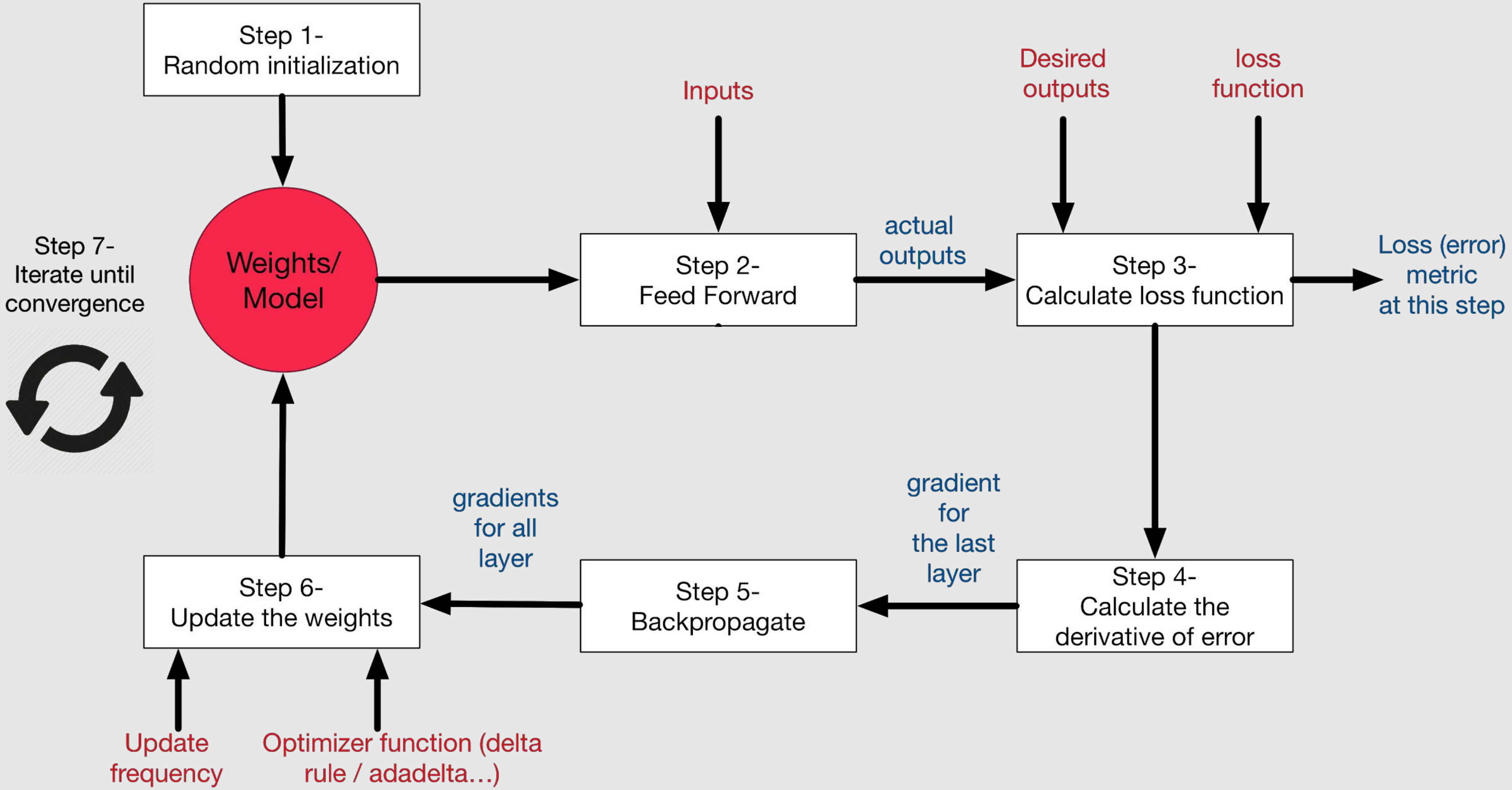
Recall from last time ...

Training a Neural Network



Backpropagation

A Simple Example

Backpropagation: A Simple Example

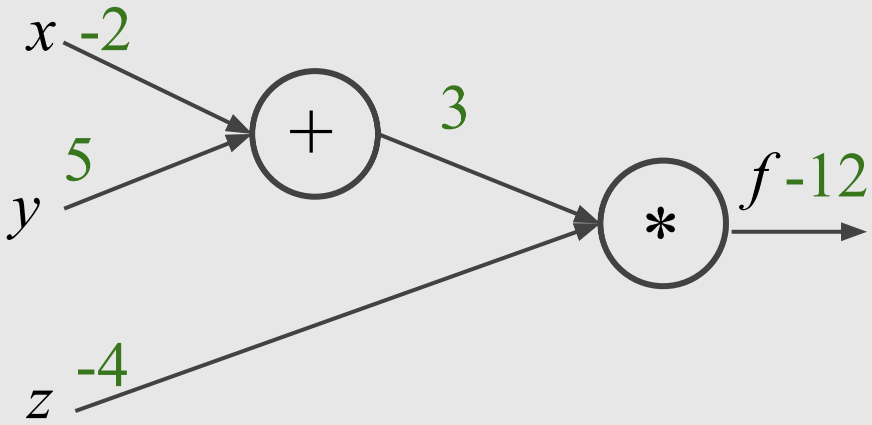
$f(x, y, z) = (x + y)z$

e.g., $x = -2$, $y = 5$, $z = -4$

Backpropagation: A Simple Example

$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$



Backpropagation: A Simple Example

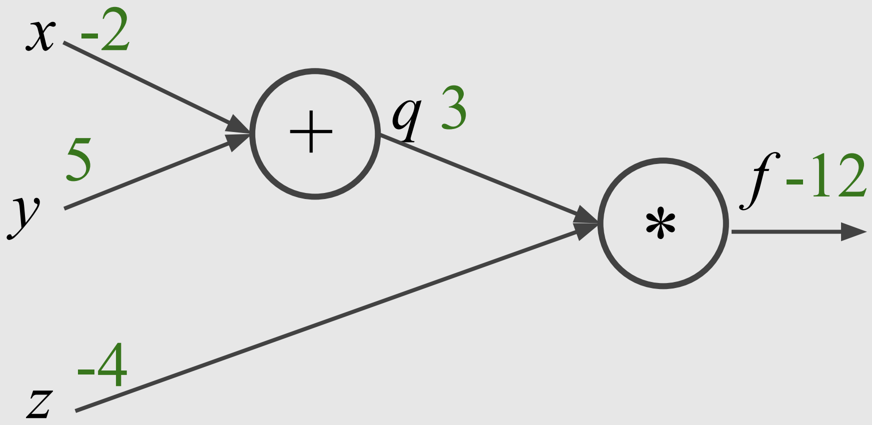
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \quad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: A Simple Example

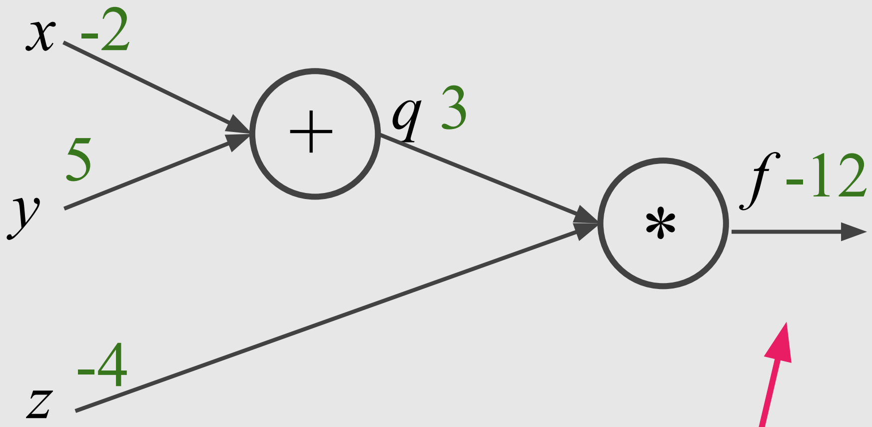
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \quad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial f}$$

