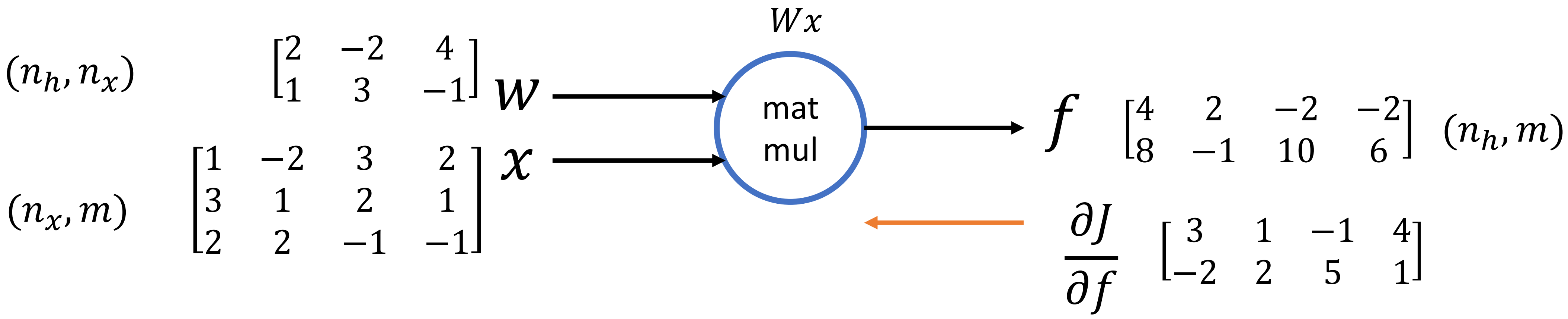
Don’t Need Jacobian Tensor at all

$$\frac{\partial f}{\partial w_{2,1}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 3 & 2 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 1 & 2 & 1 \end{bmatrix} \quad \frac{\partial f}{\partial w_{2,3}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & -1 & -1 \end{bmatrix}$$

Recall: each element of downstream gradient is inner product between slice of Jacobian and upstream gradient. But only one non-zero row!

$$\frac{\partial J}{\partial w} = \begin{bmatrix} 6 & 12 & 5 \\ 11 & 7 & \boxed{-6} \end{bmatrix}$$

Furthermore, recall, Jacobian slices are just copies of rows from x. Therefore, don’t need Jacobian at all! Just look at x!



$$(n_h, n_x) \quad \begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix} w$$

$$(n_x, m) \quad \begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix} x$$

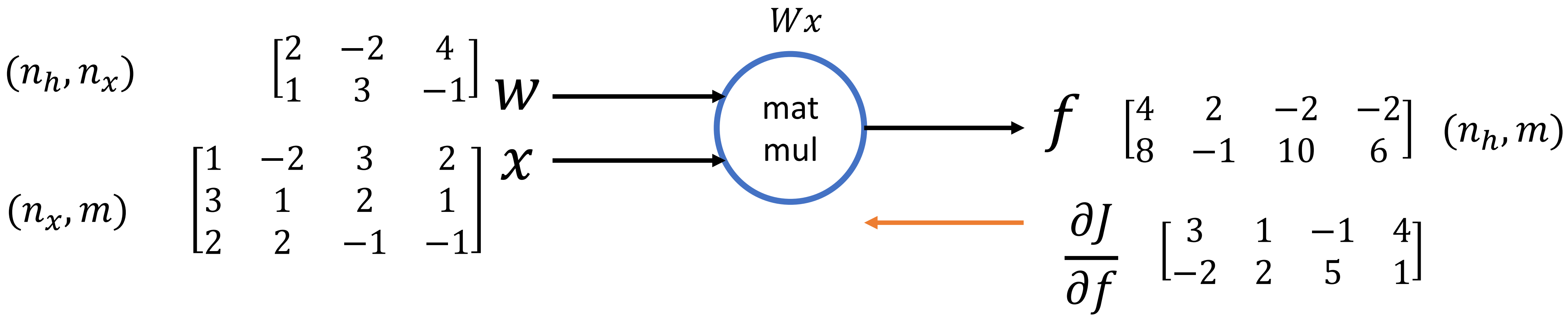
$$f \quad \begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix}3 & 1 & -1 & 4\\ -2 & 2 & 5 & 1\end{bmatrix}$$

$$\frac{\partial J}{\partial w} = \frac{\partial J}{\partial f} X^T$$

$$(n_h, n_x) \rightarrow (n_h\ , m)\ (m, n_x)$$

- No Jacobian require at all!
- This matrix multiply yields same result as doing the full 4D-Tensor Jacobian and upstream gradient matrix multiply!
- Similar intuition as scalar multiply → gradient is depending on value of other operand



(n_h, n_x) $\begin{bmatrix}2 & -2 & 4\\ 1 & 3 & -1\end{bmatrix} w$

(n_x, m) $\begin{bmatrix}1 & -2 & 3 & 2\\ 3 & 1 & 2 & 1\\ 2 & 2 & -1 & -1\end{bmatrix} x$

Wx

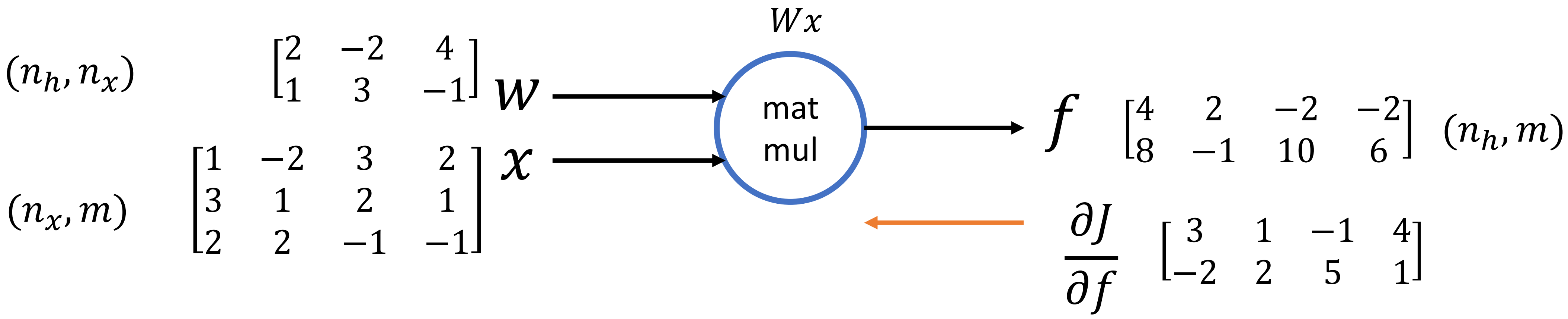
$f \begin{bmatrix}4 & 2 & -2 & -2\\ 8 & -1 & 10 & 6\end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f} \begin{bmatrix}3 & 1 & -1 & 4\\ -2 & 2 & 5 & 1\end{bmatrix}$

$$\frac{\partial J}{\partial w} = \frac{\partial J}{\partial f} X^T$$

(n_h, n_x)→$(n_h\ , m)\ (m, n_x)$

- No Jacobian require at all!
- This matrix multiply yields same result as doing the full 4D-Tensor Jacobian and upstream gradient matrix multiply!
- Similar intuition as scalar multiply → gradient is depending on value of other operand



$$Wx$$

$$(n_h, n_x) \quad \begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} w$$

$$(n_x, m) \quad \begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$$

$$f \quad \begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix} \quad (n_h, m)$$

$$\frac{\partial J}{\partial f} \quad \begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial w} = \frac{\partial J}{\partial f} X^T$$

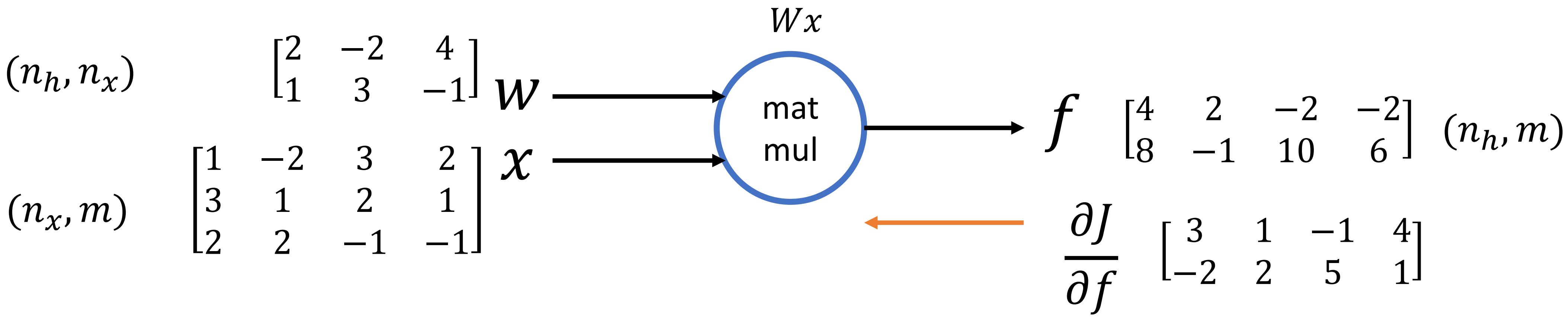
$$(n_h, n_x) \rightarrow (n_h\ , m)\ (m, n_x)$$

- No Jacobian require at all!
- This matrix multiply yields same result as doing the full 4D-Tensor Jacobian and upstream gradient matrix multiply!
- Similar intuition as scalar multiply → gradient is depending on value of other operand
- This is what you used in Assignment 2 and saw in Lecture 6

From Assignment 2 and Lecture 6

$$dW^{[2]} = \frac{1}{m} dZ^{[2]} A^{[1]T}$$

$$dW^{[1]} = \frac{1}{m} dZ^{[1]} X^T$$



Wx

mat mul

(n_h, n_x) $\begin{bmatrix} 2 & -2 & 4 \\ 1 & 3 & -1 \end{bmatrix} w$

(n_x, m) $\begin{bmatrix} 1 & -2 & 3 & 2 \\ 3 & 1 & 2 & 1 \\ 2 & 2 & -1 & -1 \end{bmatrix} x$

f $\begin{bmatrix} 4 & 2 & -2 & -2 \\ 8 & -1 & 10 & 6 \end{bmatrix}$ (n_h, m)

$\frac{\partial J}{\partial f}$ $\begin{bmatrix} 3 & 1 & -1 & 4 \\ -2 & 2 & 5 & 1 \end{bmatrix}$

$$\frac{\partial J}{\partial w} = \frac{\partial J}{\partial f} X^T$$

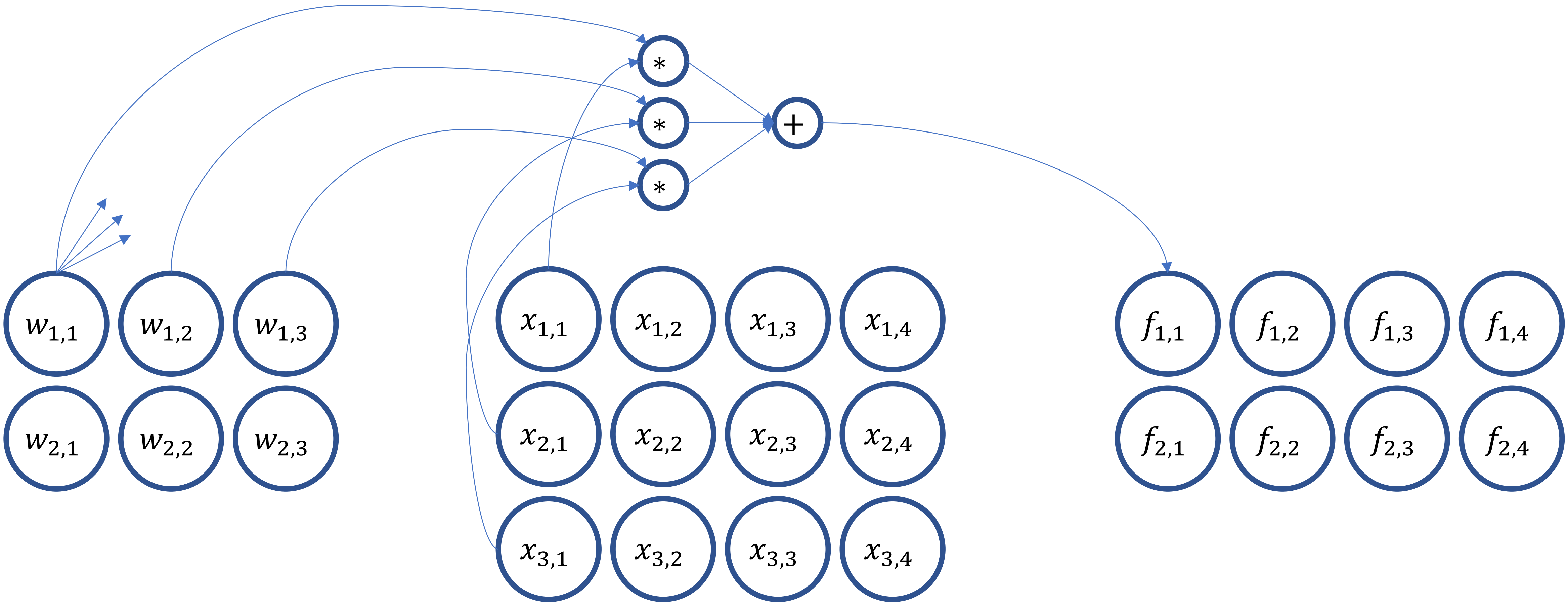
(n_h, n_x)→$(n_h\ , m)\ (m, n_x)$

$$\frac{\partial J}{\partial x} = w^T \frac{\partial J}{\partial f}$$

(n_x, m)→ $(n_x\ , n_h)\ (n_h, m)$

- No Jacobian require at all!
- This matrix multiply yields same result as doing the full 4D-Tensor Jacobian and upstream gradient matrix multiply!
- Similar intuition as scalar multiply → gradient is depending on value of other operand
- This is what you used in Assignment 2 and saw in Lecture 6

Analyzing With a Scalar View



$$\frac{\partial L}{\partial w_{1,1}} = x_{1,1} \cdot \frac{\partial L}{\partial f_{1,1}} + x_{1,2} \cdot \frac{\partial L}{\partial f_{1,2}} + x_{1,3} \cdot \frac{\partial L}{\partial f_{1,3}} + x_{1,4} \cdot \frac{\partial L}{\partial f_{1,4}}$$

Cost Function Revisited

Layer 0 (Input Layer)

$n_x = n^{[0]} = 2$

Layer 1

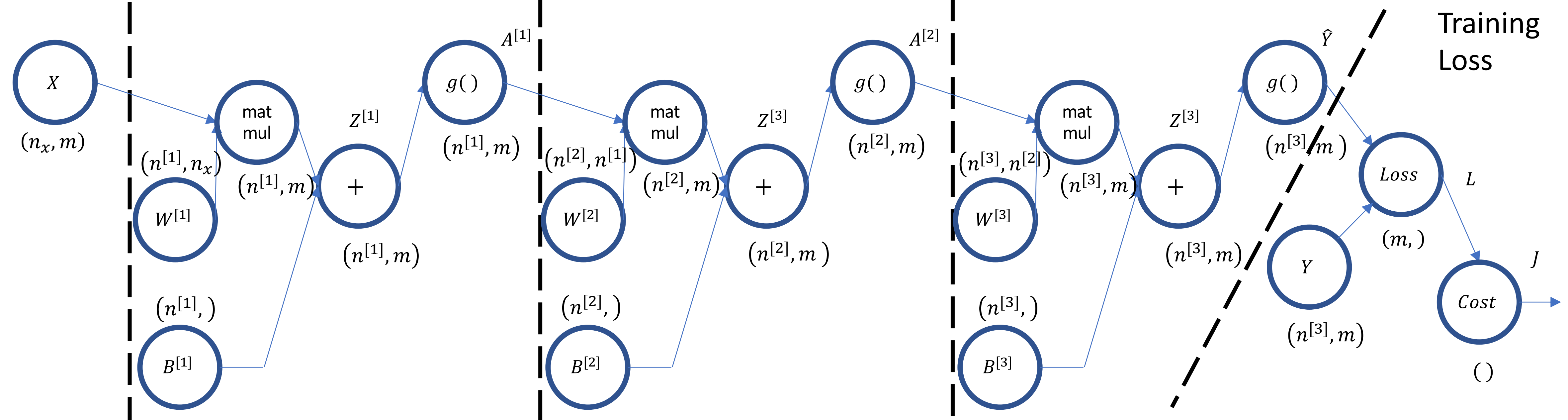
$n^{[1]} = 5$

Layer 2

$n^{[2]} = 4$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$A^{[0]} = X$
has shape (2,)

$W^{[1]}$ has shape (5, 2)

$B^{[1]}$ has shape (5,)

$A^{[1]}$ has shape (5,)

$Z^{[1]}$ has shape (5,)

$W^{[2]}$ has shape (4, 5)

$B^{[2]}$ has shape (4,)

$A^{[2]}$ has shape (4,)

$Z^{[2]}$ has shape (4,)

$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

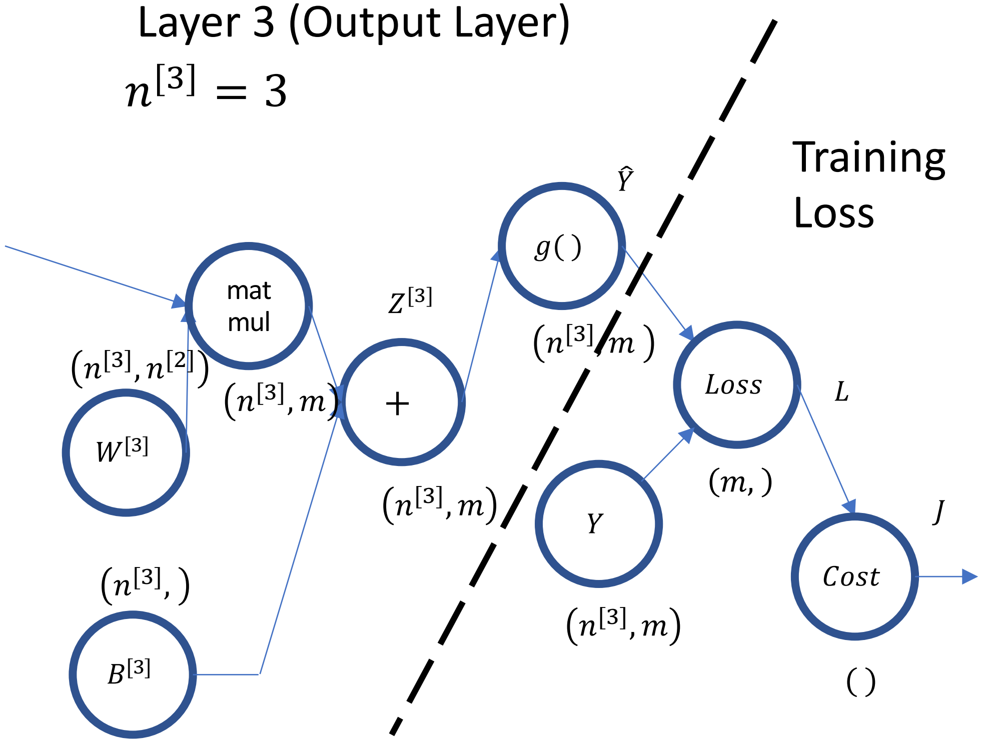
$\hat{Y} = A^{[3]}$ has shape (3,)

$Z^{[3]}$ has shape (3,)

$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

Layer 3 (Output Layer)