Backpropagation: A Simple Example

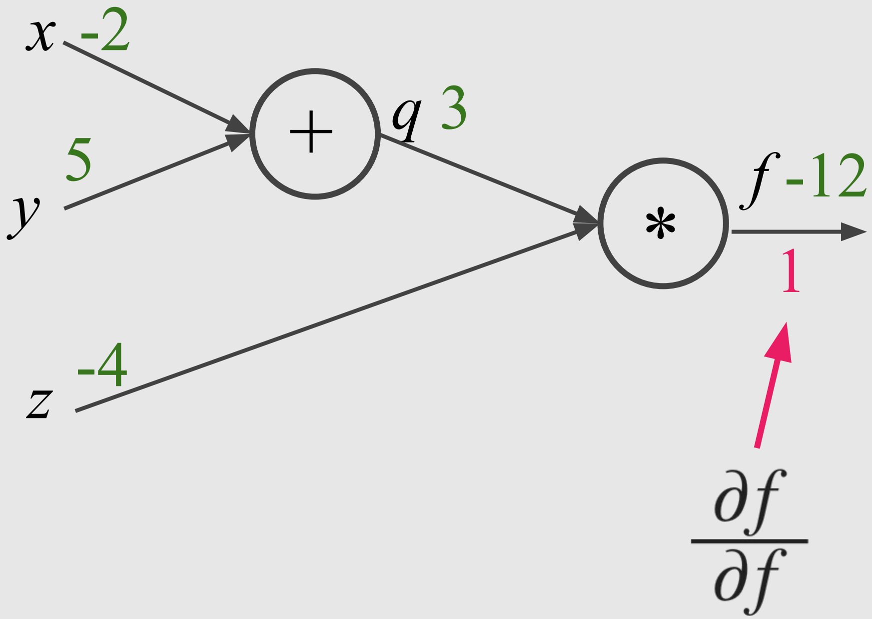
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \quad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: A Simple Example

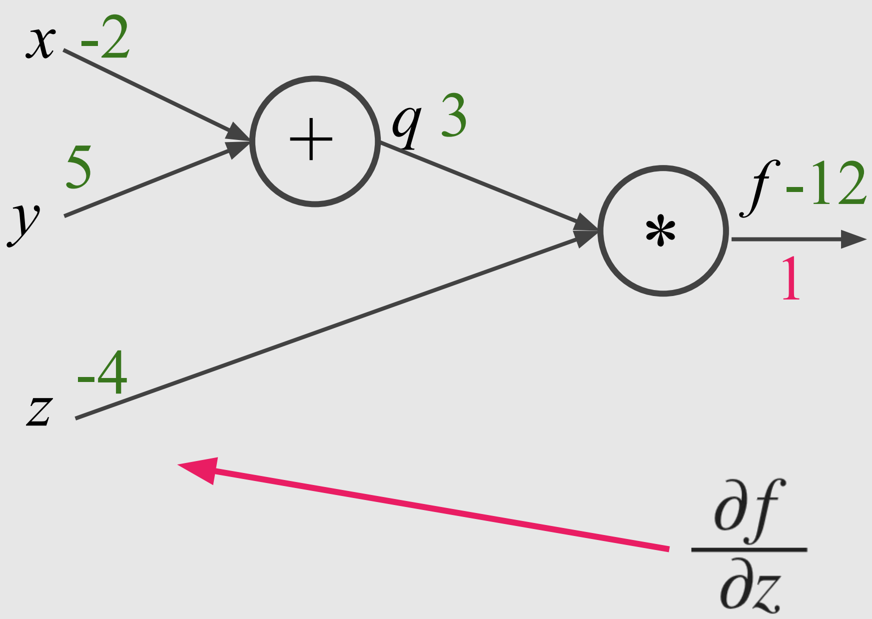
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \quad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: A Simple Example

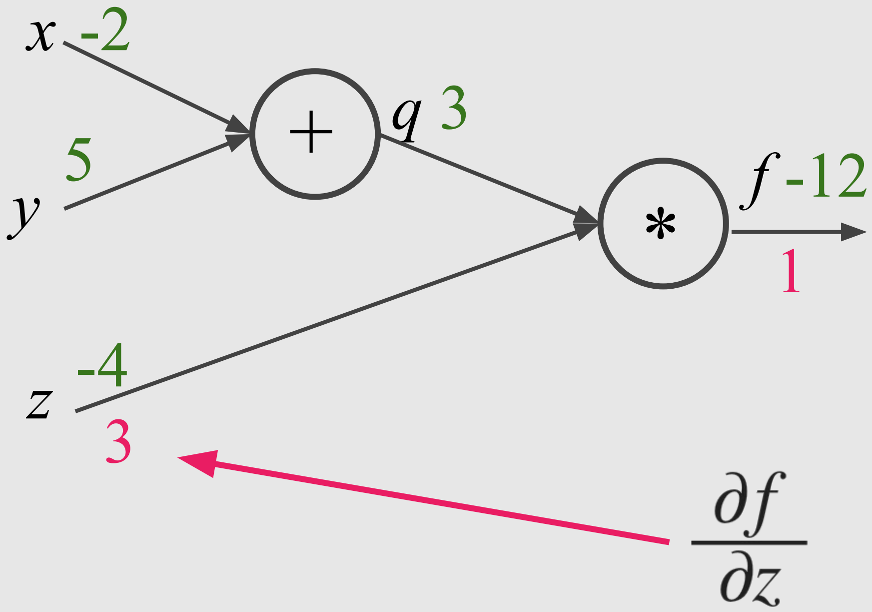
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \quad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: A Simple Example

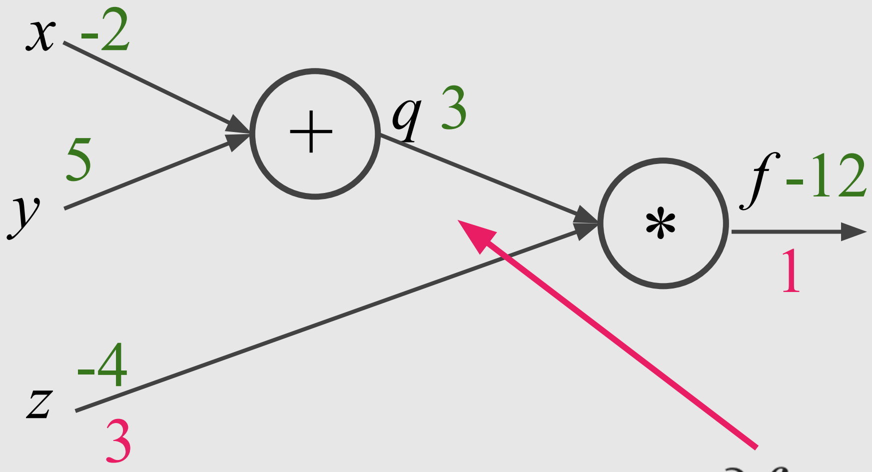
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \quad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial q}$$

Backpropagation: A Simple Example

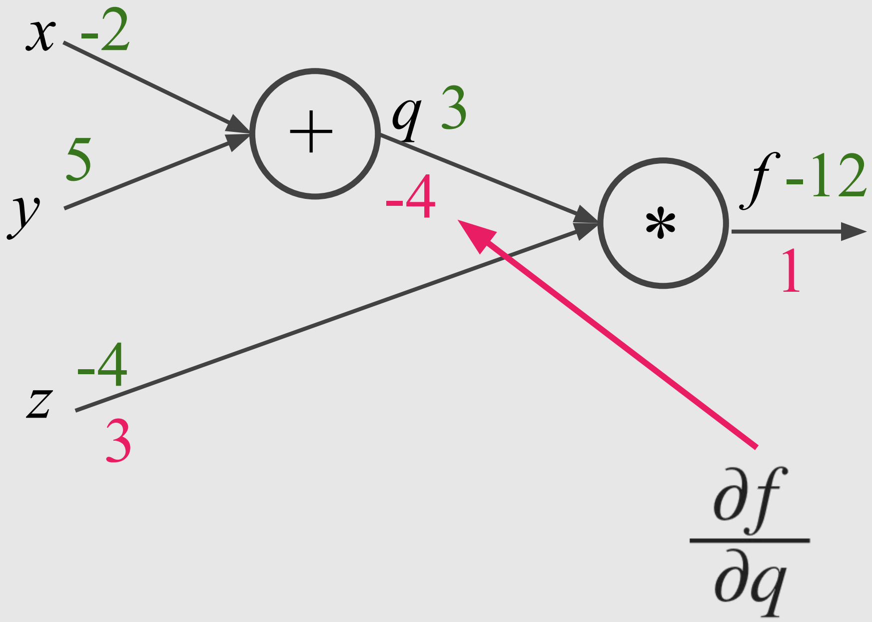
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \qquad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \qquad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: A Simple Example