$n^{[3]} = 3$



$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

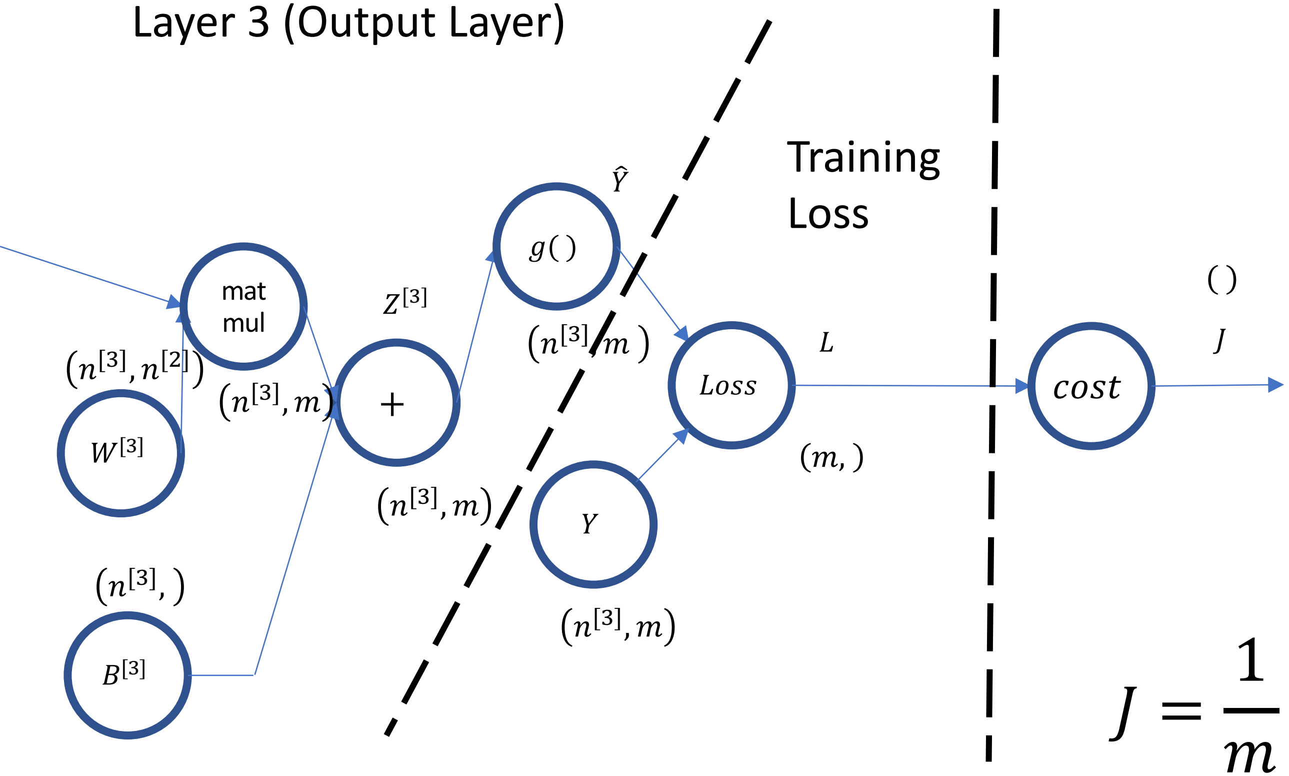
$W^{[3]}$ has shape (3, 4)

$B^{[3]}$ has shape (3,)

$\hat{Y} = A^{[3]}$ has shape (3,)

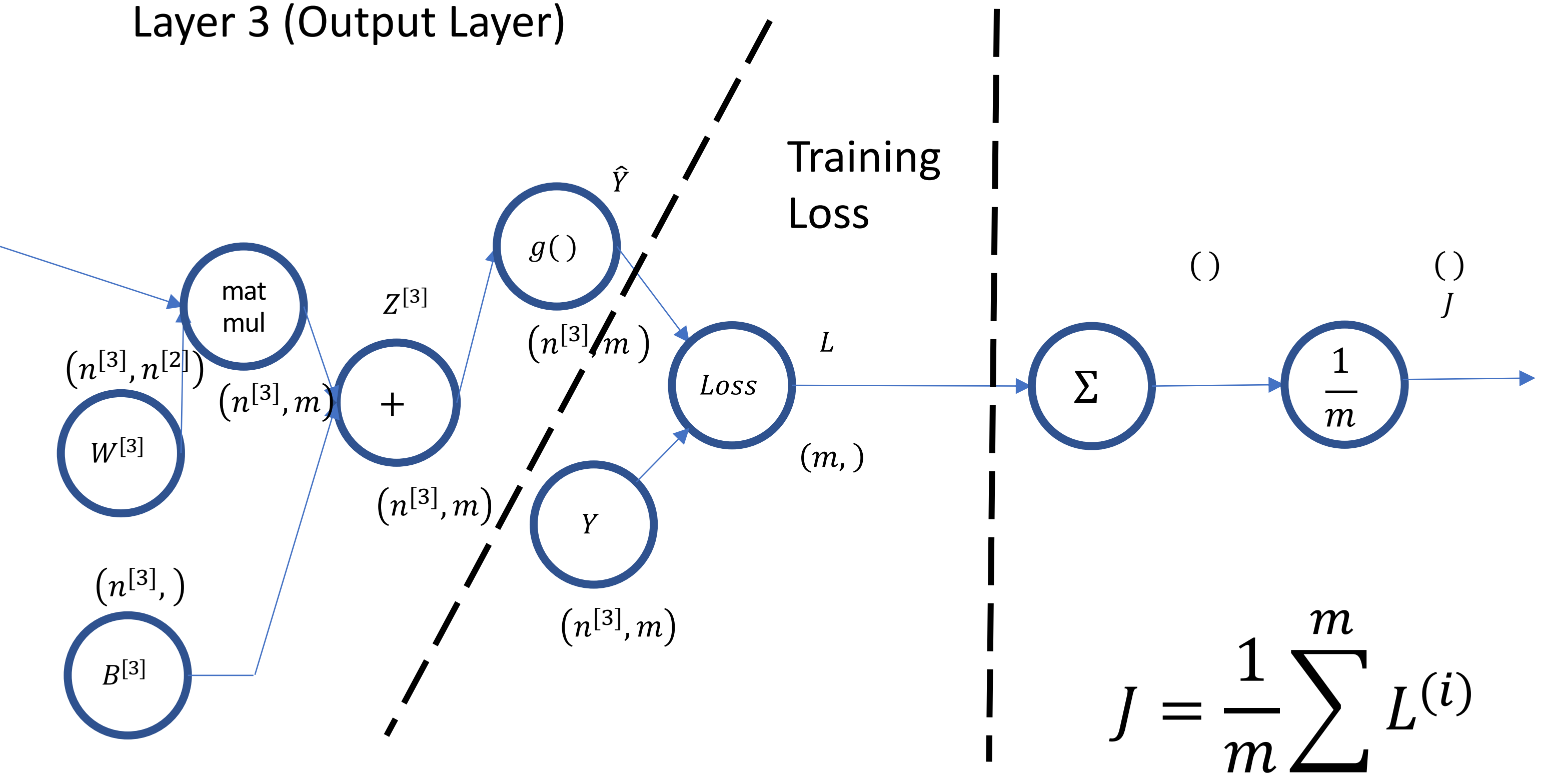
$Z^{[3]}$ has shape (3,)

Layer 3 (Output Layer)



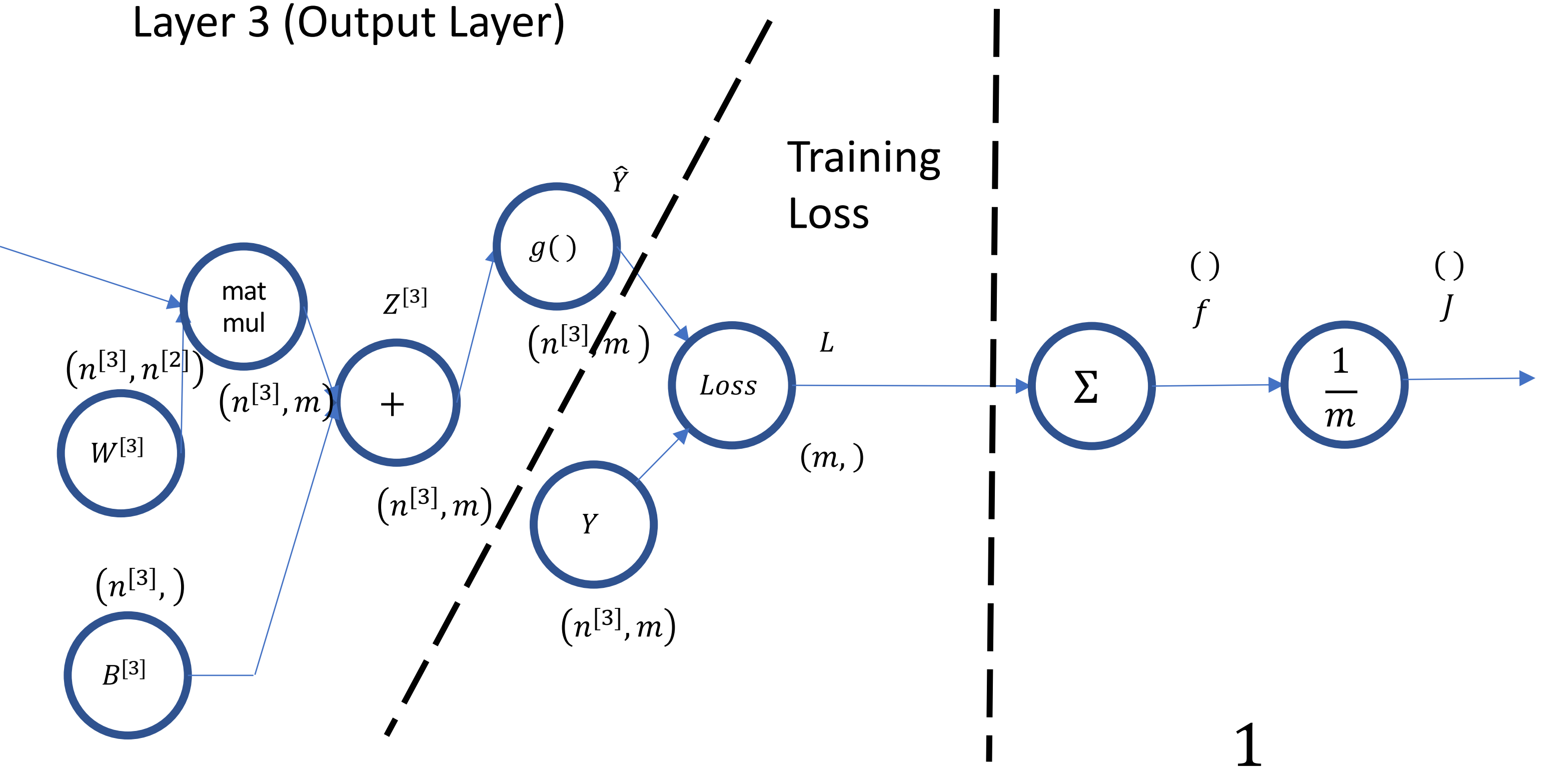
$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

Layer 3 (Output Layer)



$$J = \frac{1}{m}\sum_{i=1}^{m} L^{(i)}$$

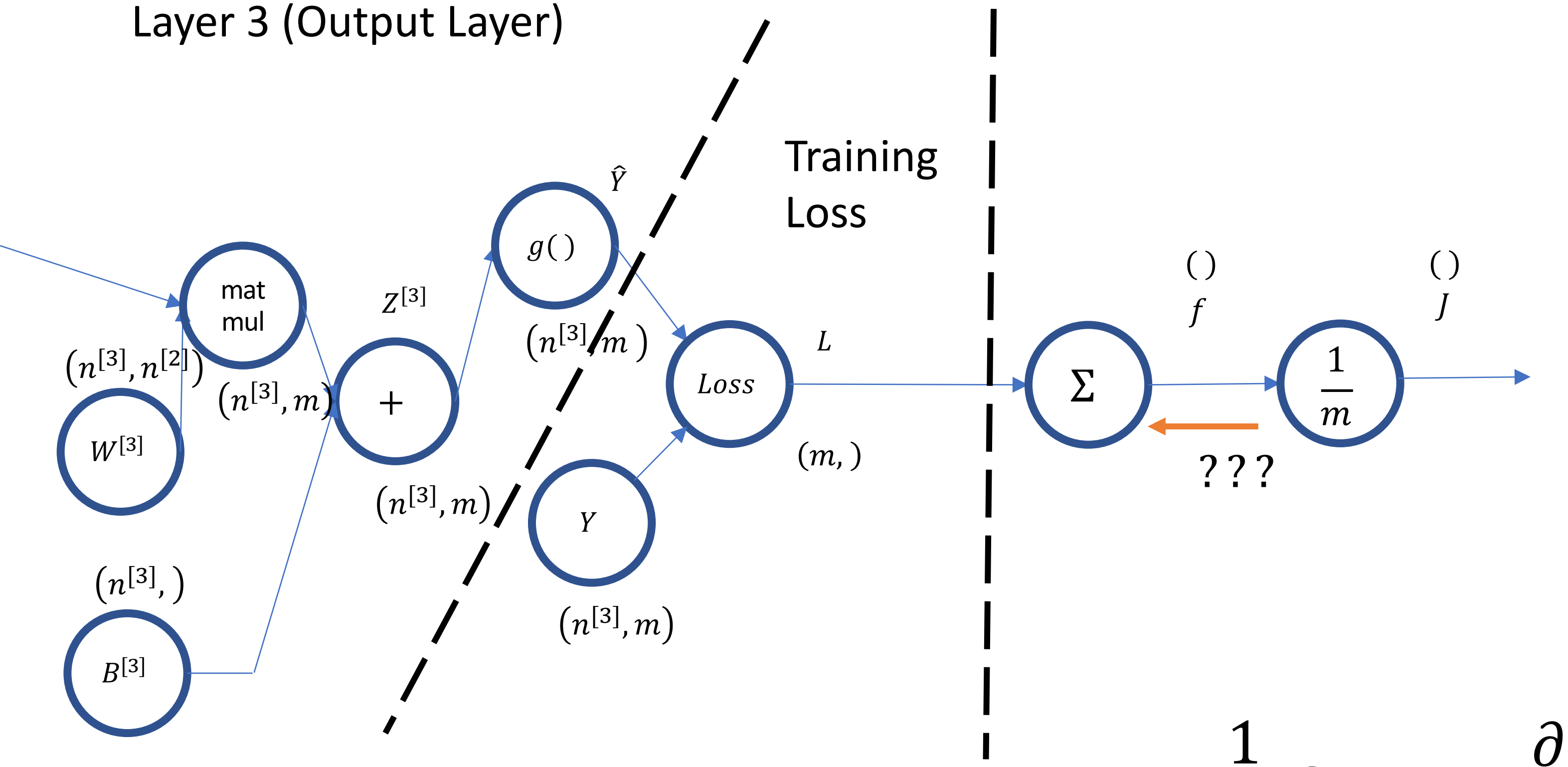
Layer 3 (Output Layer)



$$J = \frac{1}{m} f$$

$$f = \sum_{i=1}^{m} L^{(i)}$$

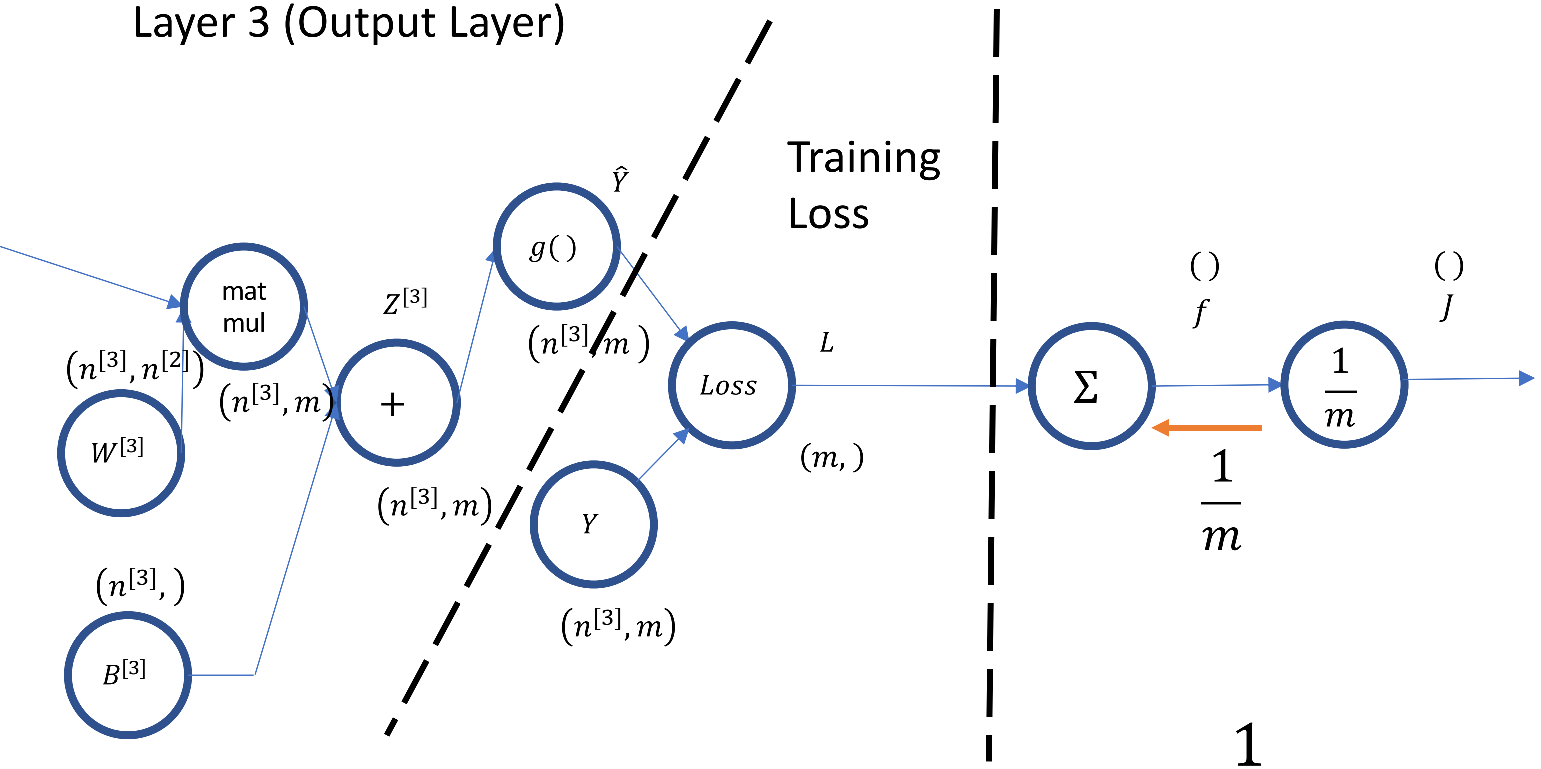
Layer 3 (Output Layer)