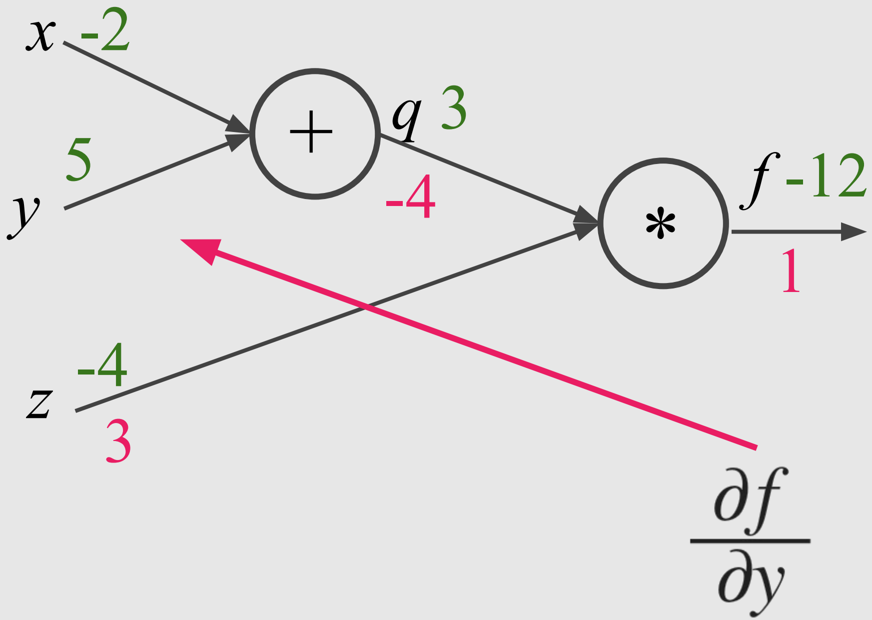
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \qquad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \qquad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: A Simple Example

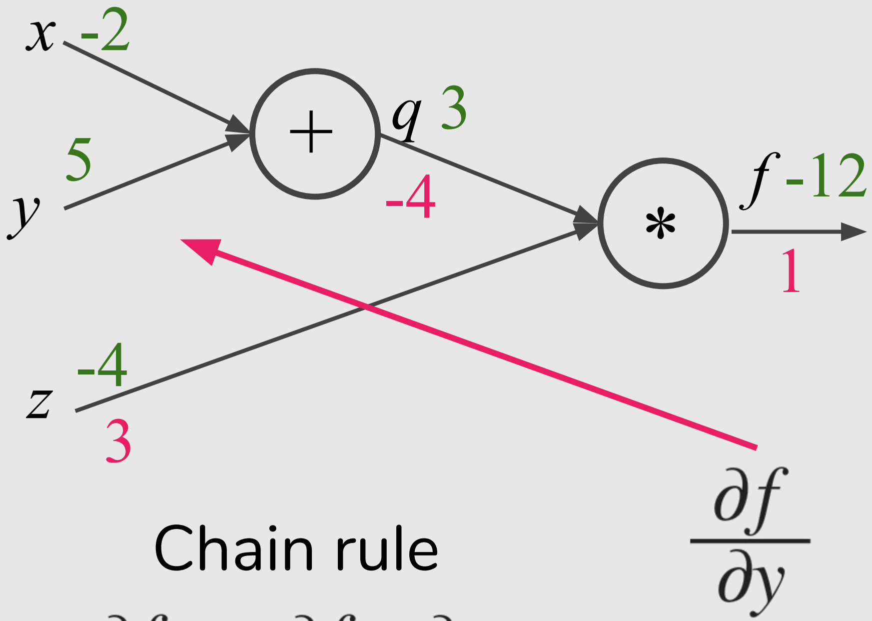
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \qquad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \qquad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Chain rule

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

Backpropagation: A Simple Example

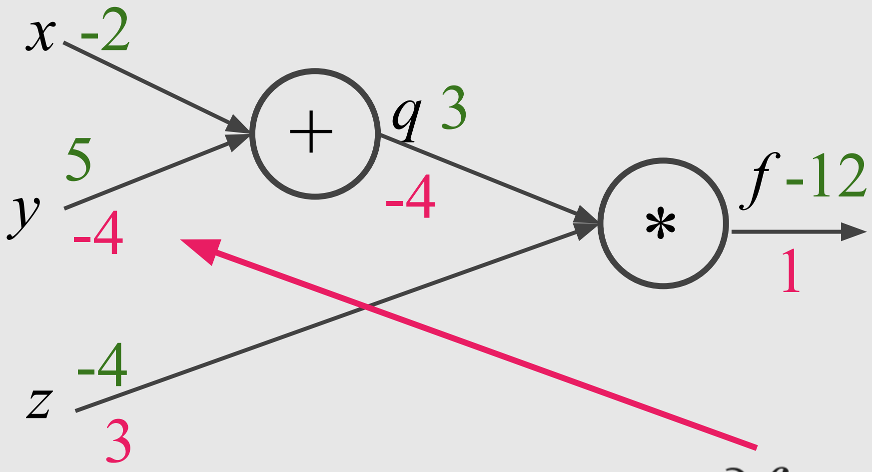
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$\boxed{q = x + y \qquad \frac{\partial q}{\partial x} = 1 \qquad \frac{\partial q}{\partial y} = 1}$$

$$\boxed{f = qz \qquad \frac{\partial f}{\partial q} = z \qquad \frac{\partial f}{\partial z} = q}$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



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

Chain rule

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

Backpropagation: A Simple Example

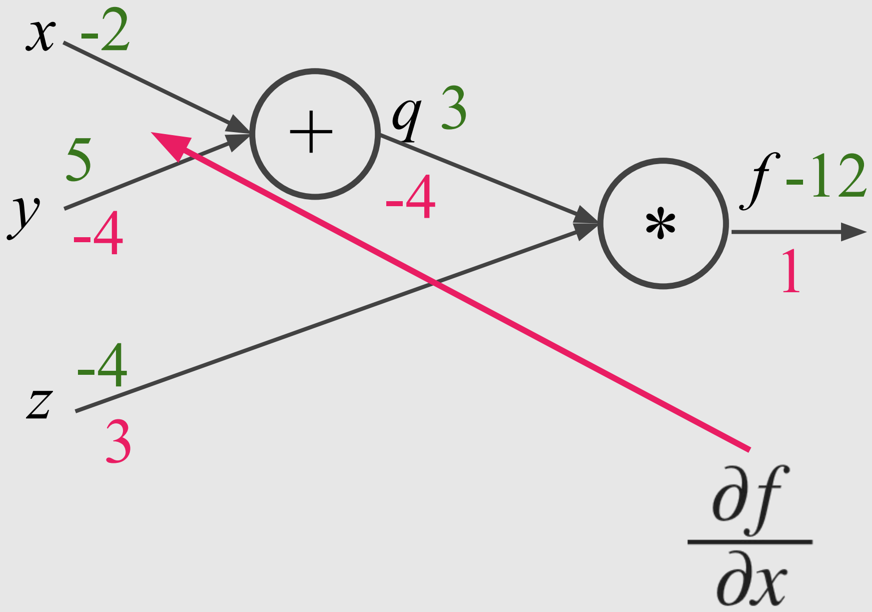
$$f(x, y, z) = (x + y)z$$

e.g., $x = -2$, $y = 5$, $z = -4$

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \qquad \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z \qquad \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation: A Simple Example