$$J = \frac{1}{m} f \qquad \frac{\partial J}{\partial f} = ???$$

$$f = \sum_{i=1}^{m} L^{(i)}$$

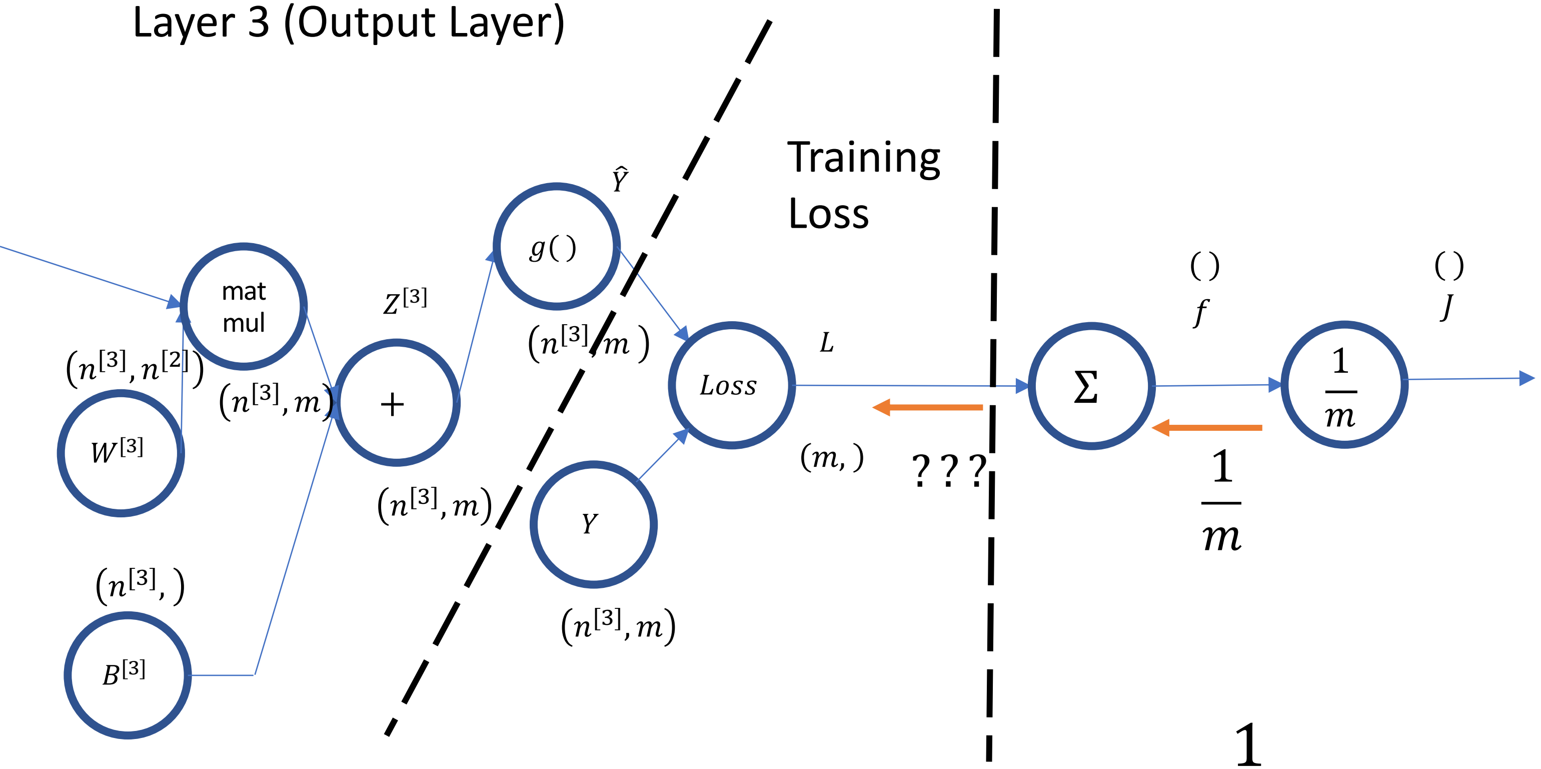
Layer 3 (Output Layer)



$$J = \frac{1}{m} f \qquad \frac{\partial J}{\partial f} = \frac{1}{m}$$

$$f = \sum_{i=1}^{m} L^{(i)}$$

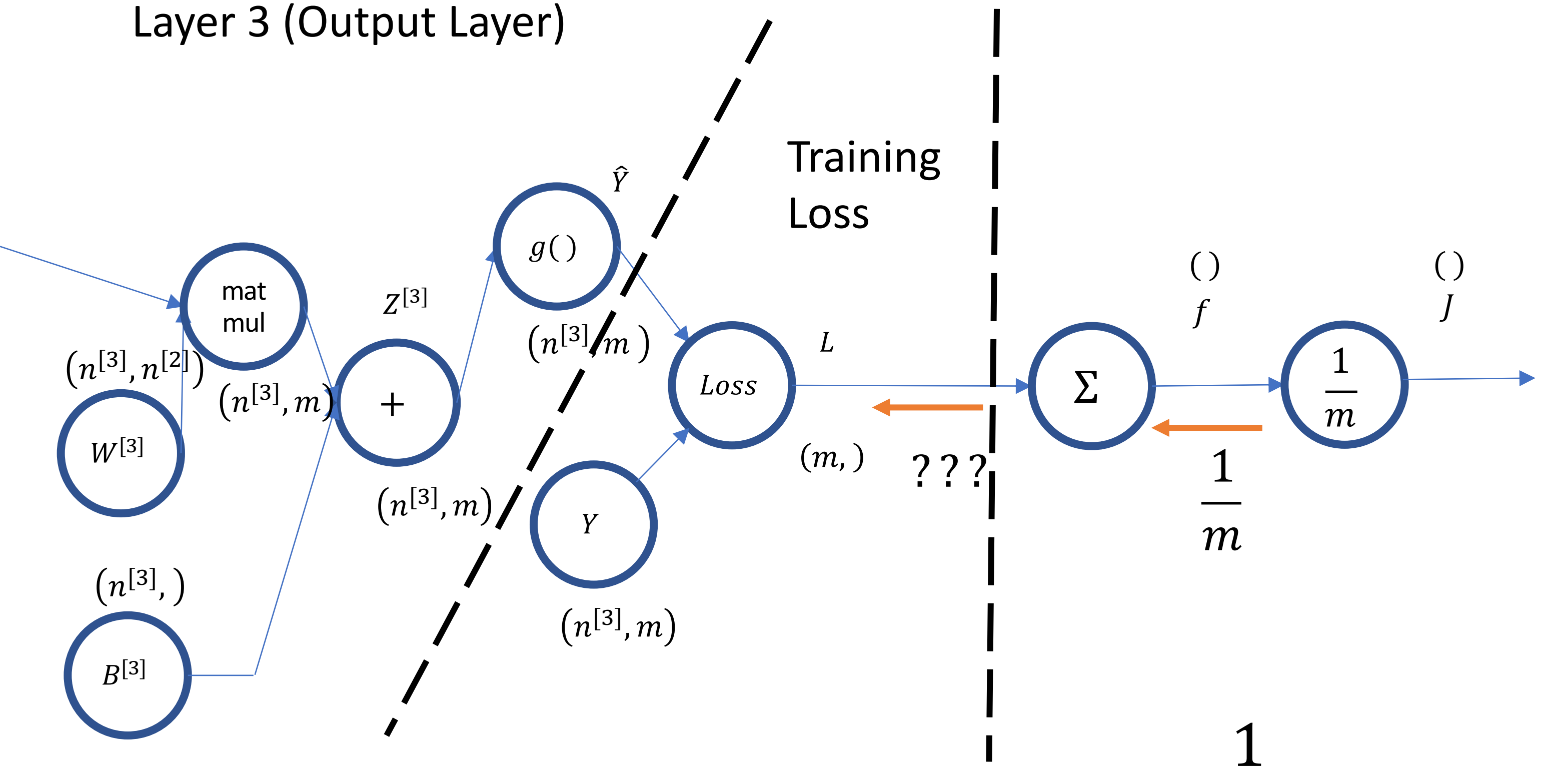
Layer 3 (Output Layer)



$$J = \frac{1}{m} f \qquad \frac{\partial J}{\partial f} = \frac{1}{m}$$

$$f = \sum_{i=1}^{m} L^{(i)} \qquad \frac{\partial J}{\partial L} = \frac{\partial f}{\partial L} \frac{\partial J}{\partial f}$$

Layer 3 (Output Layer)

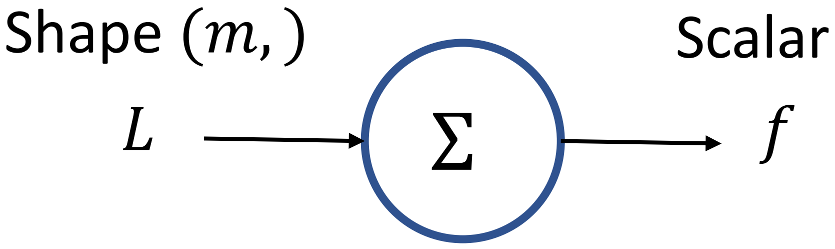


$$J = \frac{1}{m} f \qquad \frac{\partial J}{\partial f} = \frac{1}{m}$$

$$f = \sum_{i=1}^{m} L^{(i)} \qquad \frac{\partial J}{\partial L} = \frac{\partial f}{\partial L} \frac{\partial J}{\partial f} \qquad \frac{\partial f}{\partial L} = ???$$

Summation is just addition

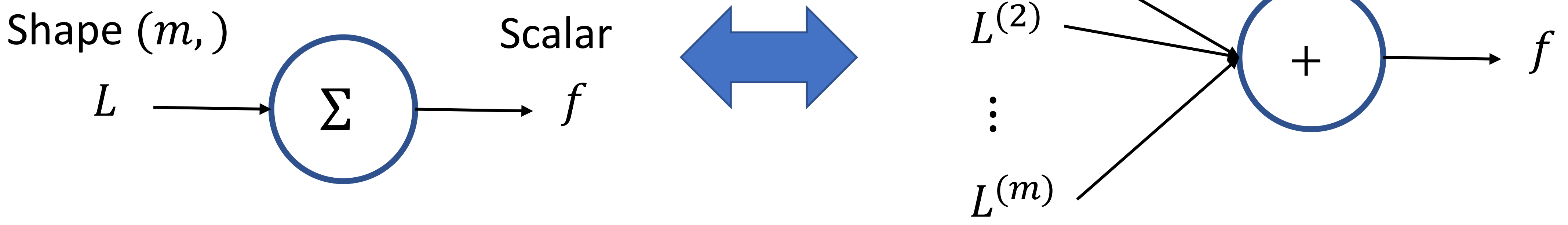
$$f = \sum_{i=1}^{m} L^{(i)}$$



Summation is just addition

$$f = \sum_{i=1}^{m} L^{(i)}$$

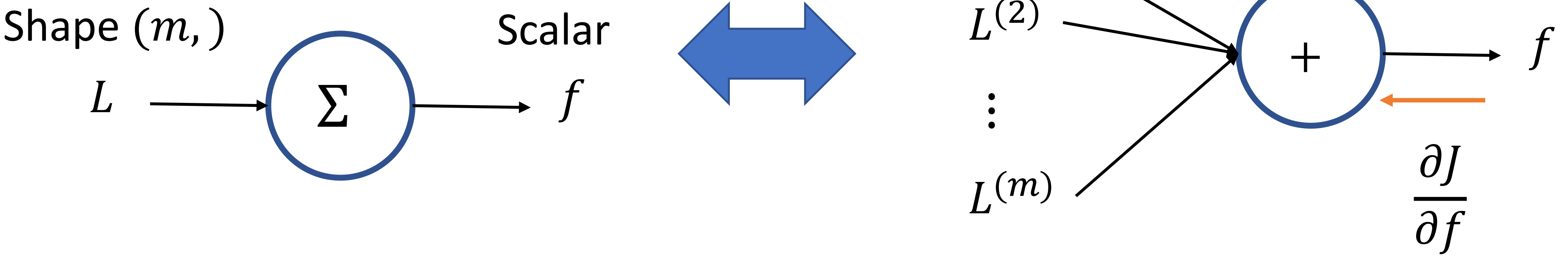
$$f = L^{(1)} + L^{(2)} + \cdots + L^{(m)}$$



Summation is just addition

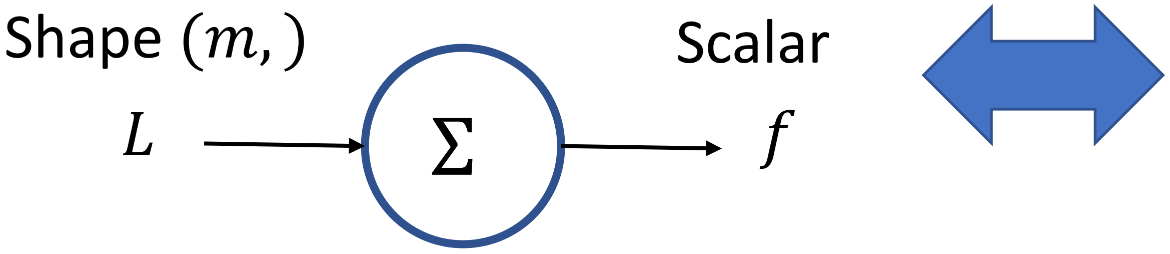
$$f = \sum_{i=1}^{m} L^{(i)}$$

$$f = L^{(1)} + L^{(2)} + \cdots + L^{(m)}$$

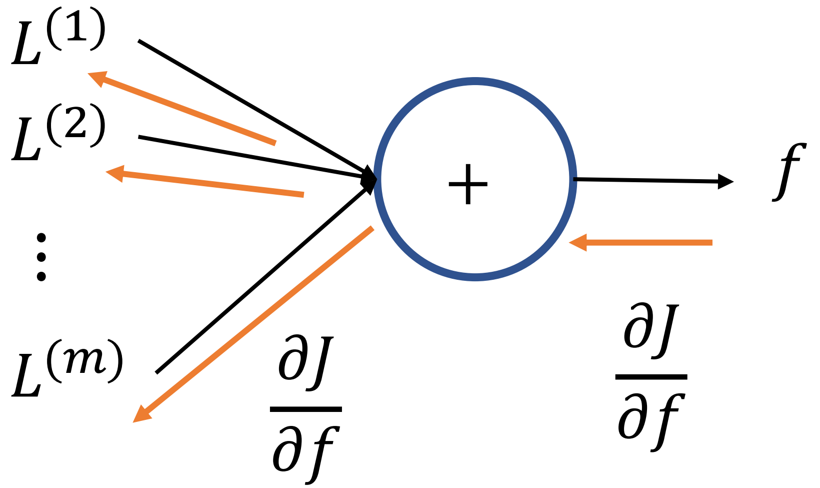


Summation is just addition

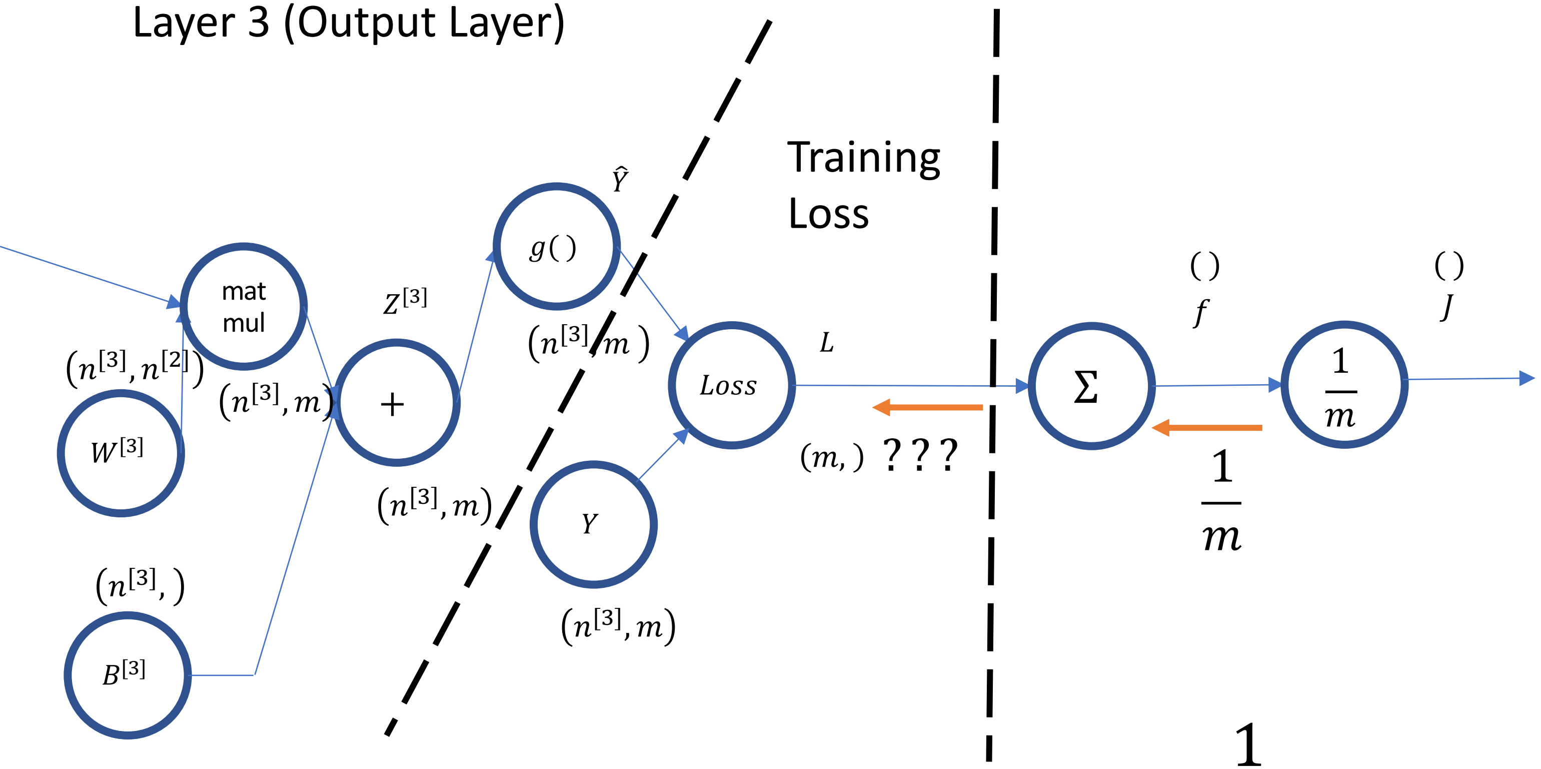
$$f = \sum_{i=1}^{m} L^{(i)}$$



$$f = L^{(1)} + L^{(2)} + \cdots + L^{(m)}$$


$L^{(1)}$
$L^{(2)}$
$\vdots$
$L^{(m)}$
$+$
f
$\frac{\partial J}{\partial f}$
$\frac{\partial J}{\partial f}$

Layer 3 (Output Layer)

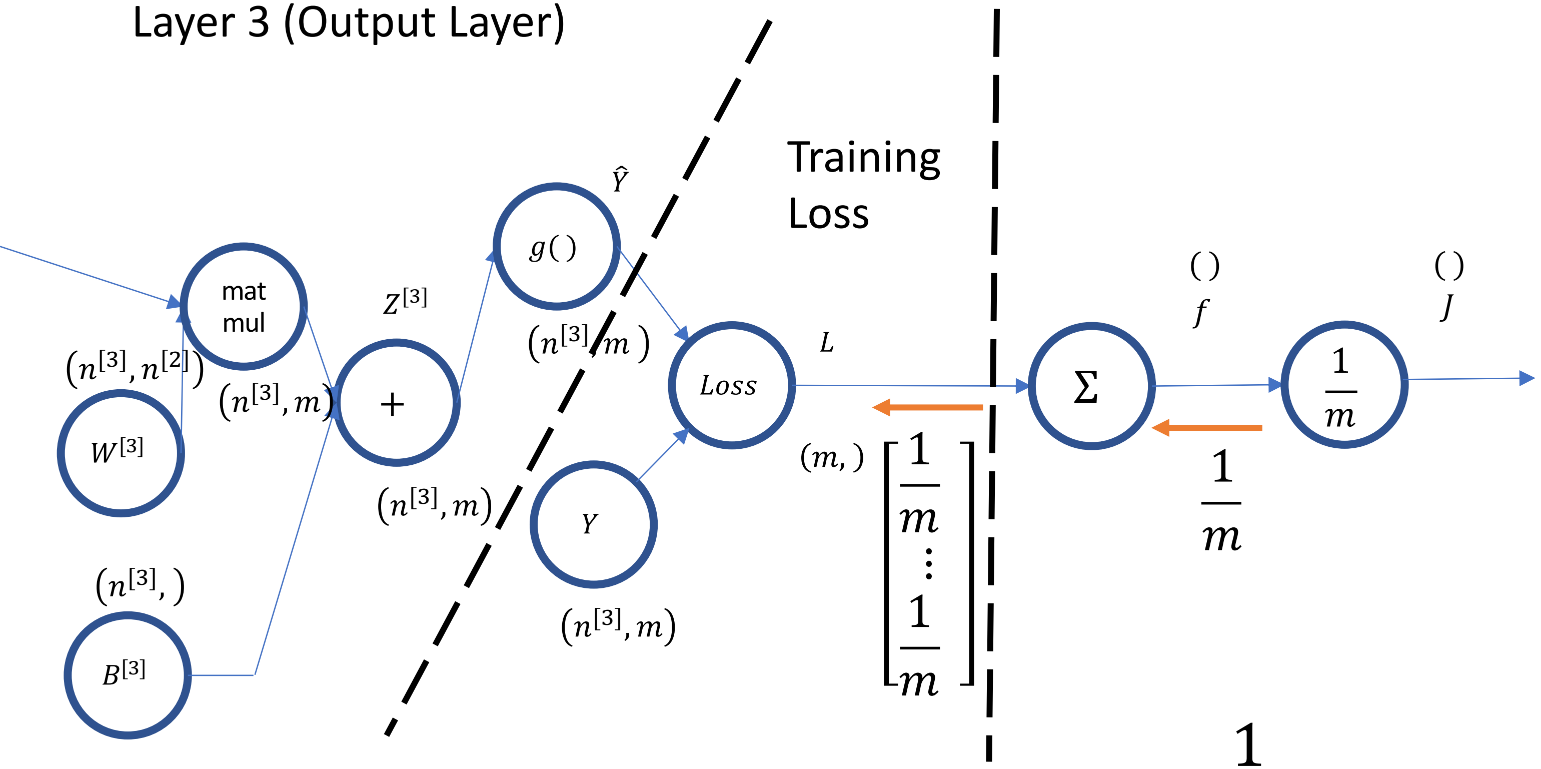


$$J = \frac{1}{m} f$$

$$\frac{\partial J}{\partial f} = \frac{1}{m}$$

$$f = \sum_{i=1}^{m} L^{(i)}$$

Layer 3 (Output Layer)



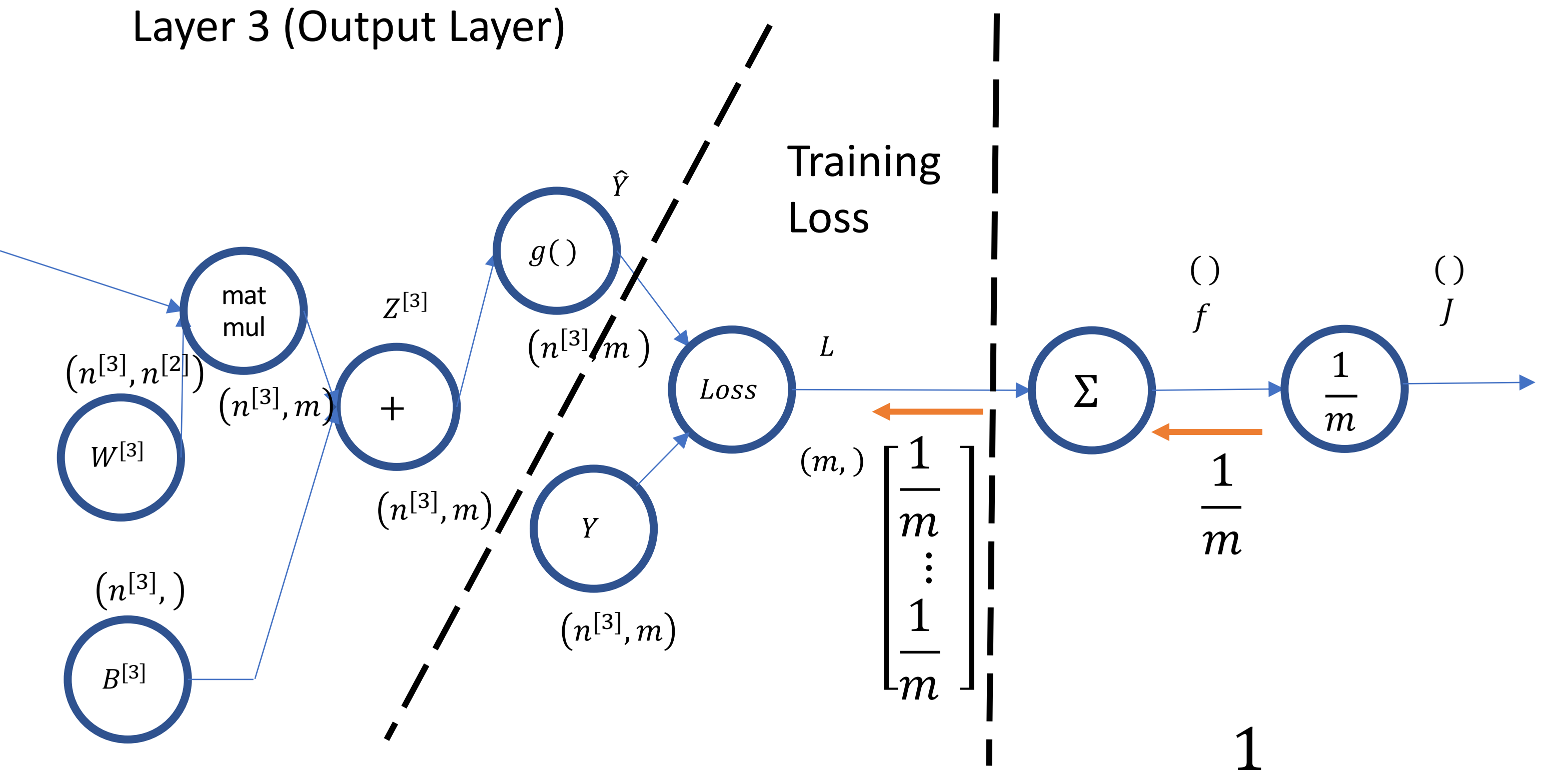
$$J = \frac{1}{m} f$$

$$\frac{\partial J}{\partial f} = \frac{1}{m}$$

$$f = \sum_{i=1}^{m} L^{(i)}$$

$$\frac{\partial J}{\partial L} = \begin{bmatrix} \frac{1}{m} \\ \vdots \\ \frac{1}{m} \end{bmatrix}$$

Shape (m,)



$$J = \frac{1}{m} f \qquad \frac{\partial J}{\partial f} = \frac{1}{m}$$

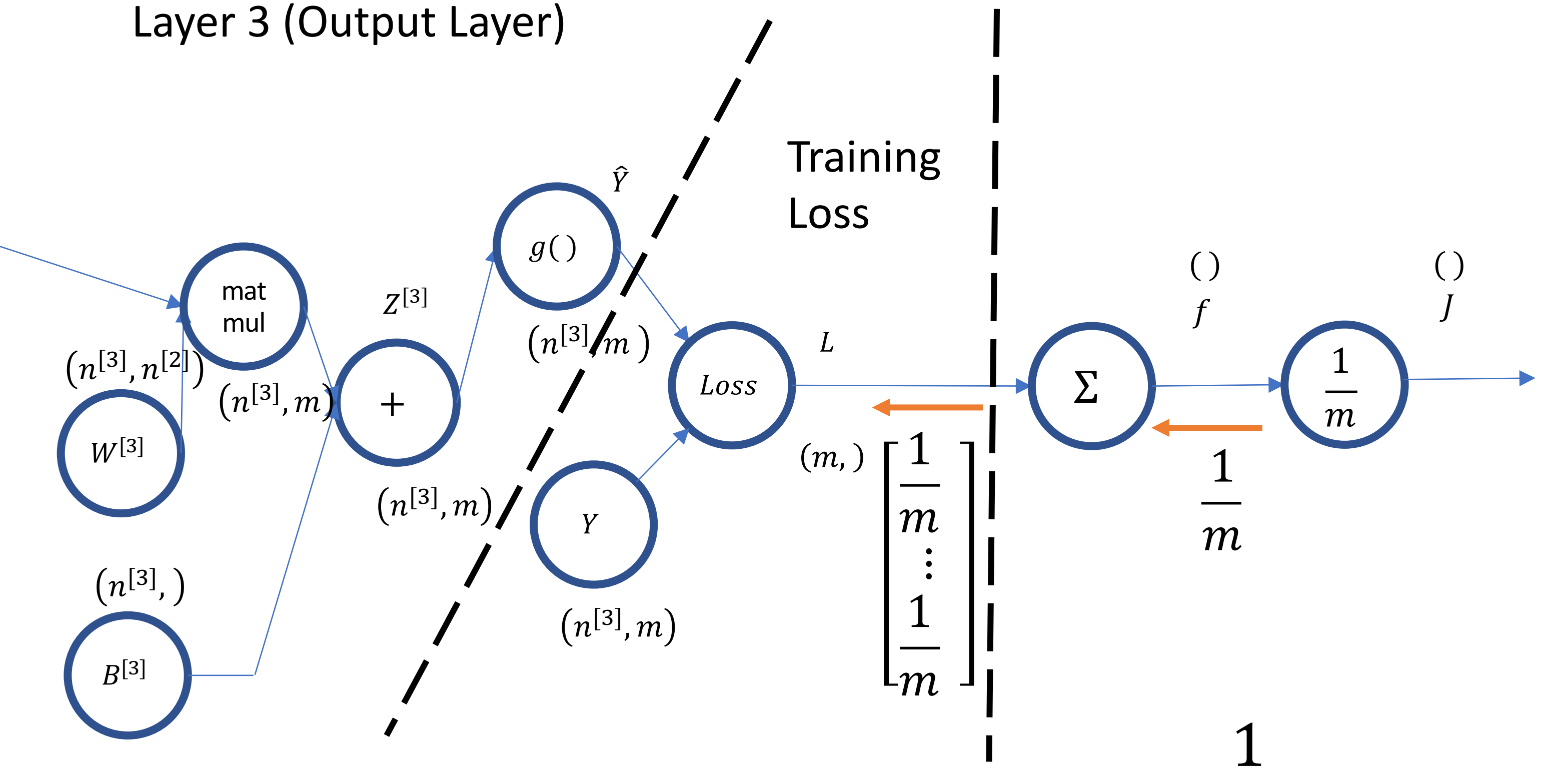
$$f = \sum_{i=1}^{m} L^{(i)}$$

$$\frac{\partial J}{\partial L} = \begin{bmatrix} \frac{1}{m} \\ \vdots \\ \frac{1}{m} \end{bmatrix}$$

Shape (m,)

- Downstream gradients will be scaled by 1/ m
- Each sample is only making a 1/ m contribution to the final cost

Layer 3 (Output Layer)



From Assignment 2 and Lecture 6

$$dW^{[2]} = \frac{1}{m} dZ^{[2]} A^{[1]T}$$

$$dW^{[1]} = \frac{1}{m} dZ^{[1]} X^{T}$$

$$J = \frac{1}{m} f \qquad \frac{\partial J}{\partial f} = \frac{1}{m}$$

$$f = \sum_{i=1}^{m} L^{(i)}$$

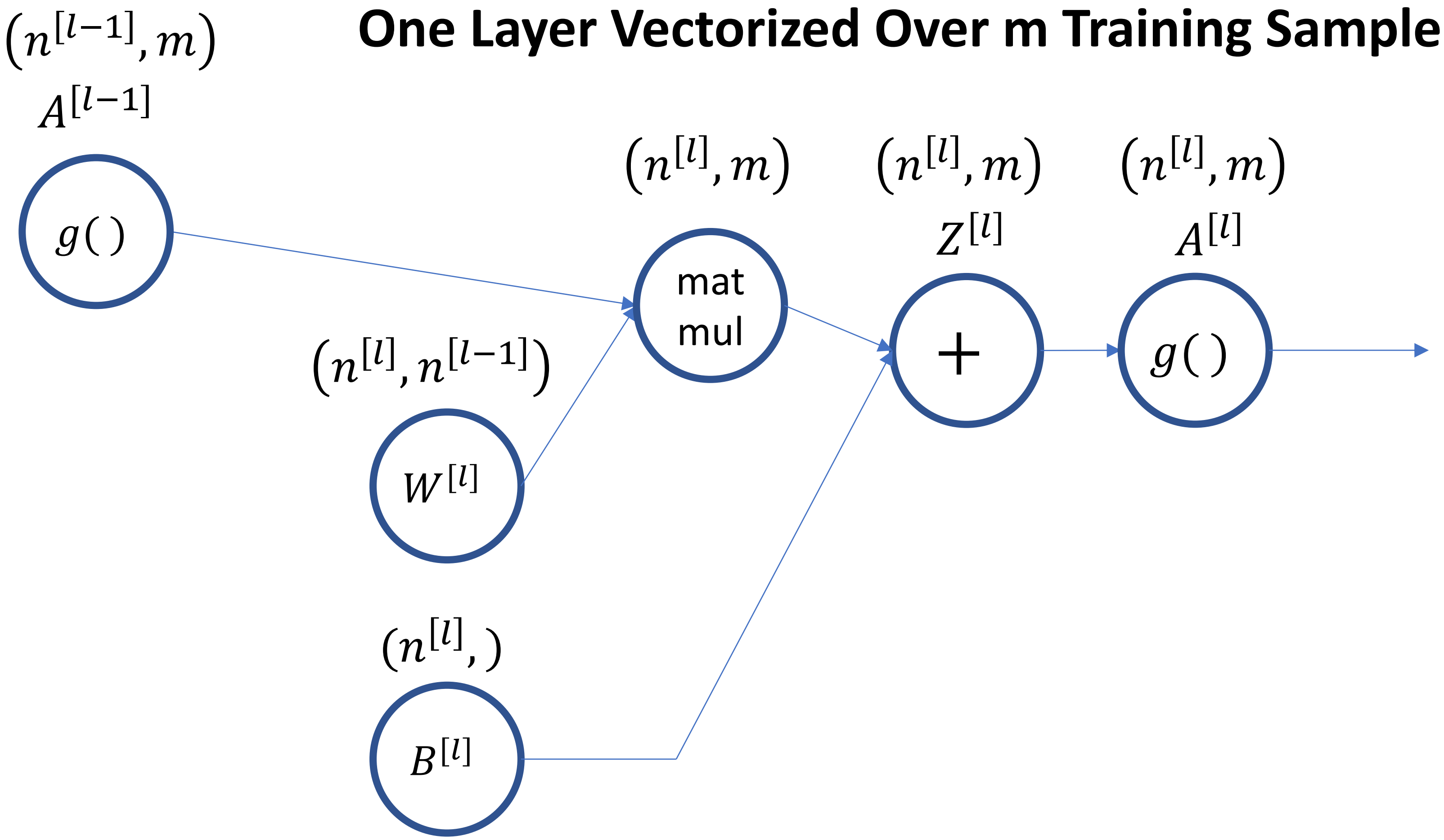
$$\frac{\partial J}{\partial L} = \begin{bmatrix}\frac{1}{m}\\ \vdots \\ \frac{1}{m}\end{bmatrix}$$

Shape (m,)

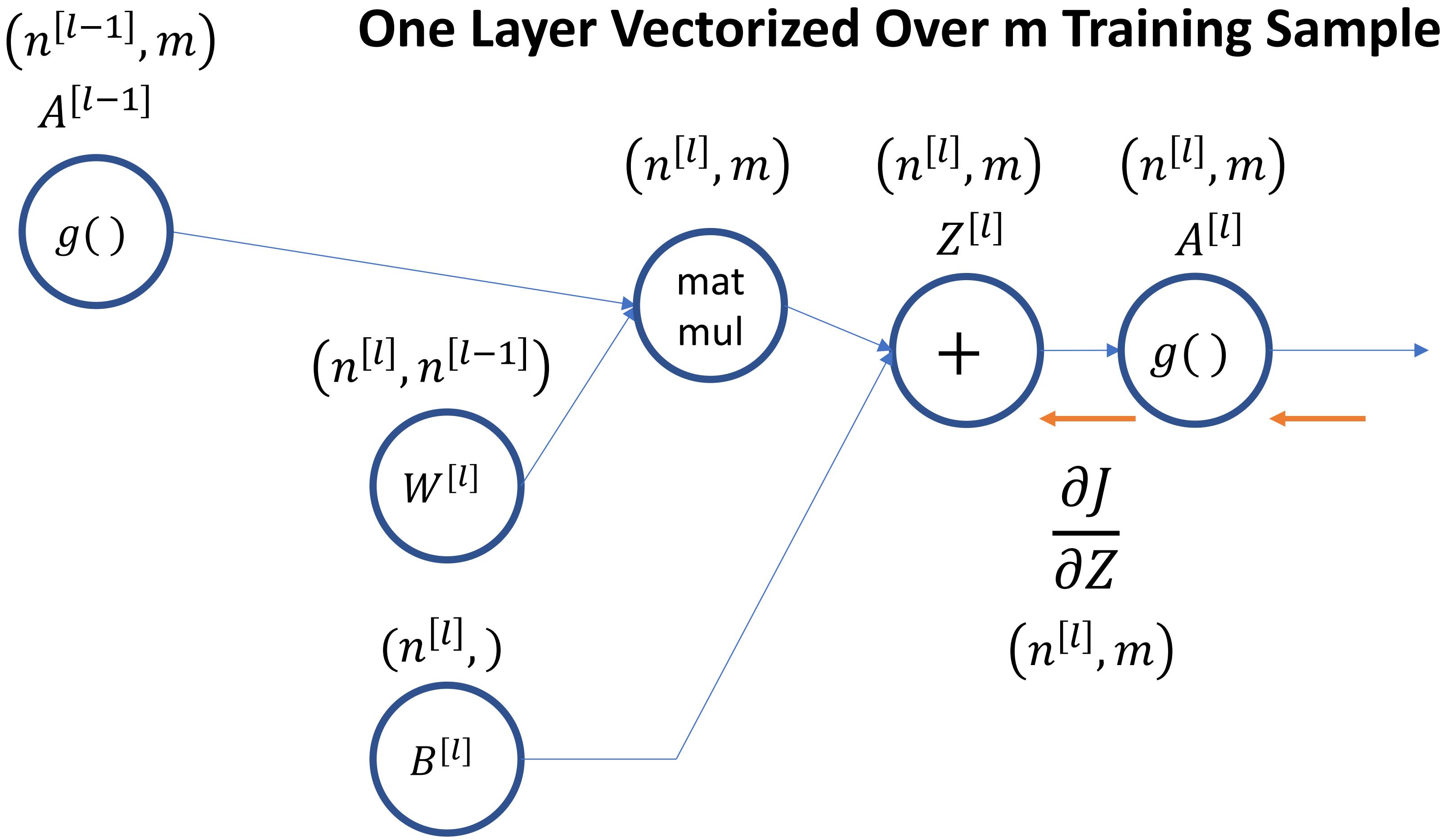
- Downstream gradients will be scaled by 1/ m
- Each sample is only making a 1/ m contribution to the final cost

Broadcasting
(Addition of the Bias)

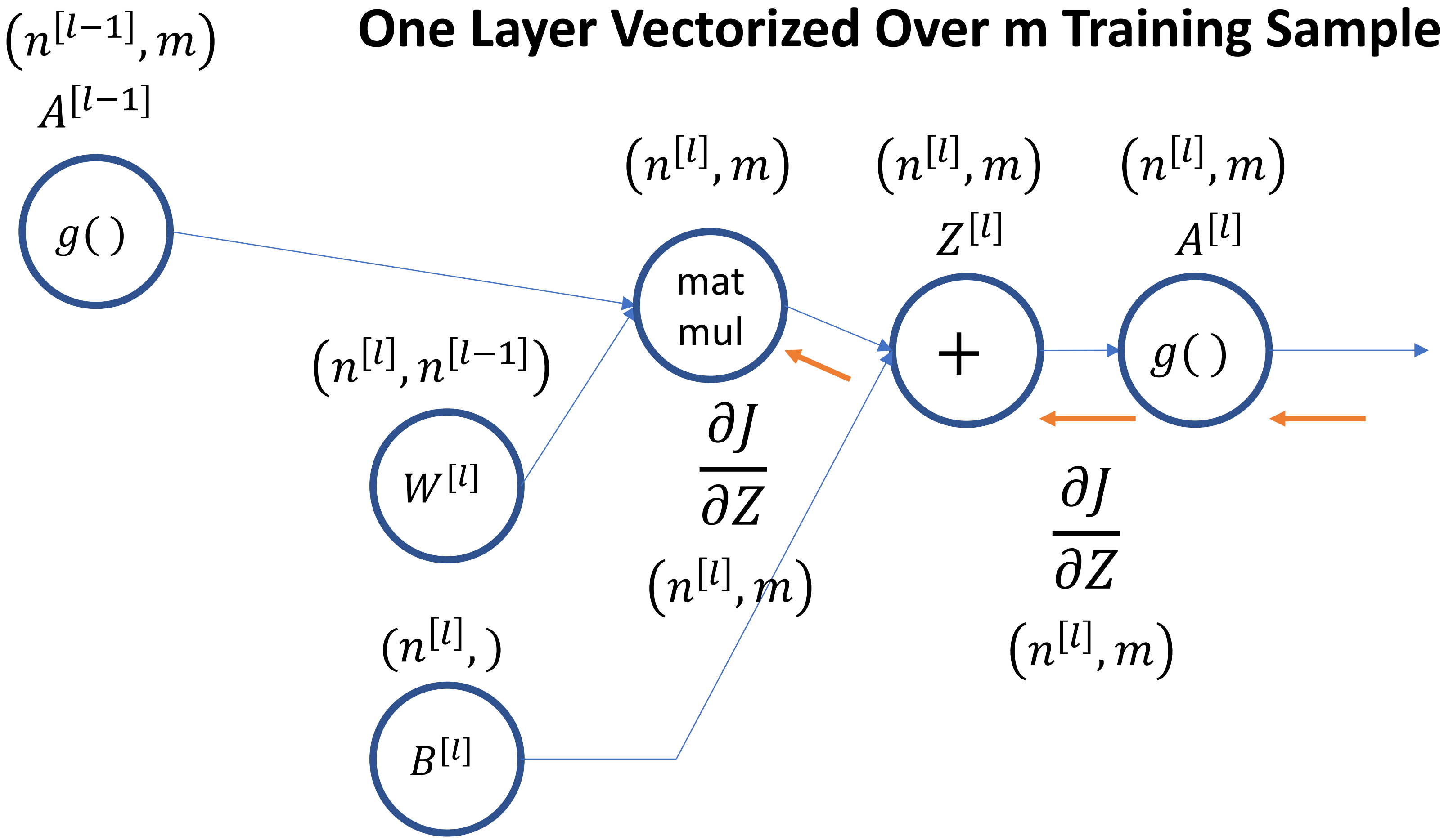
One Layer Vectorized Over m Training Samples