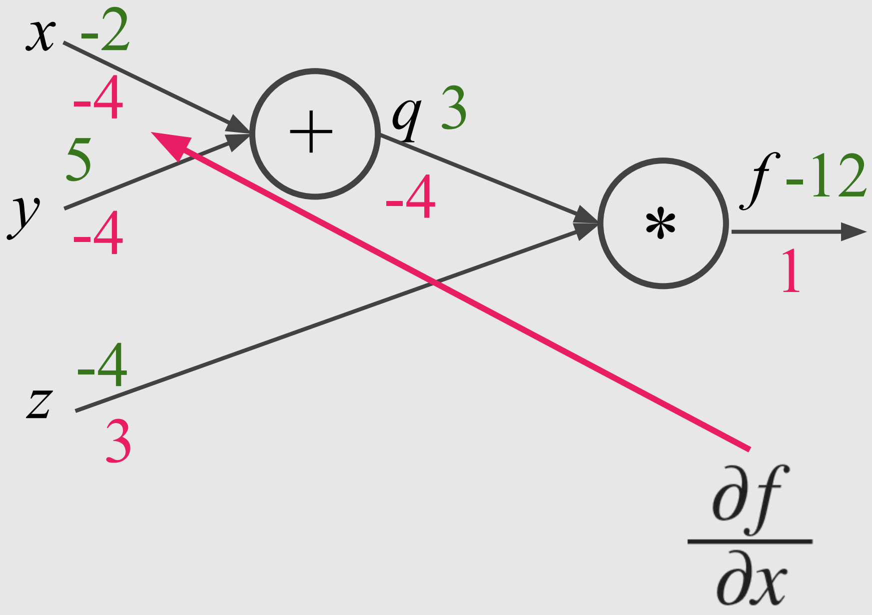
$$f(x, y, z) = (x + y)z$$

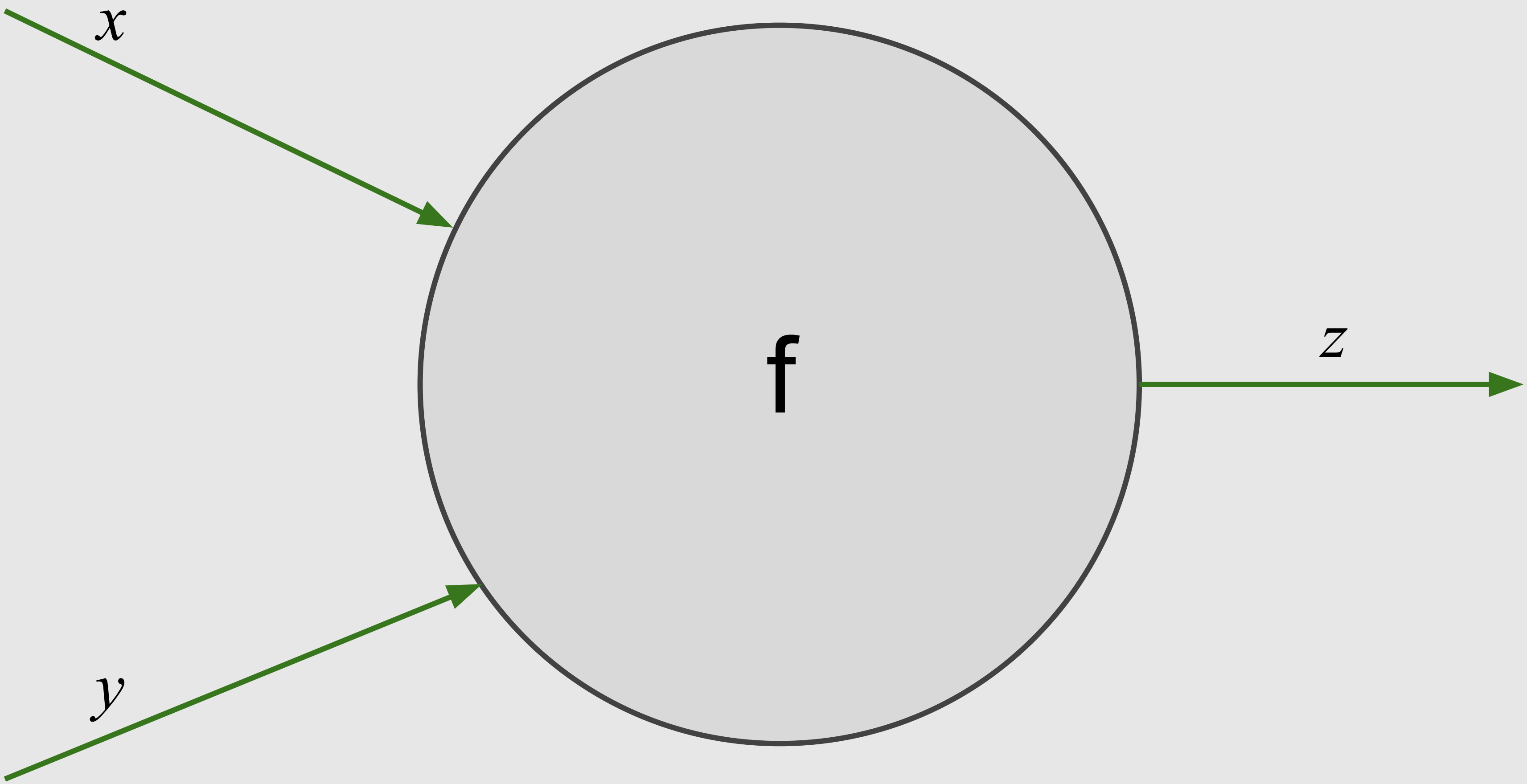
e.g., $x = -2$, $y = 5$, $z = -4$

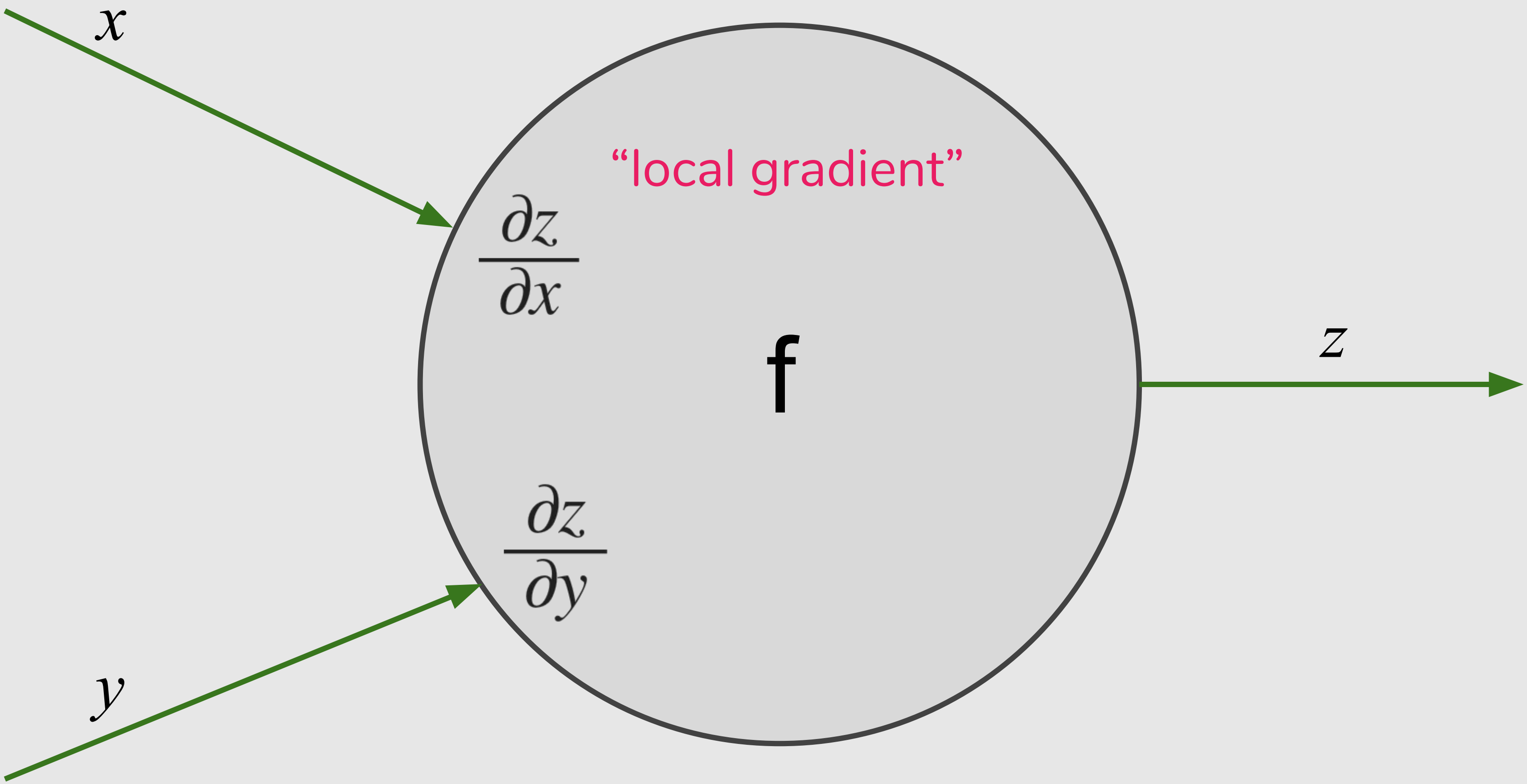
$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

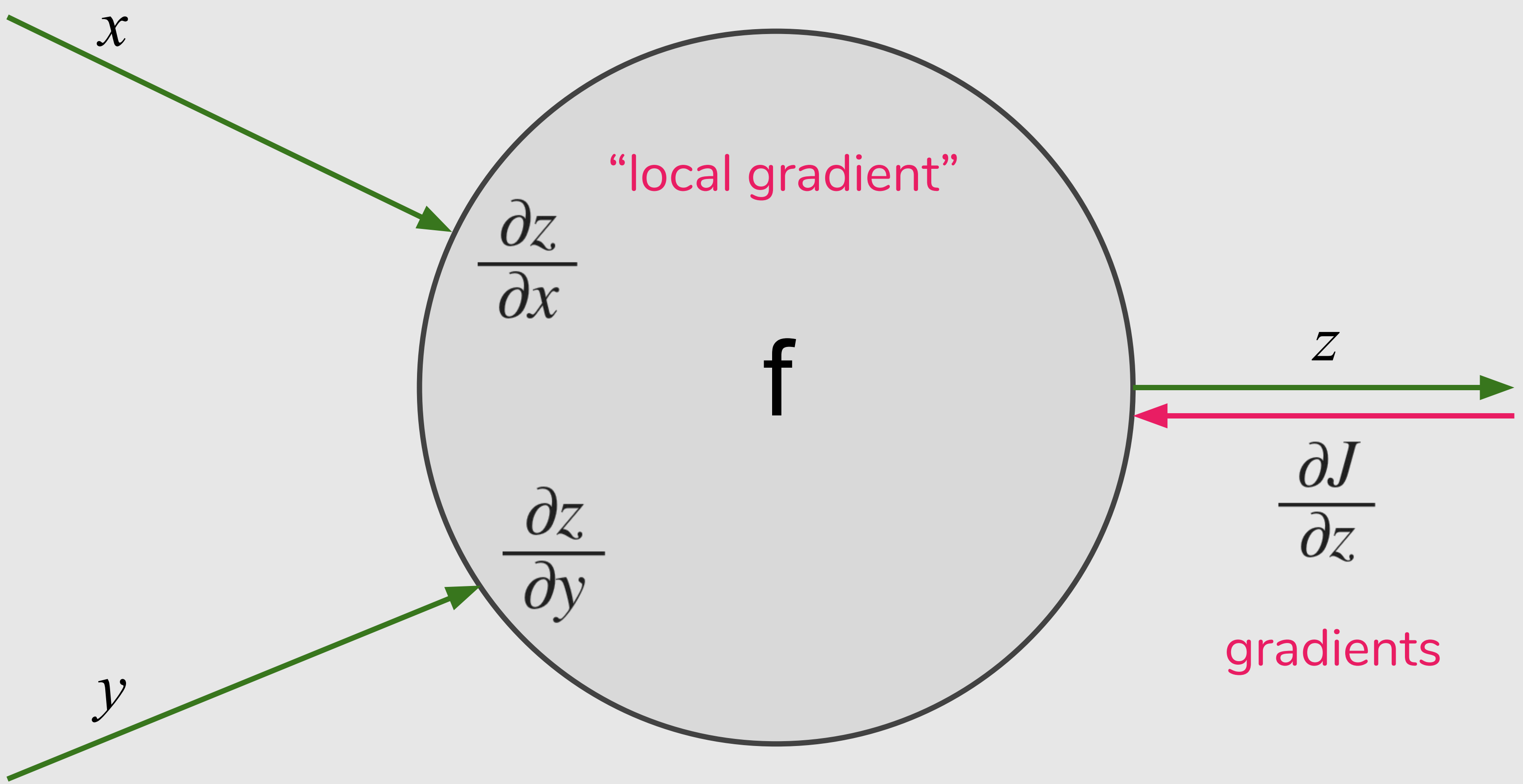
$$f = qz \qquad \frac{\partial f}{\partial q} = z \quad \frac{\partial f}{\partial z} = q$$

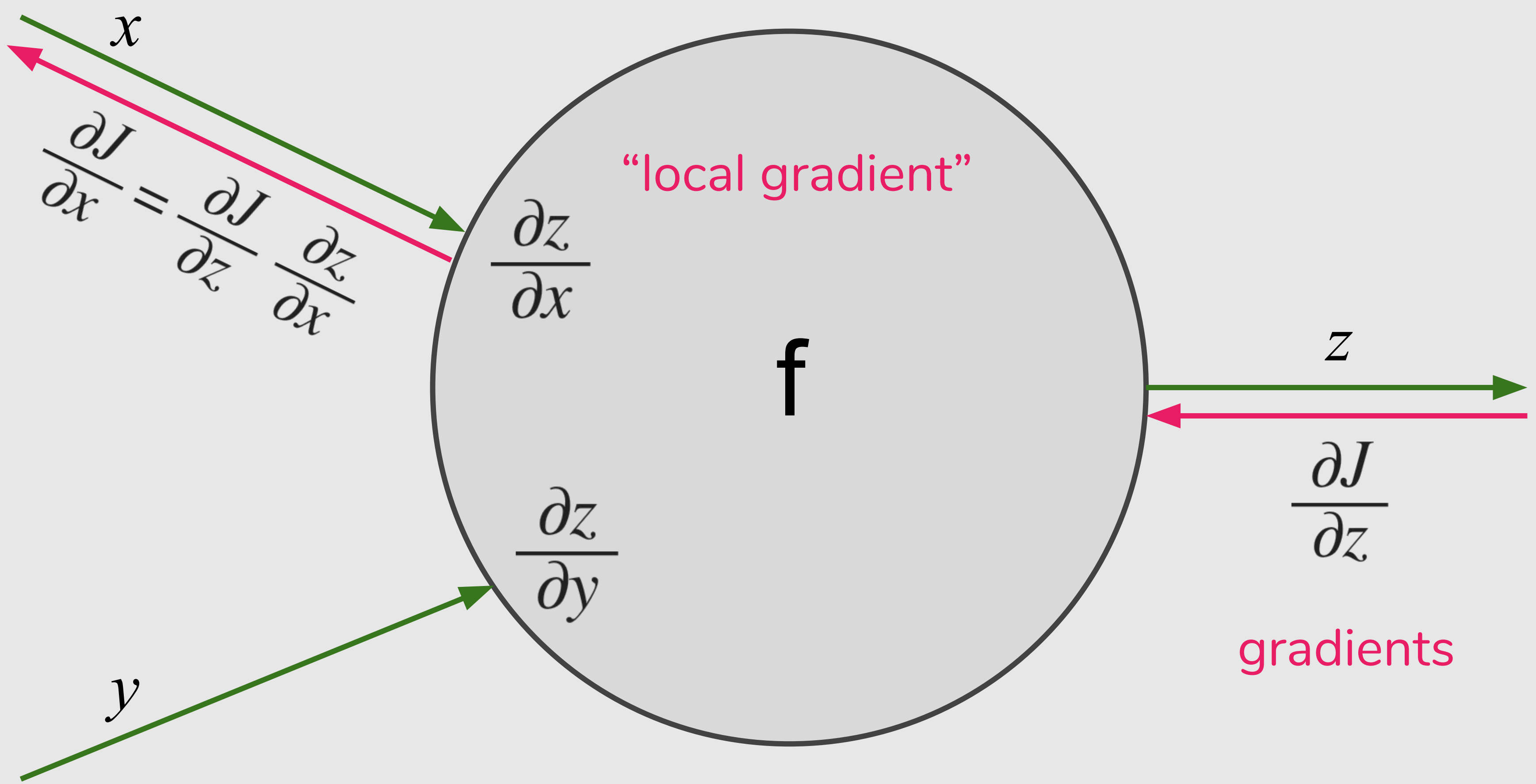
Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