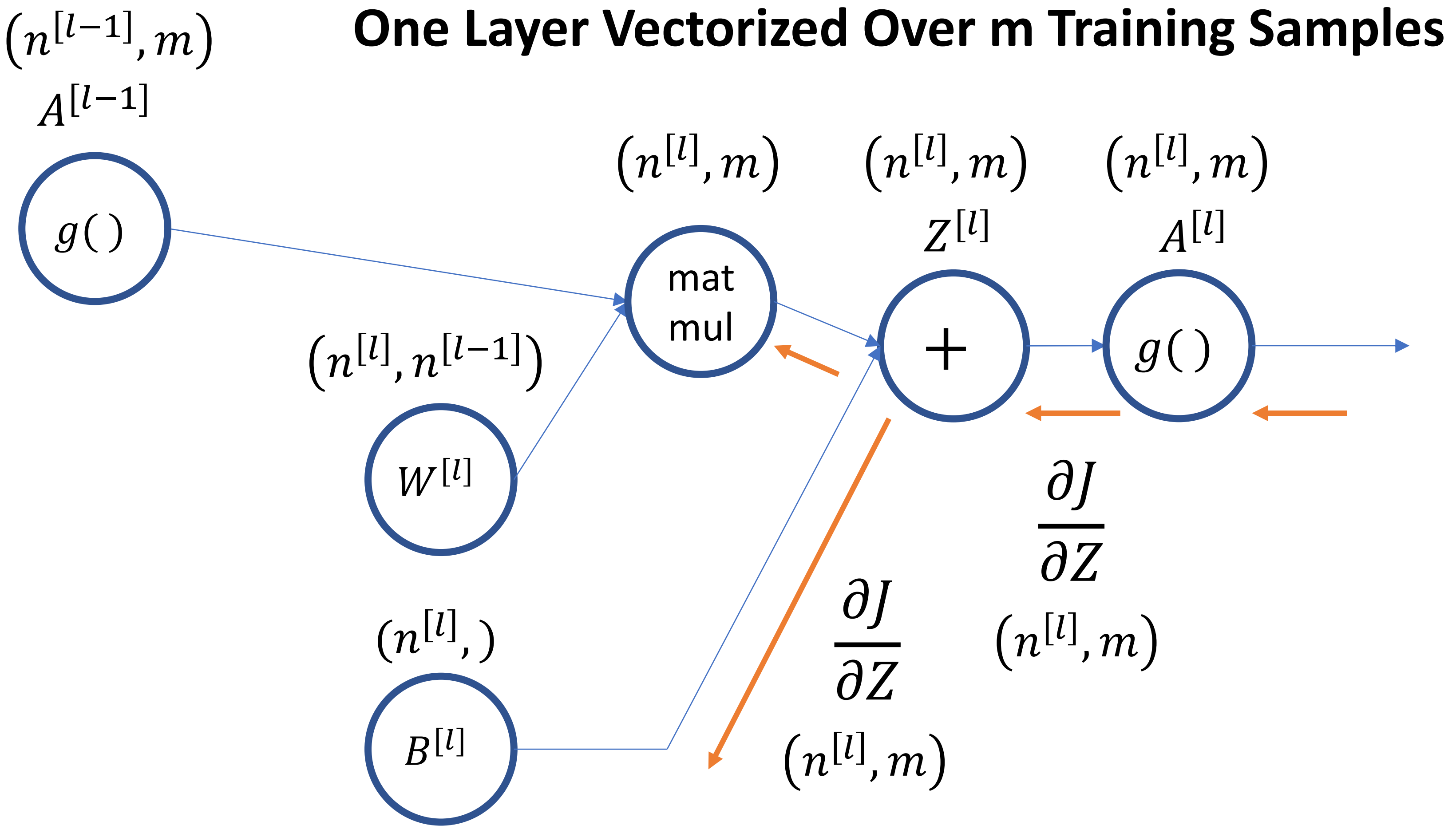
One Layer Vectorized Over m Training Samples



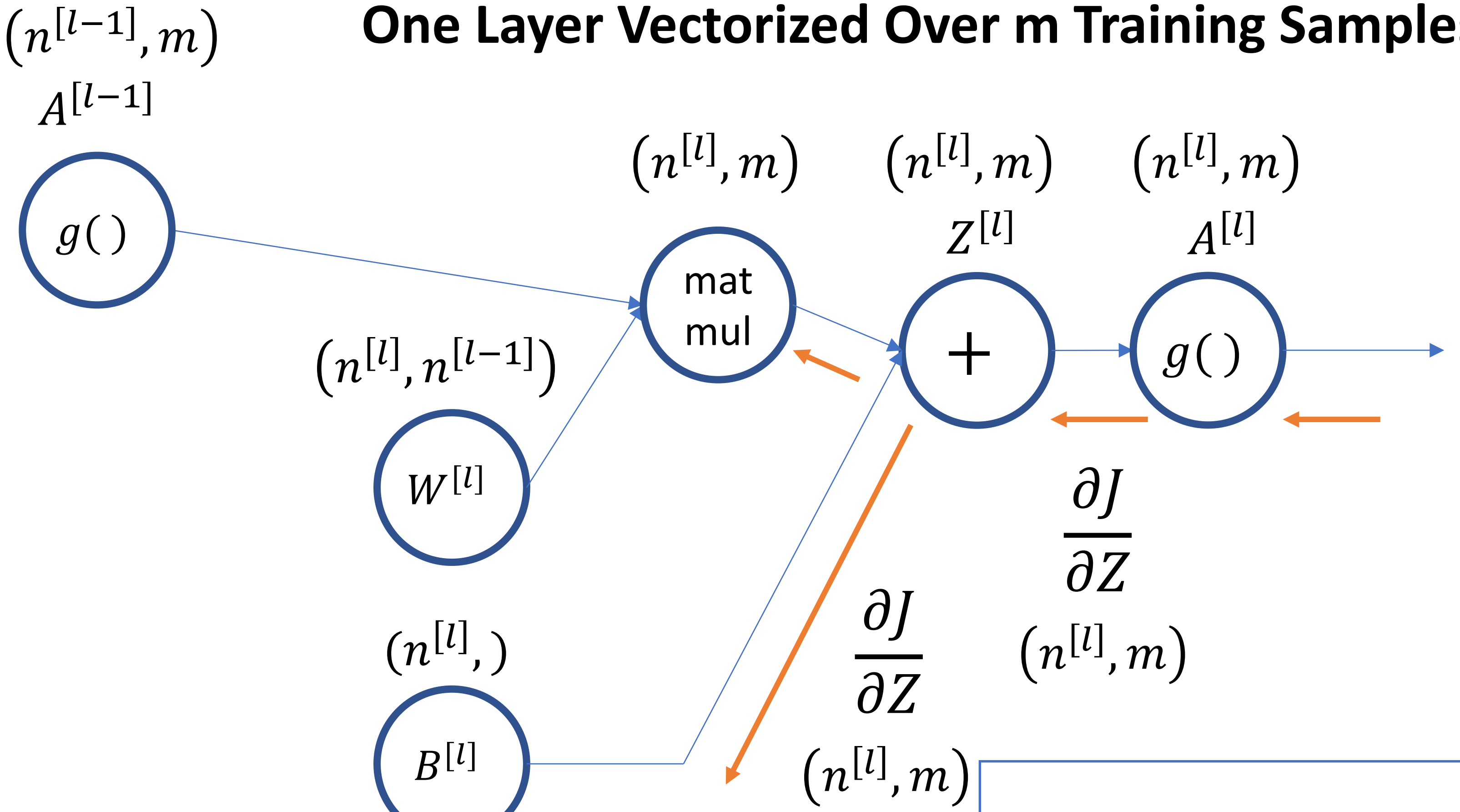
One Layer Vectorized Over m Training Samples



One Layer Vectorized Over m Training Samples



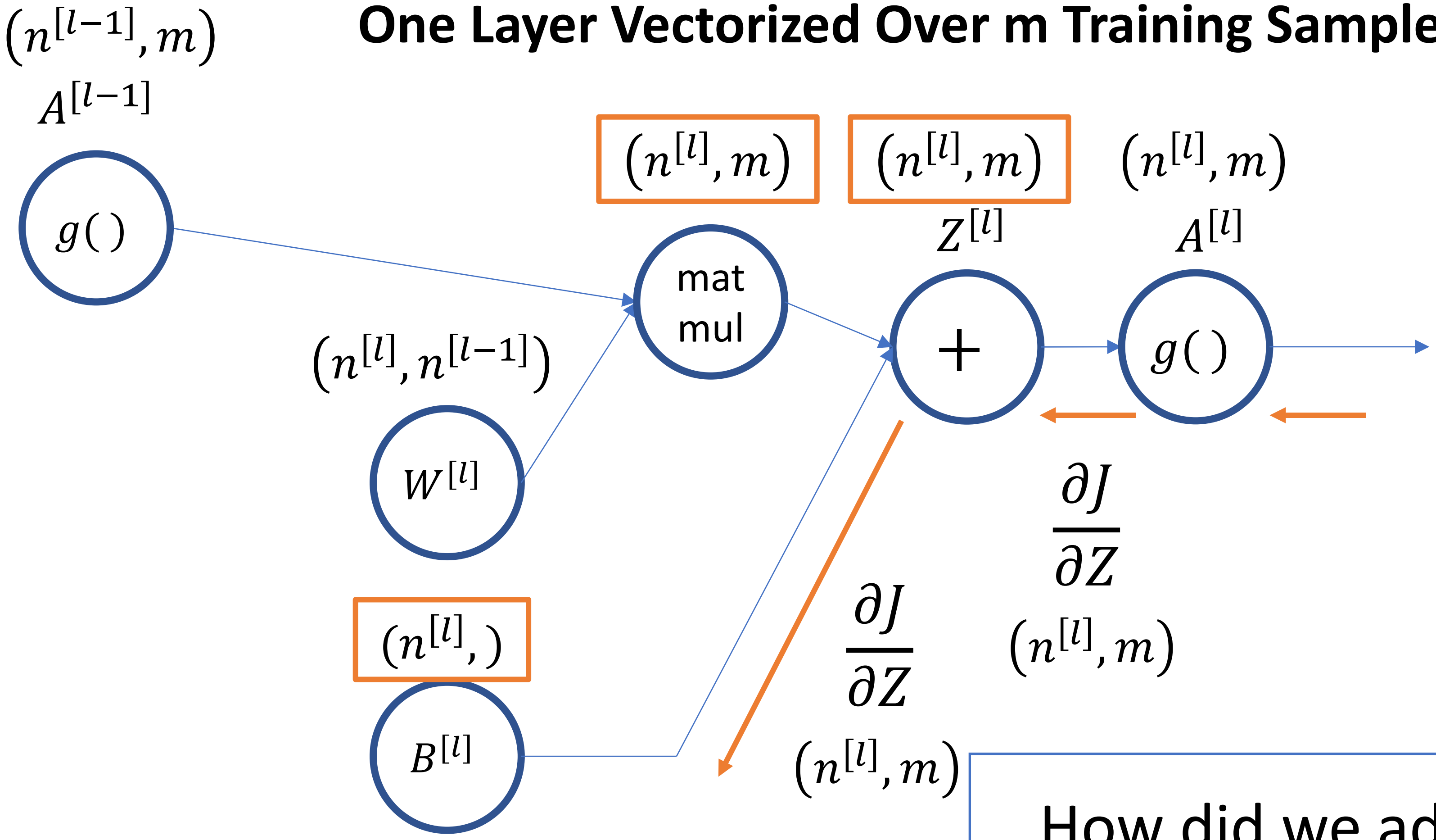
One Layer Vectorized Over m Training Samples



$$\frac{\partial J}{\partial B^{[l]}} = \frac{\partial J}{\partial Z}$$

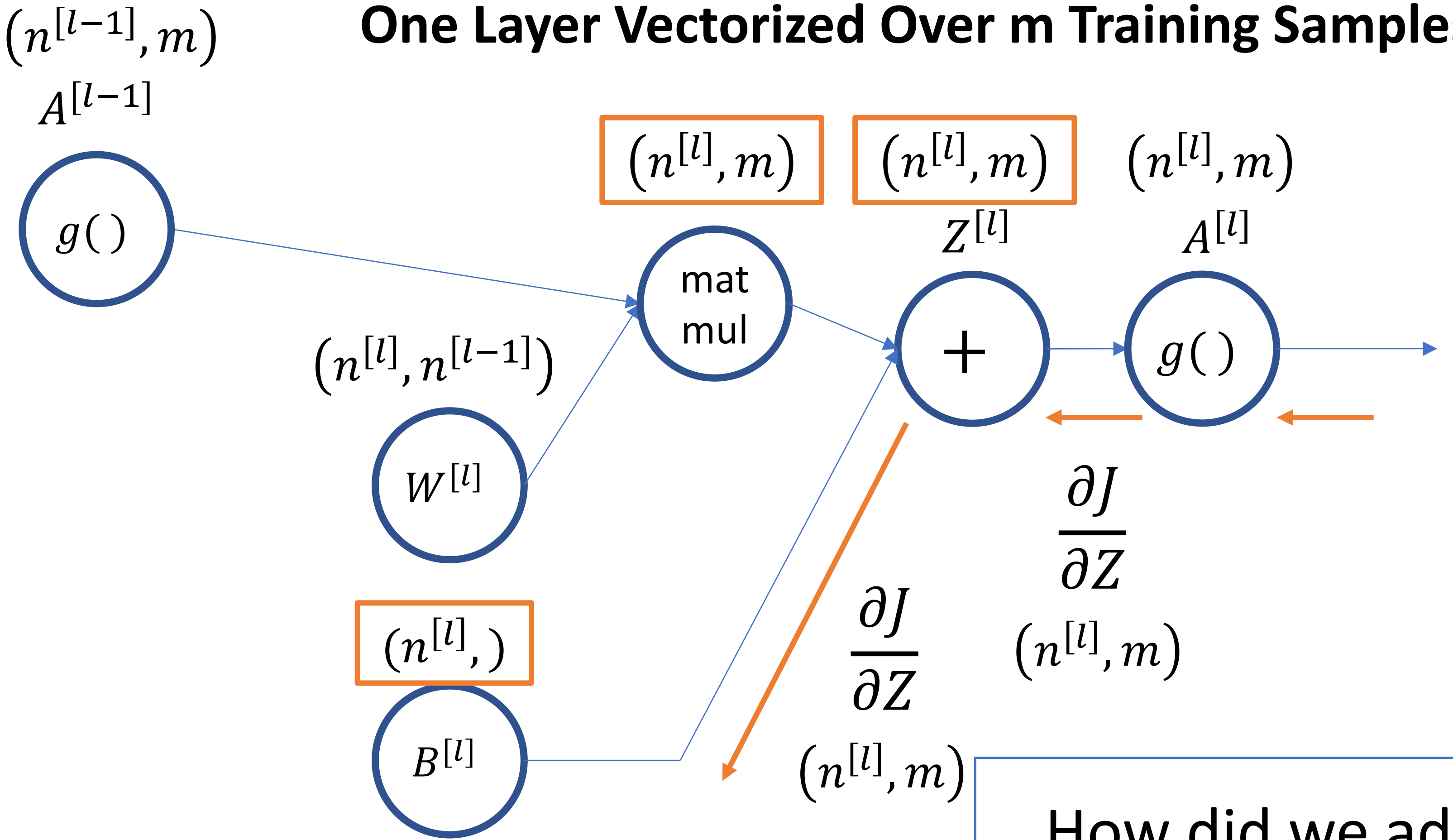
But, $B^{[l]}$ is shape $(n^{[l]},)$
and $\frac{\partial J}{\partial Z}$ is shape $(n^{[l]}, m)$

One Layer Vectorized Over m Training Samples



How did we add a $(n^{[l]}, m)$ tensor with a $(n^{[l]},)$ tensor to get a $(n^{[l]}, m)$ tensor?!

One Layer Vectorized Over m Training Samples



Broadcasting!

How did we add a $(n^{[l]}, m)$ tensor with a $(n^{[l]},)$ tensor to get a $(n^{[l]}, m)$ tensor?!

```
# In NumPy
Z2 = np.matmul(W2, A1) + B2
```

Broadcasting/Replicating

$$B^{[l]} = \begin{bmatrix} b_1^{[l]} \\ b_2^{[l]} \\ \vdots \\ b_{n^{[l]}}^{[l]} \end{bmatrix} \quad \Rightarrow \quad \overset{\text{Replicated } m \text{ times}}{\overleftrightarrow{[B^{[l]} \quad B^{[l]} \quad \dots \quad B^{[l]}]}} = \begin{bmatrix} b_1^{[l]} & b_1^{[l]} & & b_1^{[l]} \\ b_2^{[l]} & b_2^{[l]} & \dots & b_2^{[l]} \\ \vdots & \vdots & & \vdots \\ b_{n^{[l]}}^{[l]} & b_{n^{[l]}}^{[l]} & & b_{n^{[l]}}^{[l]} \end{bmatrix}$$

$(n^{[l]},)$

$(n^{[l]}, m)$

- So now all operands of the addition are shape $(n^{[l]}, m)$
- Intuition: When calculating activations on the layer, the same parameters are used for each of the m sample

Broadcasting/Replicating

Replicated m times

$$B^{[l]} = \begin{bmatrix} b_1^{[l]} \\ b_2^{[l]} \\ \vdots \\ b_{n^{[l]}}^{[l]} \end{bmatrix} \Rightarrow \overset{\longleftrightarrow}{[B^{[l]} \quad B^{[l]} \quad \dots \quad B^{[l]}]} = \begin{bmatrix} b_1^{[l]} & b_1^{[l]} & & b_1^{[l]} \\ b_2^{[l]} & b_2^{[l]} & \dots & b_2^{[l]} \\ \vdots & \vdots & & \vdots \\ b_{n^{[l]}}^{[l]} & b_{n^{[l]}}^{[l]} & & b_{n^{[l]}}^{[l]} \end{bmatrix}$$

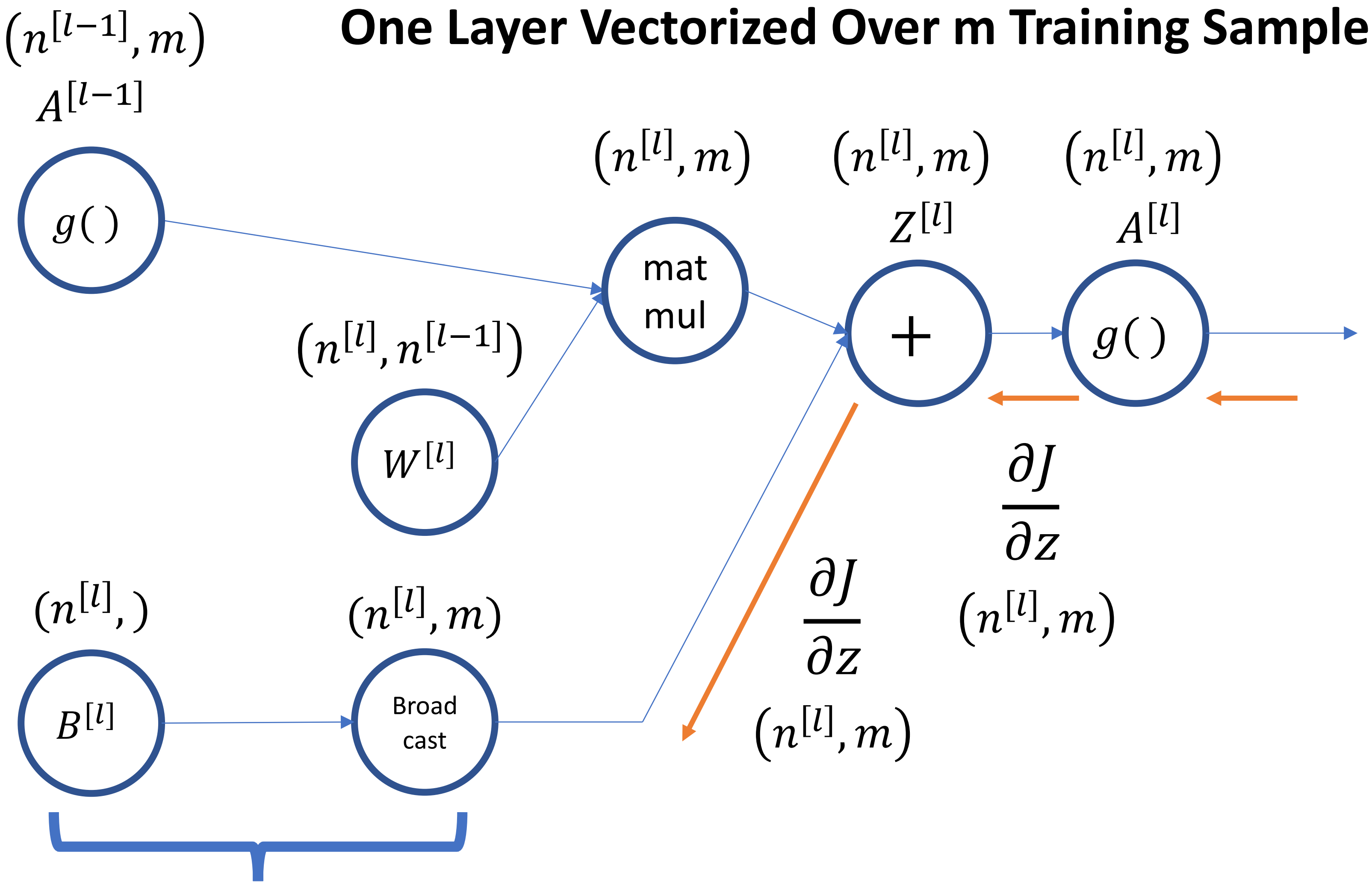
$$(n^{[l]},) \qquad\qquad (n^{[l]}, m)$$

$$Z^{[1]} = matmul\left(W^{[1]}, X\right) + B^{[1]}$$

$$= \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} \\ w_{3,1}^{[1]} & w_{3,2}^{[1]} \\ w_{4,1}^{[1]} & w_{4,2}^{[1]} \\ w_{5,1}^{[1]} & w_{5,2}^{[1]} \end{bmatrix} \begin{bmatrix} x_1^{(1)} & x_1^{(2)} \\ x_2^{(1)} & x_2^{(2)} \end{bmatrix} + \begin{bmatrix} b_1^{[1]} & b_1^{[1]} \\ b_2^{[1]} & b_2^{[1]} \\ b_3^{[1]} & b_3^{[1]} \\ b_4^{[1]} & b_4^{[1]} \\ b_5^{[1]} & b_5^{[1]} \end{bmatrix} \longleftarrow$$

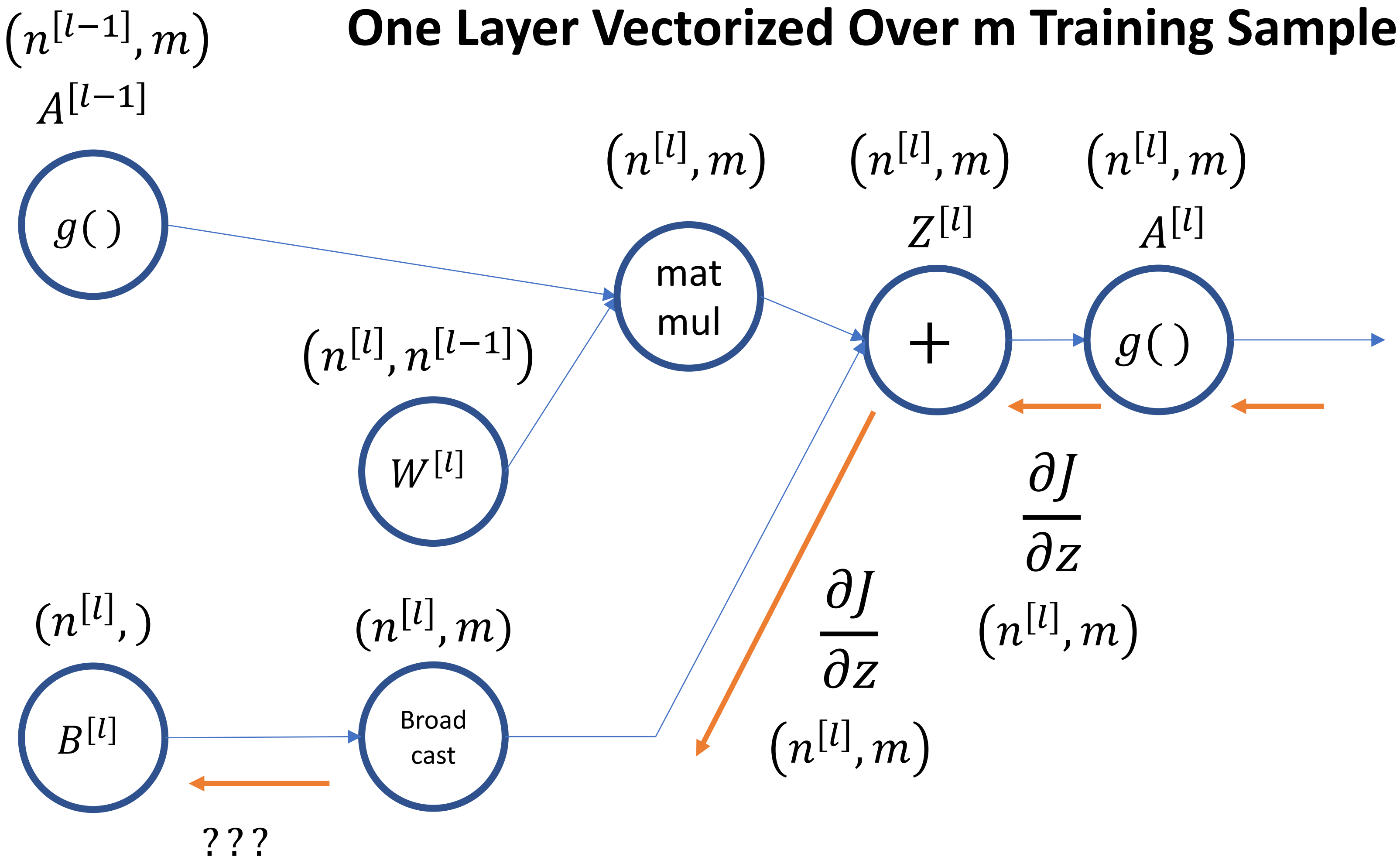
Recall from slide 51

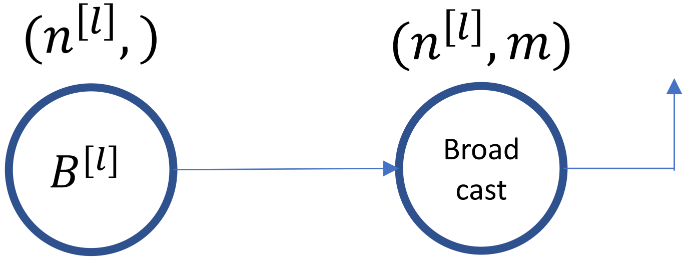
One Layer Vectorized Over m Training Samples