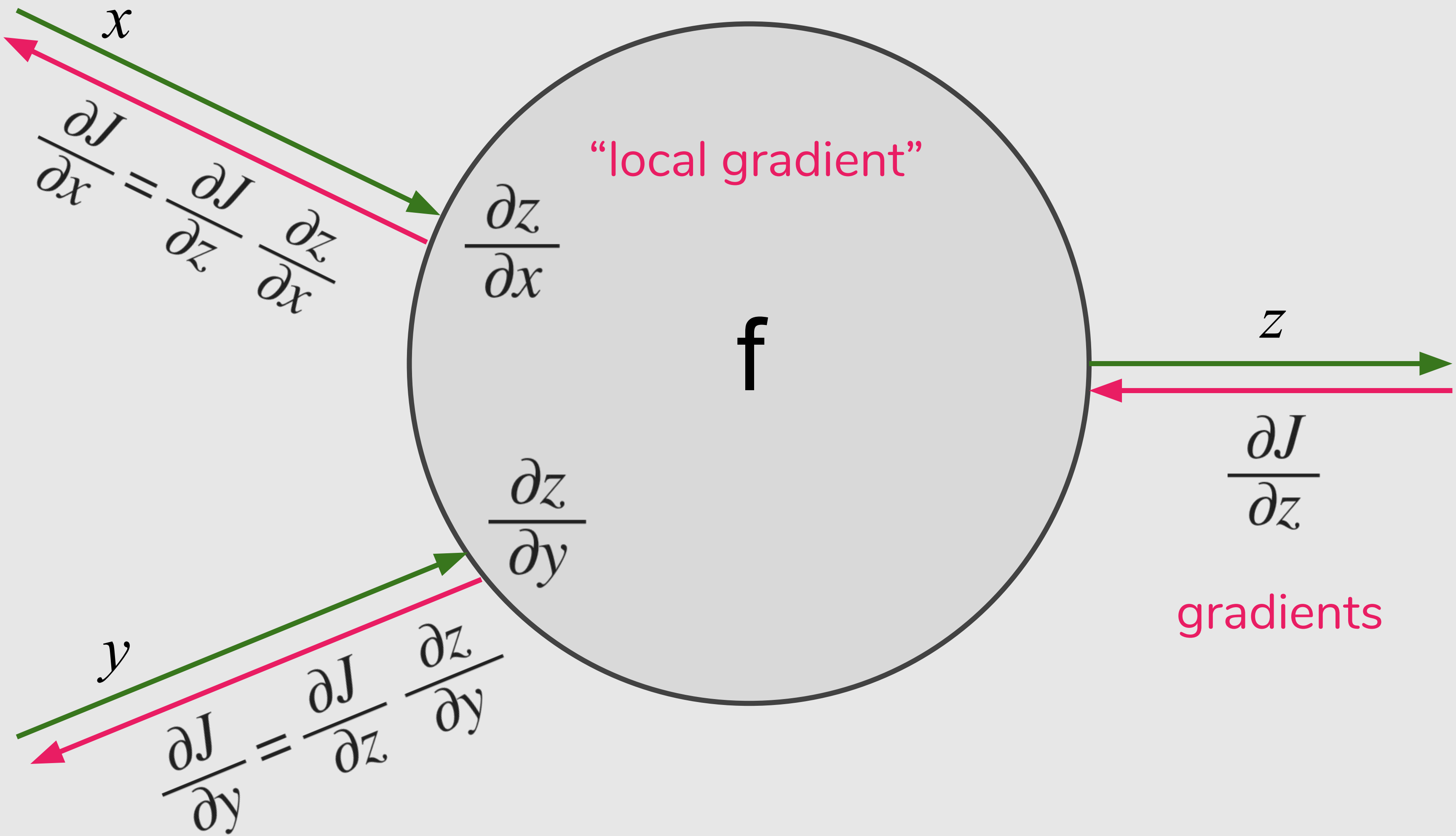

x
f
z
y


x
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial z}{\partial y}$
y

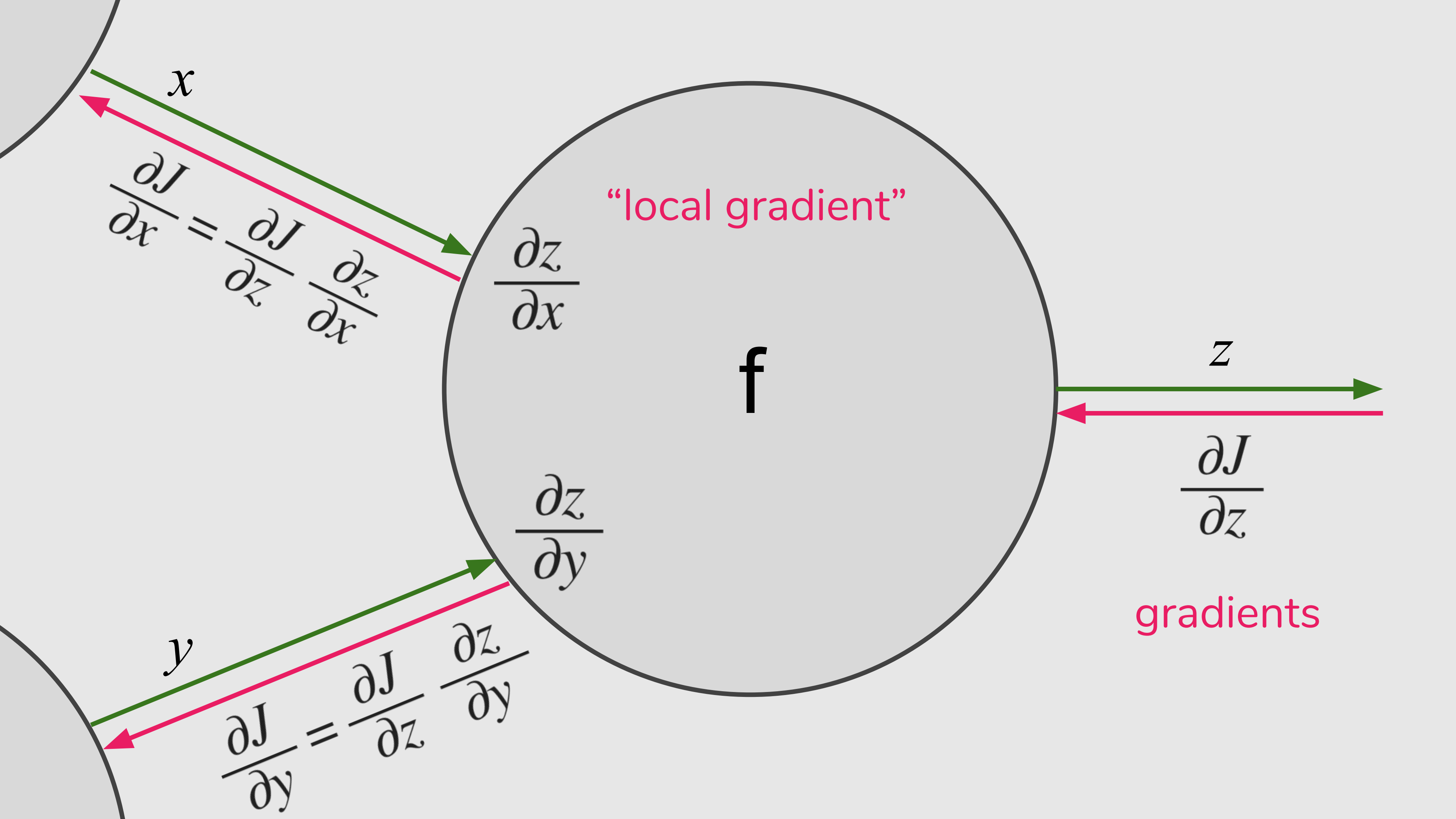

x
y
"local gradient"
$\frac{\partial z}{\partial x}$
$\frac{\partial z}{\partial y}$
f
z
$\frac{\partial J}{\partial z}$
gradients



x
$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial z} \frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial z}{\partial y}$
$\frac{\partial J}{\partial z}$
gradients
y