Modifying the compute graph to be a bit more explicit in what is happening

One Layer Vectorized Over m Training Samples



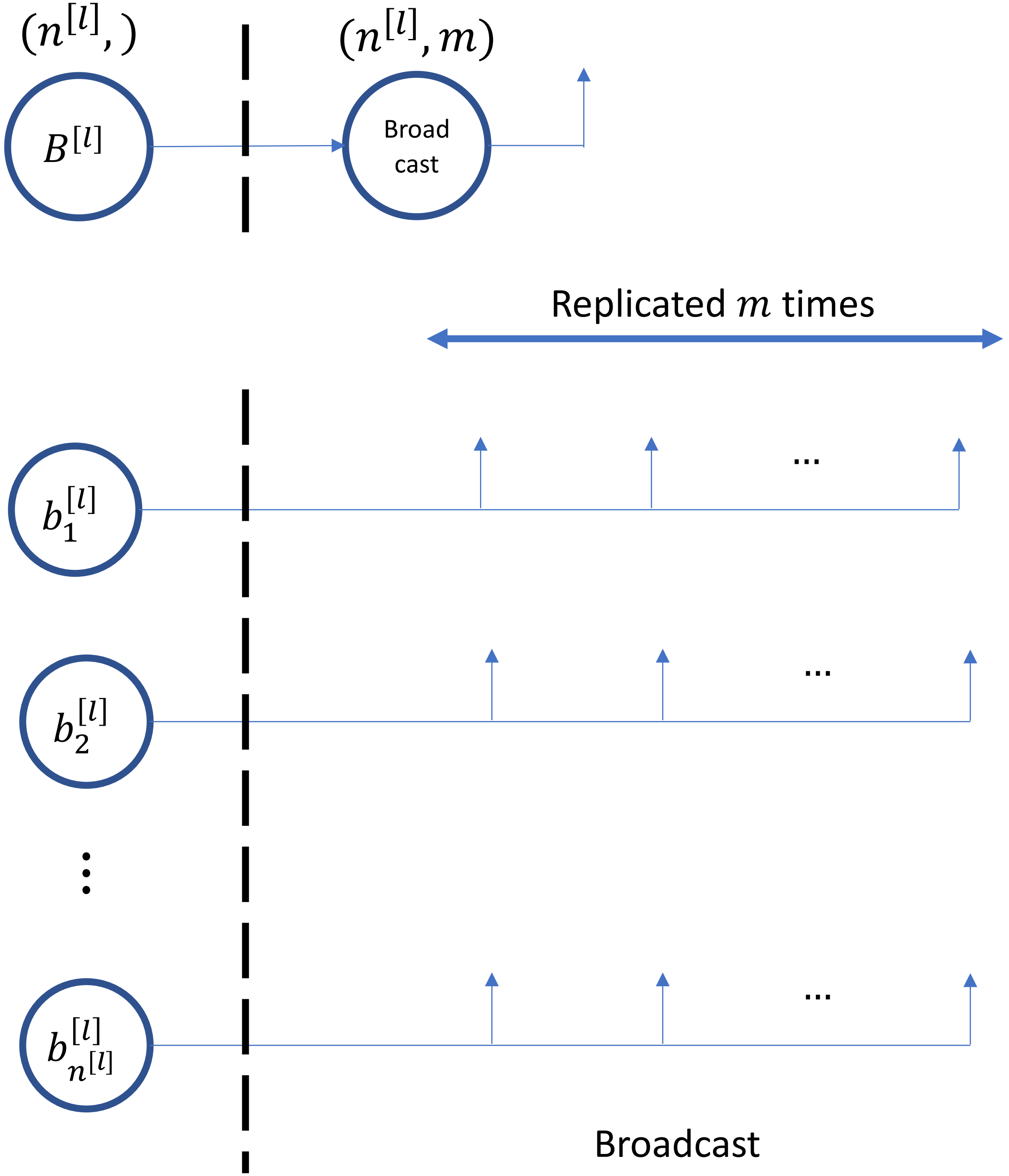


$(n^{[l]},)$

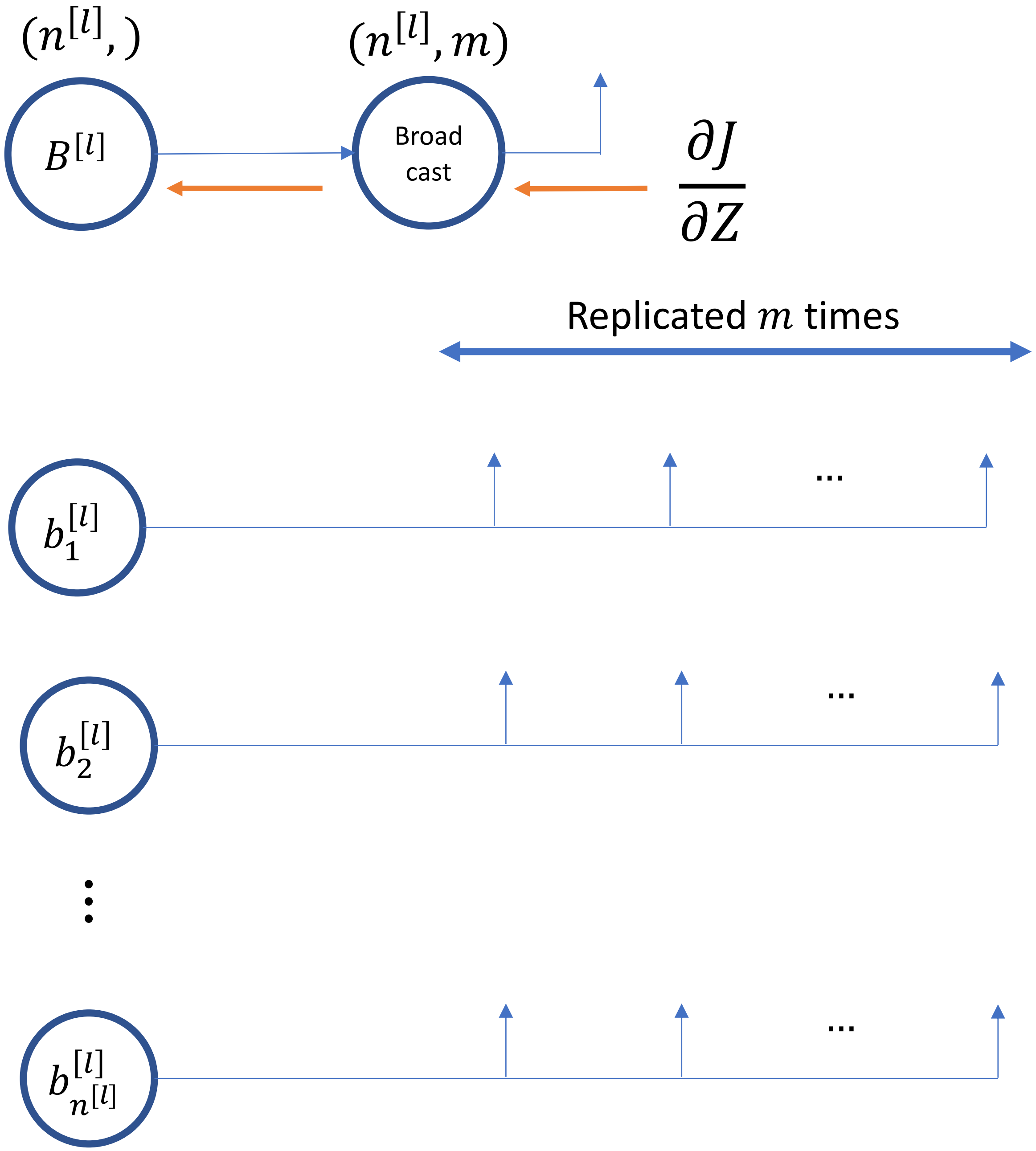
$B^{[l]}$

$(n^{[l]}, m)$

Broad cast



Broadcast

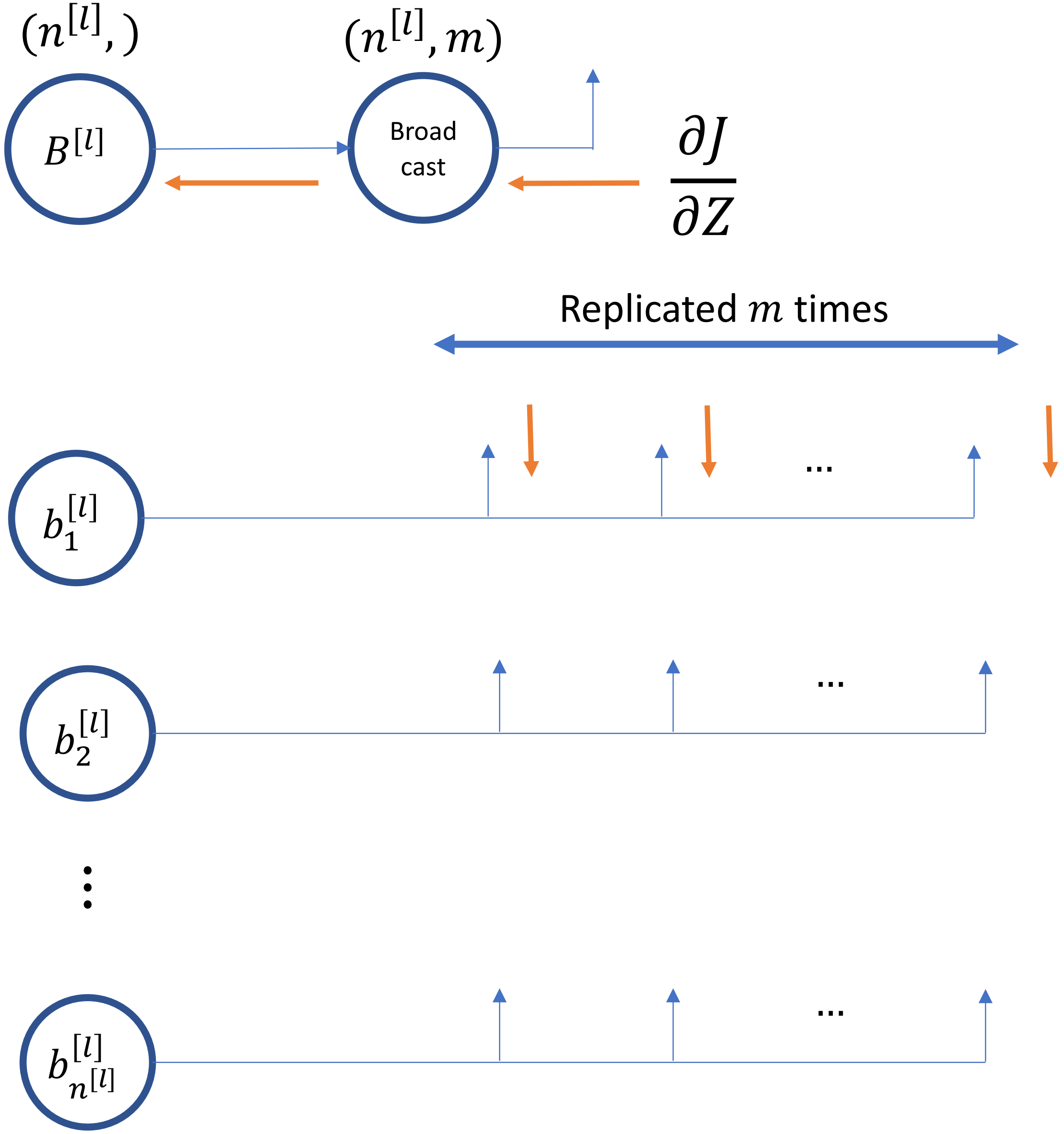


Broadcast

Each column is for one sample

Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$

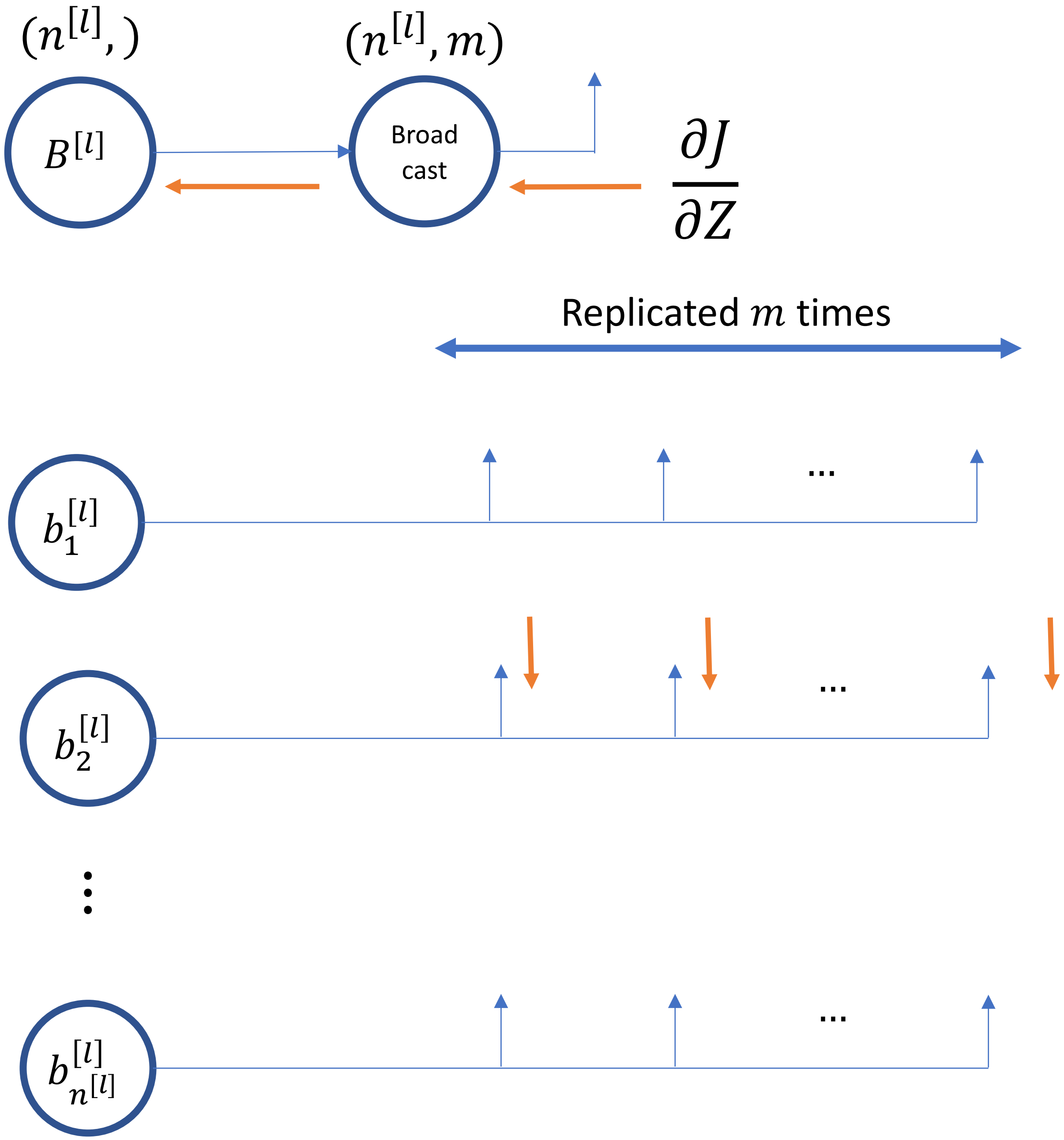


Broadcast

Each column is for one sample

Each row is for one unit of the layer

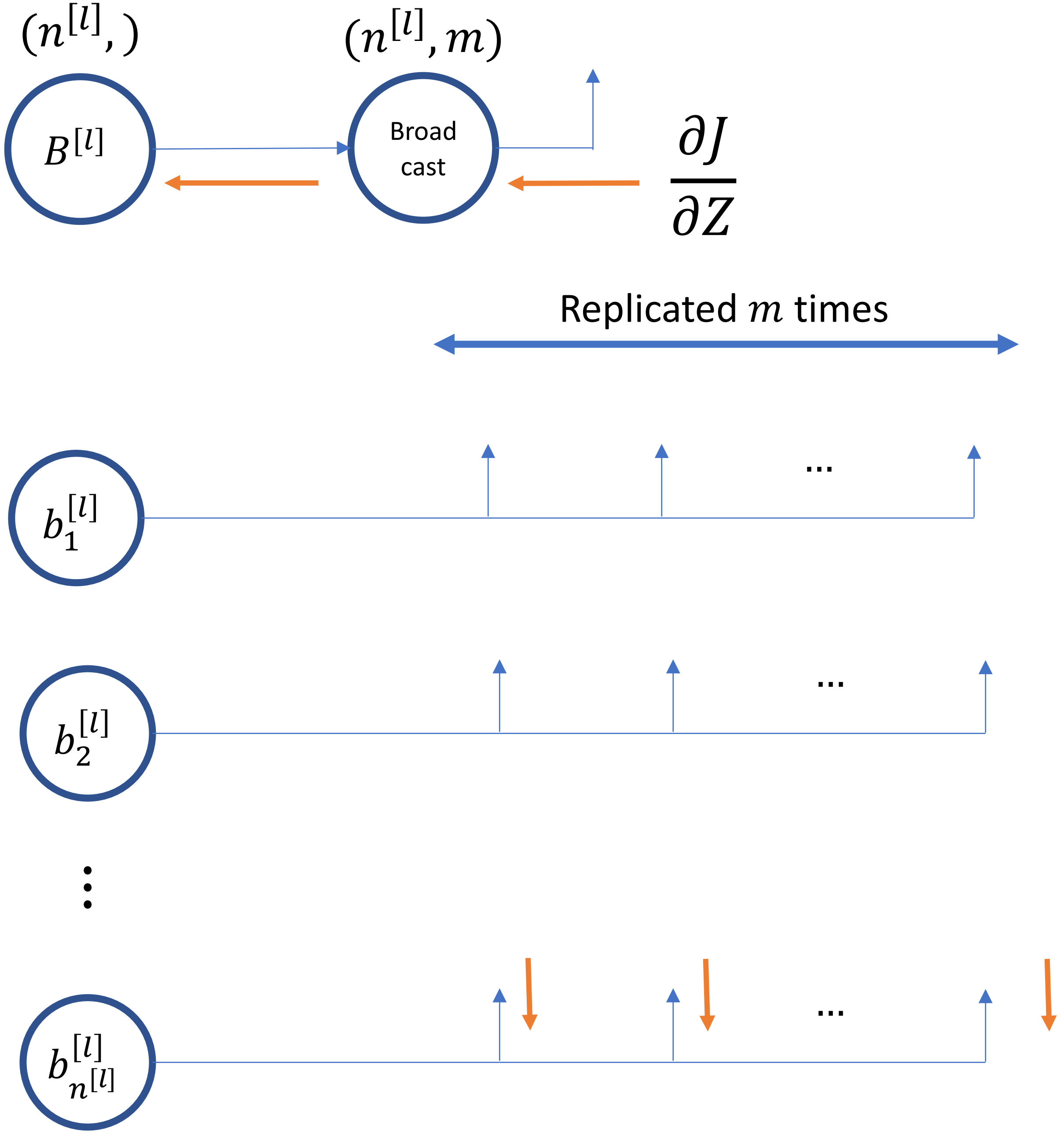
$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$



Broadcast

Each column is for one sample
Each row is for one unit of the layer

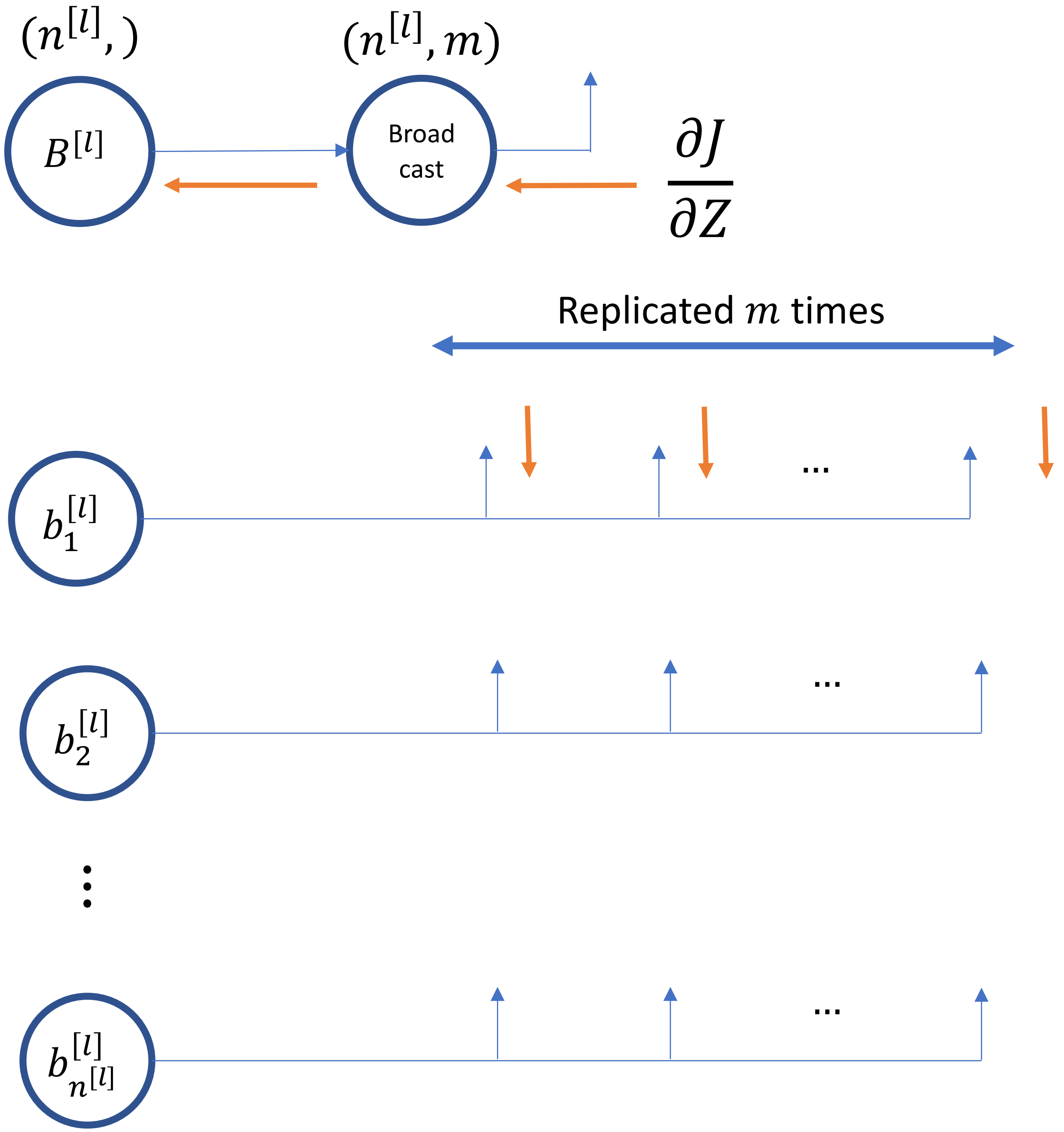
$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$



Broadcast

Each column is for one sample
Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$

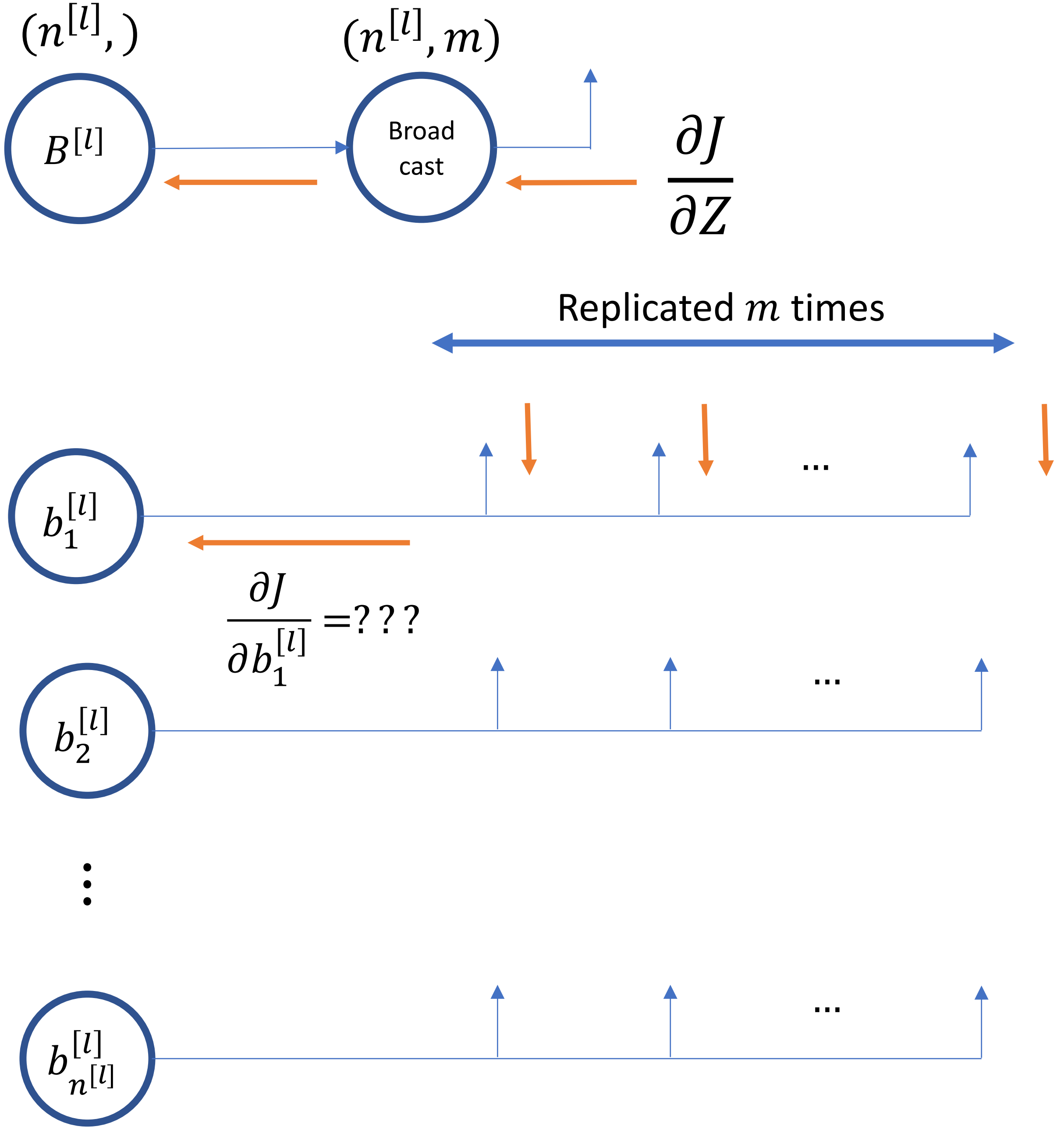


Broadcast

Each column is for one sample

Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$



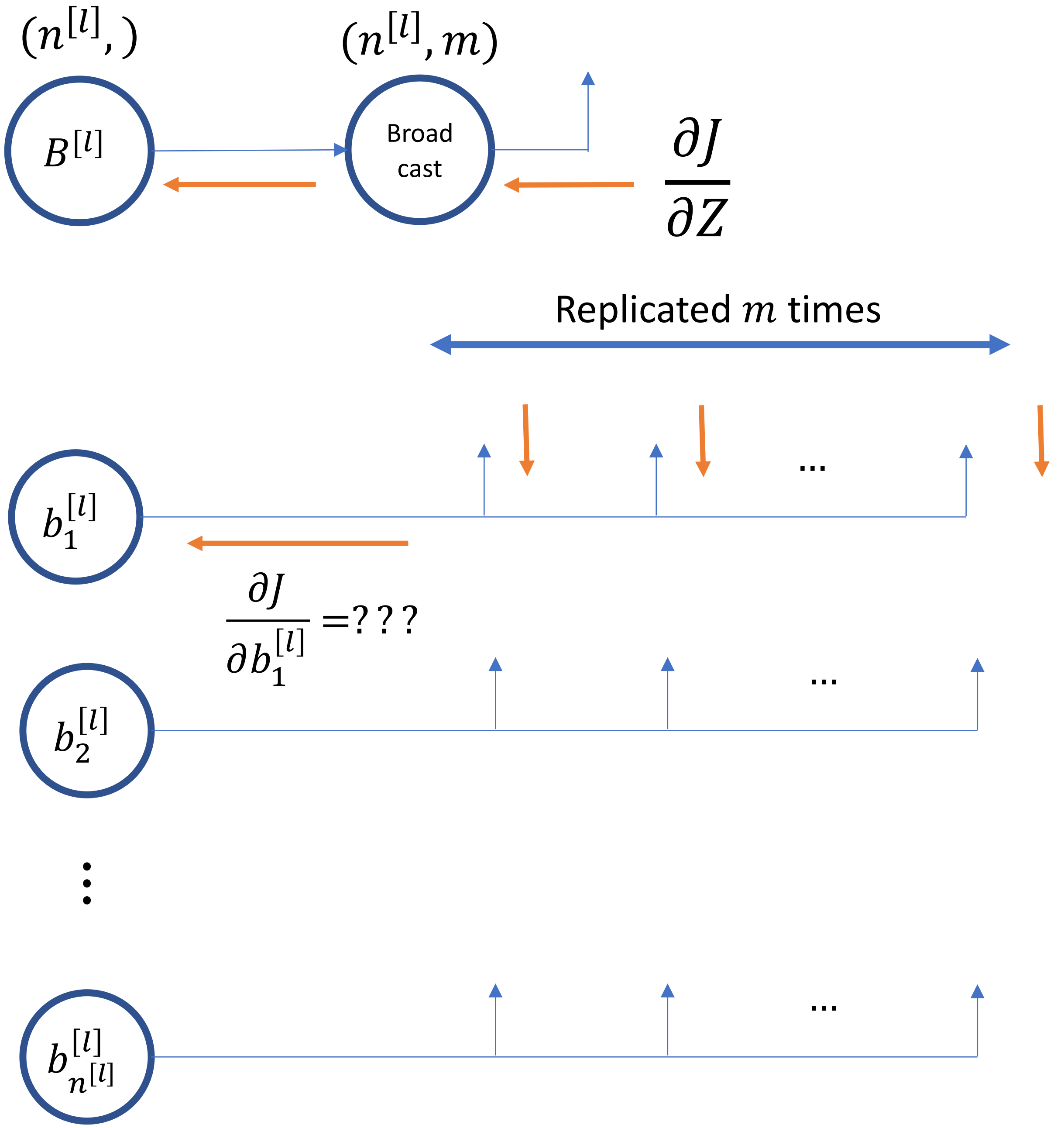
Broadcast

Each column is for one sample

Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$

Brad Quinton, Scott Chin



Broadcast

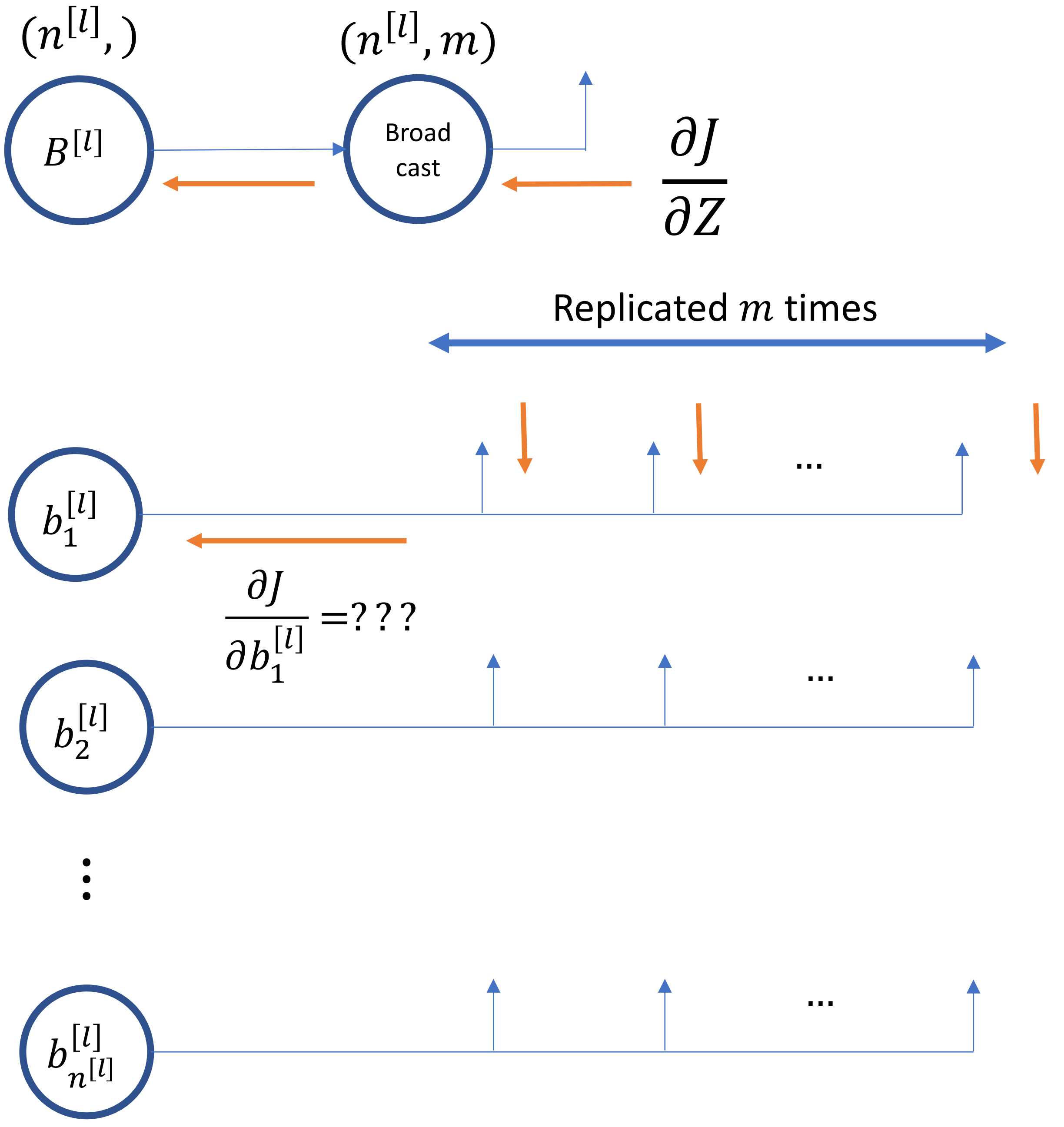
Each column is for one sample

Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$

$$\frac{\partial J}{\partial b_1^{[l]}} = \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} + \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} + \cdots + \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)}$$

Brad Quinton, Scott Chin



Broadcast

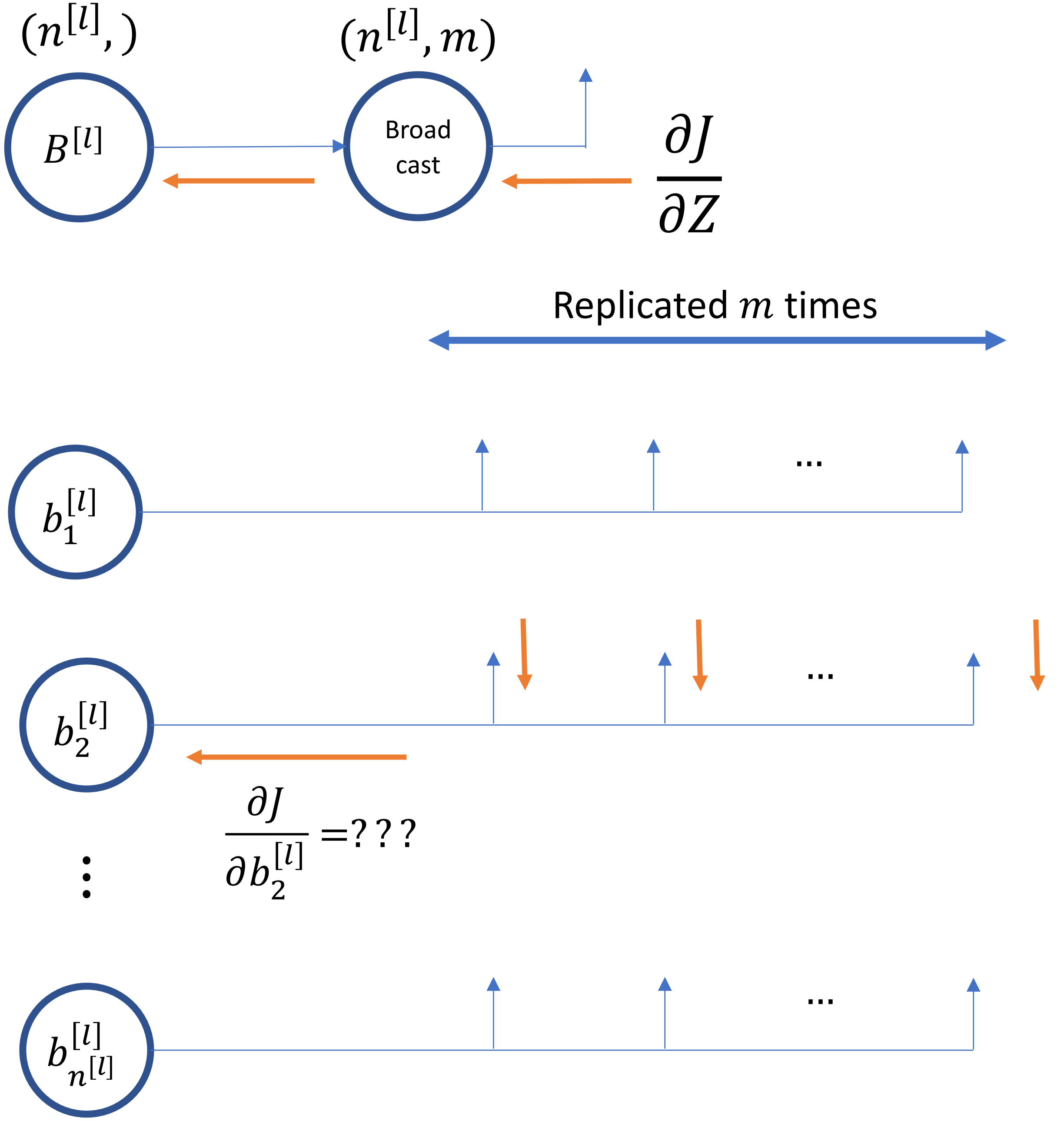
Each column is for one sample

Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$

$$\frac{\partial J}{\partial b_1^{[l]}} = \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} + \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} + \cdots + \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)}$$

$$= \sum_{j=1}^{m} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(j)}$$

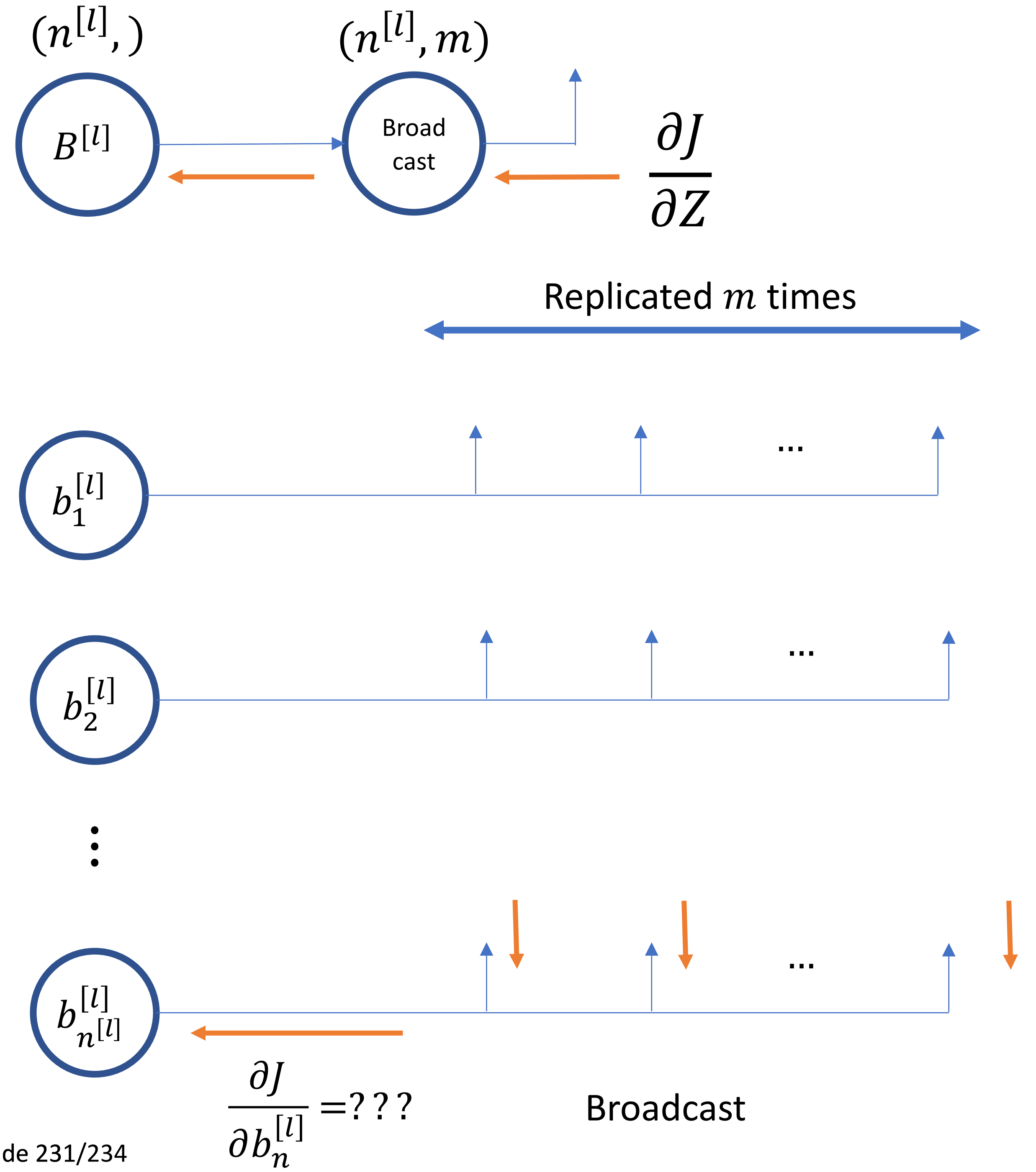


Broadcast

Each column is for one sample
Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$

$$\frac{\partial J}{\partial b_2^{[l]}} = \sum_{i=1}^{m} \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(i)}$$

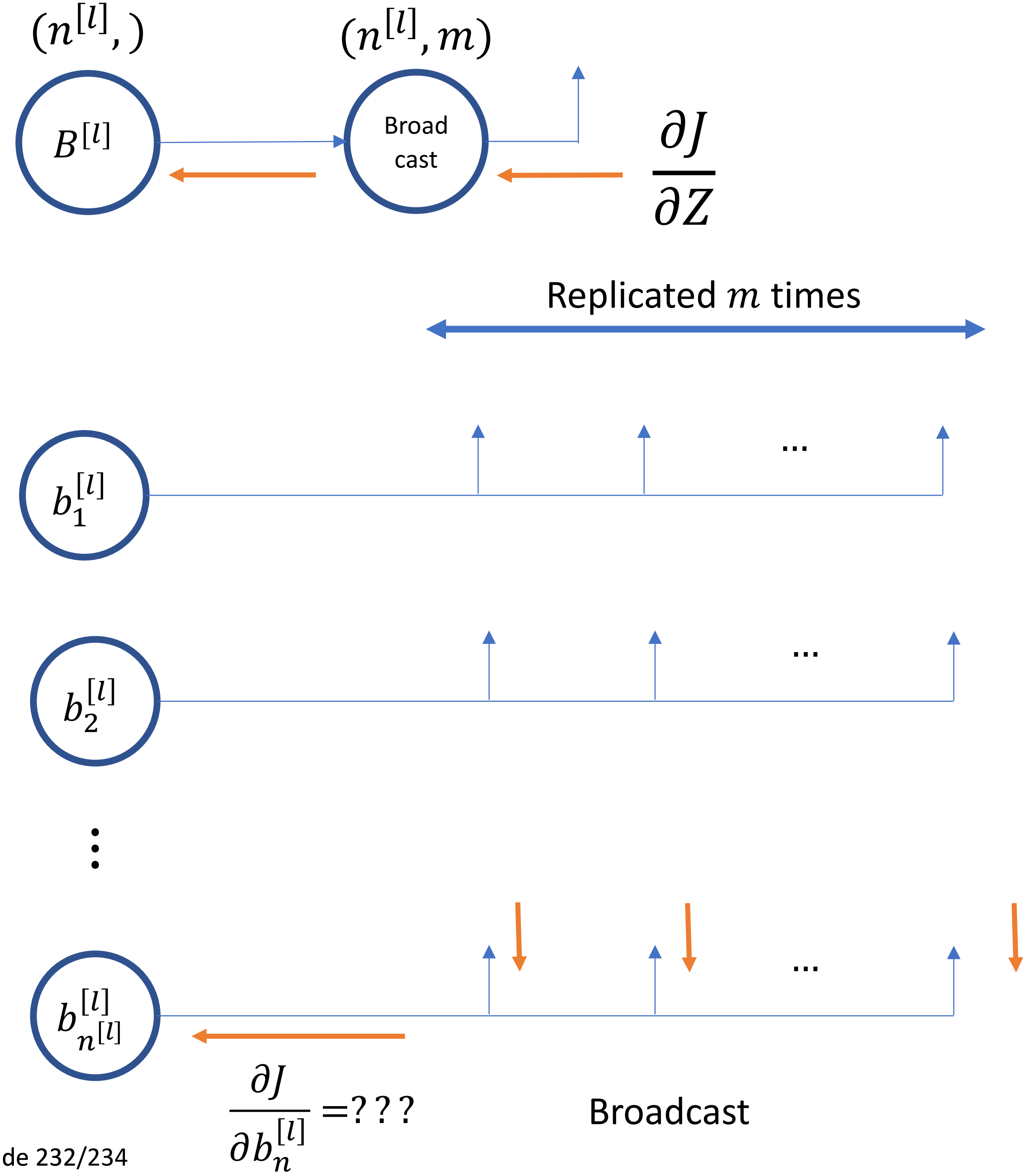


Each column is for one sample

Each row is for one unit of the layer

$$\frac{\partial J}{\partial Z} = \begin{bmatrix} \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_1^{[l]}}\right)^{(m)} \\ \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(2)} & \cdots & \left(\frac{\partial J}{\partial z_2^{[l]}}\right)^{(m)} \\ \vdots & \vdots & & \vdots \\ \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(1)} & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(2)} & & \left(\frac{\partial J}{\partial z_{n^{[l]}}^{[l]}}\right)^{(m)} \end{bmatrix}$$

$$\frac{\partial J}{\partial b_n^{[l]}} = \sum_{i=1}^{m} \left(\frac{\partial J}{\partial z_n^{[l]}}\right)^{(i)}$$

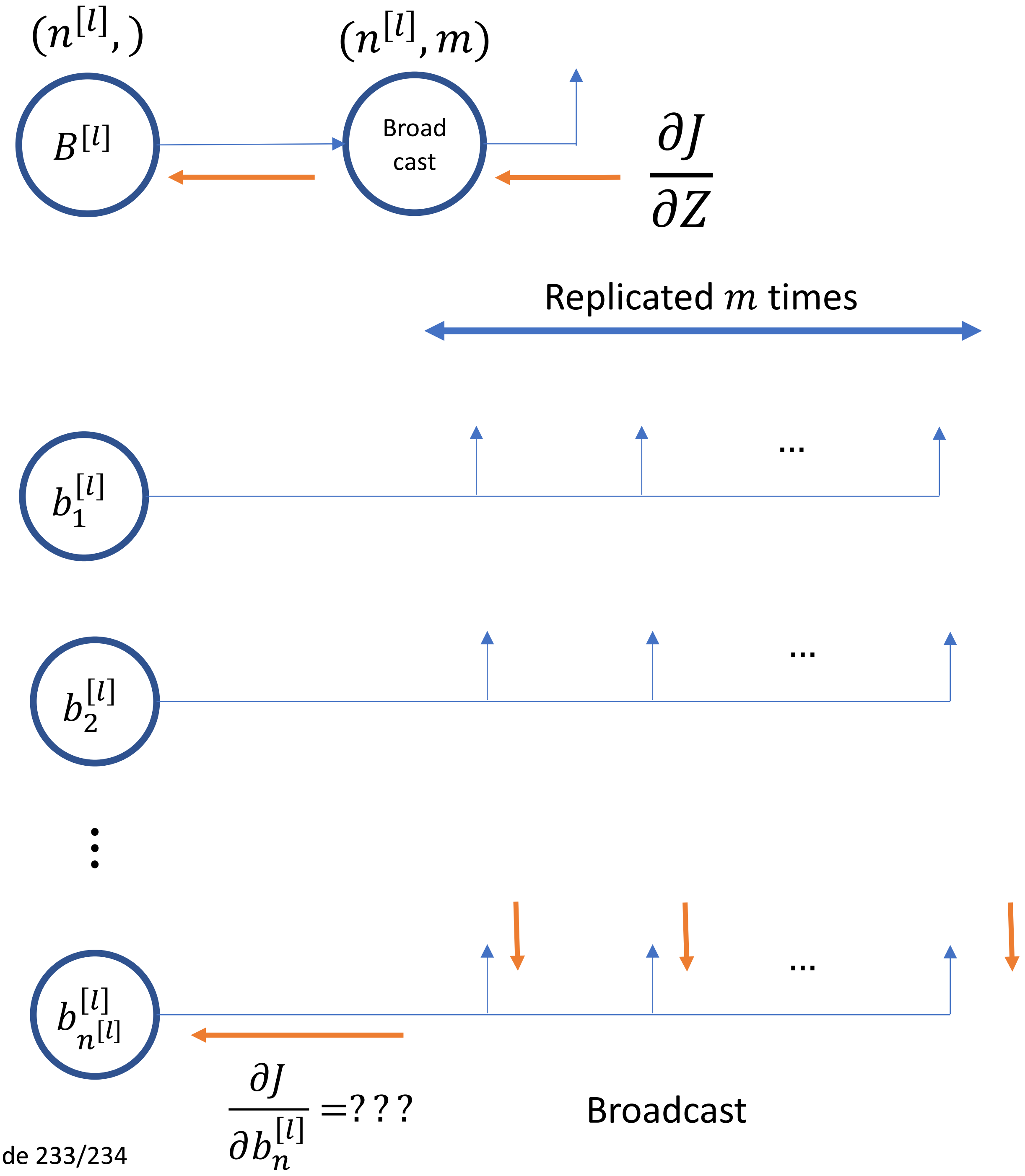


From Lecture 5

$$dB^{[2]} = \frac{1}{m} \sum_{rows} dZ^{[2]}$$

$$dB^{[1]} = \frac{1}{m} \sum_{rows} dZ^{[1]}$$

```
# From Assignment 2
dB2 = 1/m * np.sum(dZ2, axis=1)
```



From Lecture 5

$$dB^{[2]} = \frac{1}{m} \sum_{rows} dZ^{[2]}$$

$$dB^{[1]} = \frac{1}{m} \sum_{rows} dZ^{[1]}$$

```
# From Assignment 2
dB2 = 1/m * np.sum(dZ2, axis=1)
```

Side Note:

- This highlights why average loss is more practical than total loss.
- It provides the 1/m term. Without it, the gradients on our parameters would increase as m increases, and cause numerical overflow issues.

Learning Objectives

- Extend our understanding of backpropagation to vectorized operations