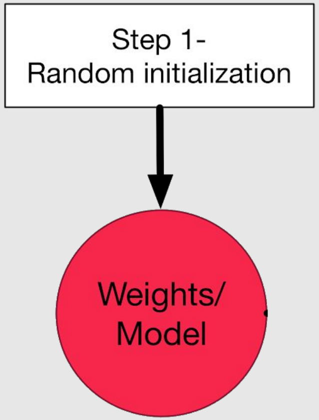


x
$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial z} \frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial J}{\partial z}$
$\frac{\partial z}{\partial y}$
gradients
y
$\frac{\partial J}{\partial y} = \frac{\partial J}{\partial z} \frac{\partial z}{\partial y}$



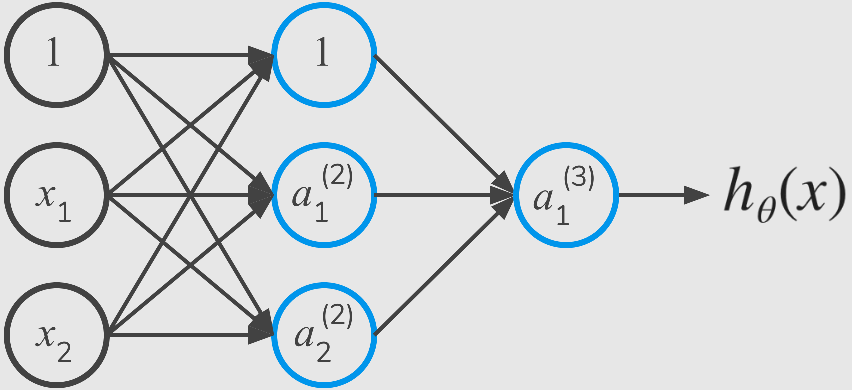
x
$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial z}\frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial J}{\partial z}$
$\frac{\partial z}{\partial y}$
gradients
y
$\frac{\partial J}{\partial y} = \frac{\partial J}{\partial z}\frac{\partial z}{\partial y}$

Training a Neural Network



Zero Initialization

Symmetric Weights



$$a_1^{(2)} = a_2^{(2)}$$

After each update, parameters corresponding to inputs going into each of two hidden units are identical.

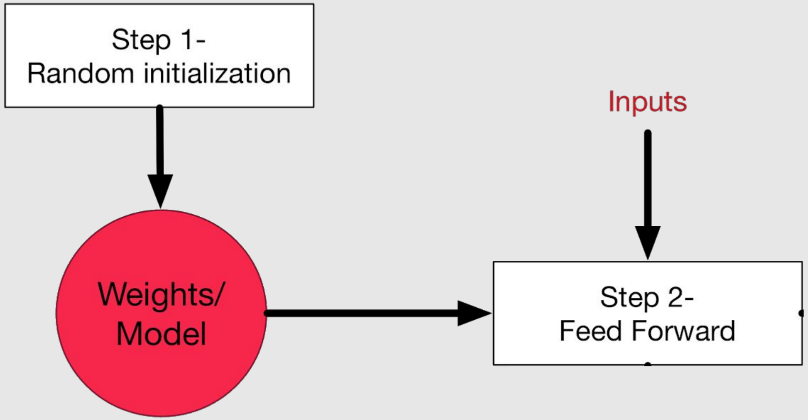
Symmetric Breaking

- We must initialize Θ to a **random value** in $[-\varepsilon, \varepsilon]$ (i.e. $[-\varepsilon \le \Theta \le \varepsilon]$)

- If the dimensions of `Theta1` is 3x4, `Theta2` is 3x4 and `Theta3` is 1x4.

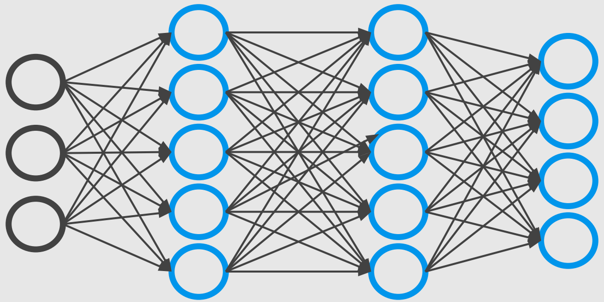
```
Theta1 = random(3,4) * (2 * EPSILON) - EPSILON;
Theta2 = random(3,4) * (2 * EPSILON) - EPSILON;
Theta3 = random(1,4) * (2 * EPSILON) - EPSILON;
```

Training a Neural Network



Forward Propagation

Given one training example (x, y):



Forward Propagation

Given one training example (x, y):

$$a^{(1)} = x$$

$$z^{(2)} = \Theta^{(1)} a^{(1)}$$

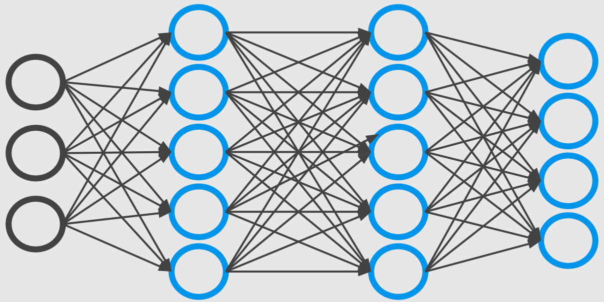
$$a^{(2)} = g(z^{(2)}) \quad (\text{add } a_0^{(2)})$$

$$z^{(3)} = \Theta^{(2)} a^{(2)}$$

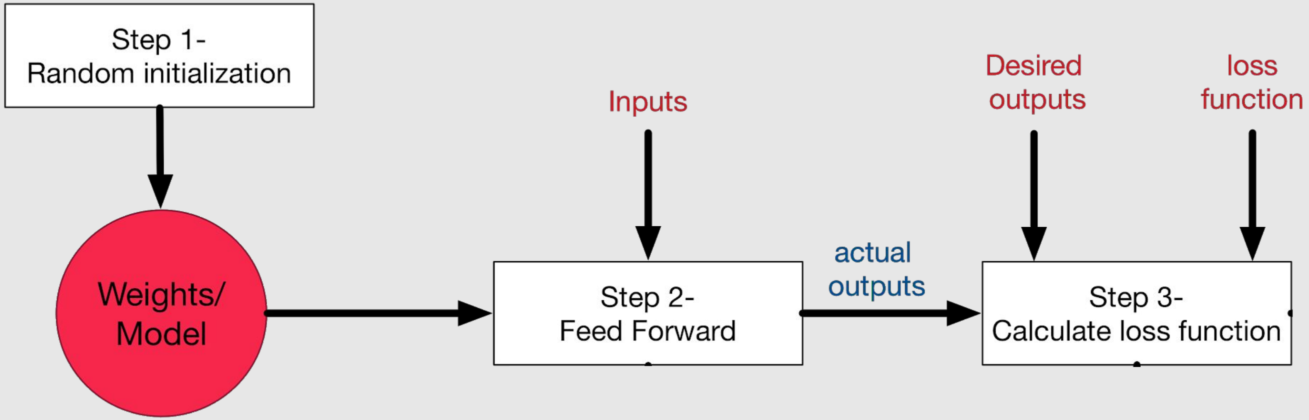
$$a^{(3)} = g(z^{(3)}) \quad (\text{add } a_0^{(3)})$$

$$z^{(4)} = \Theta^{(3)} a^{(3)}$$

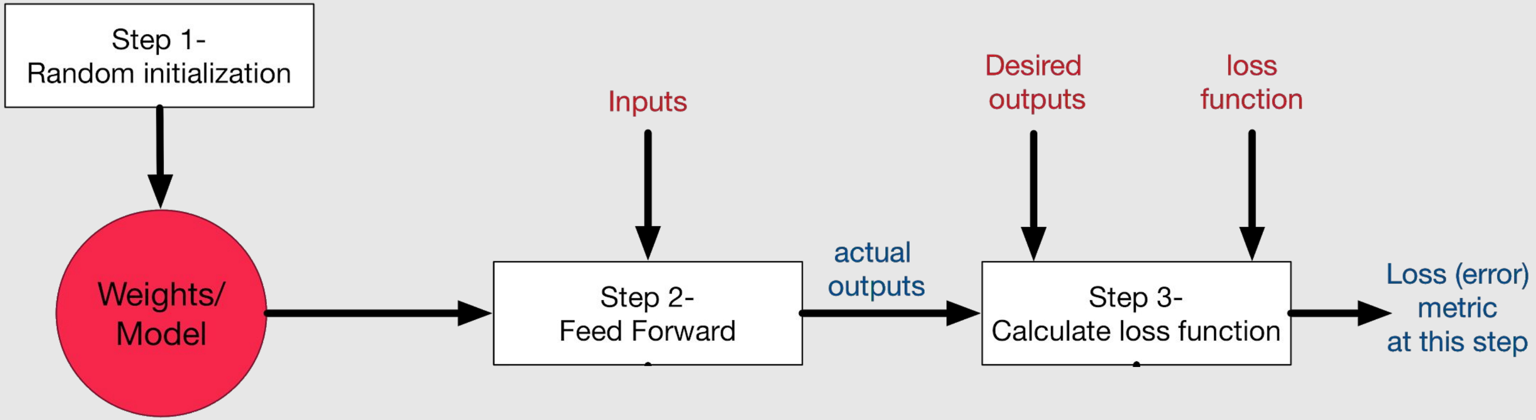
$$a^{(4)} = h_\Theta(x) = g(z^{(4)})$$



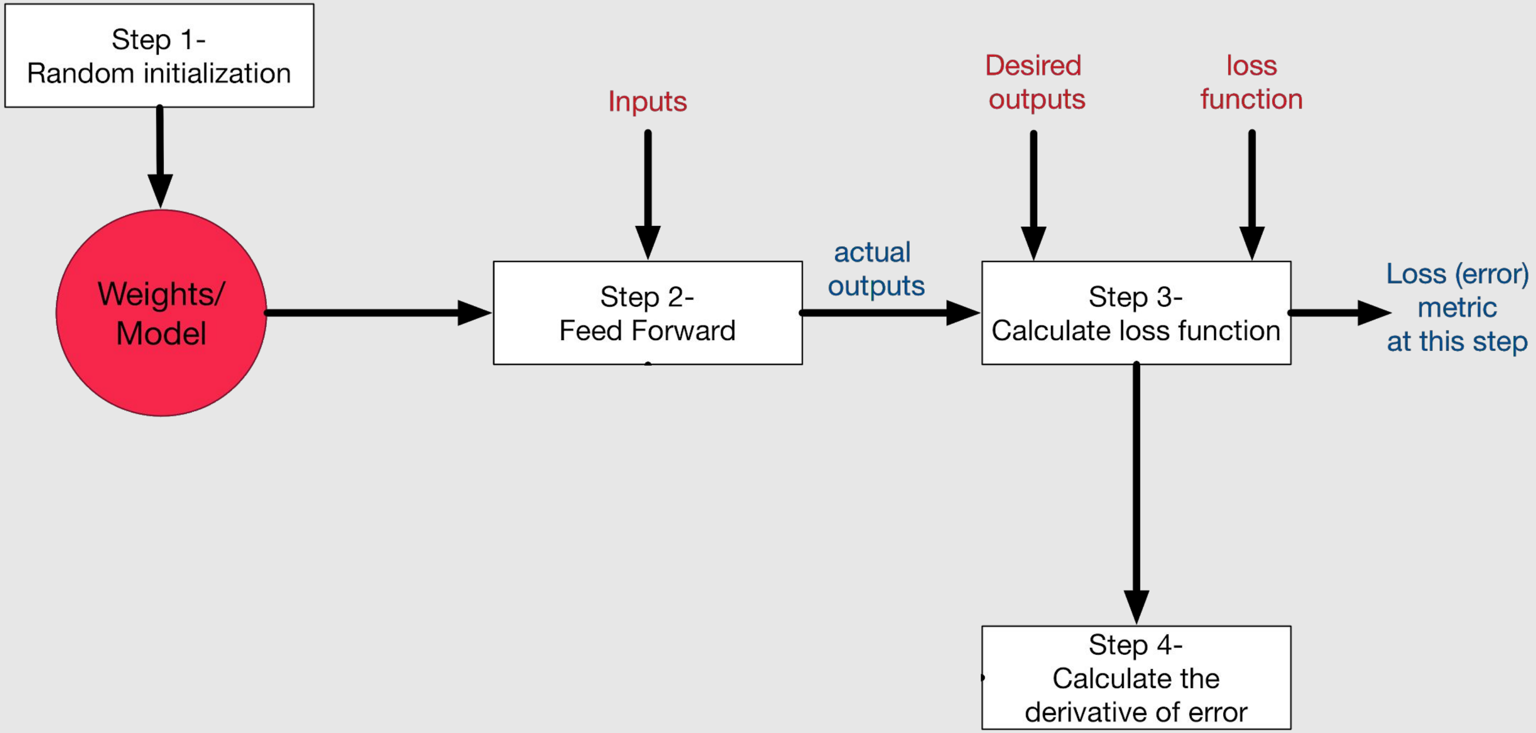
Training a Neural Network



Training a Neural Network

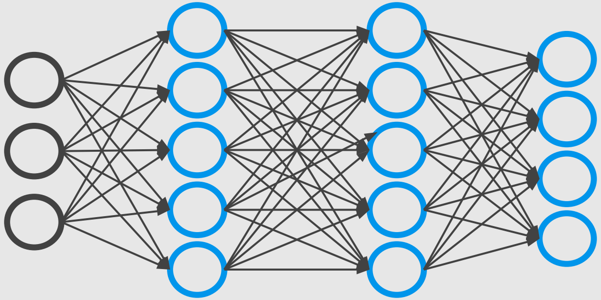


Training a Neural Network



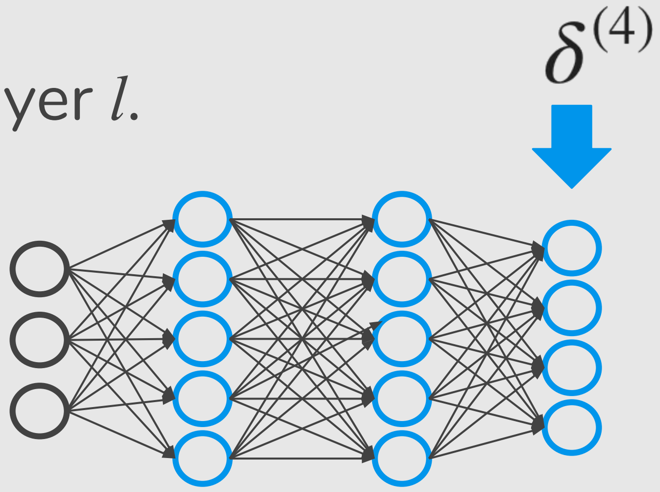
Gradient Computation: Backpropagation Algorithm

Intuition: $\delta_j^{(l)} =$ "error" of node j in layer l.



Gradient Computation: Backpropagation Algorithm

Intuition: $\delta_j^{(l)}$ = "error" of node j in layer l.

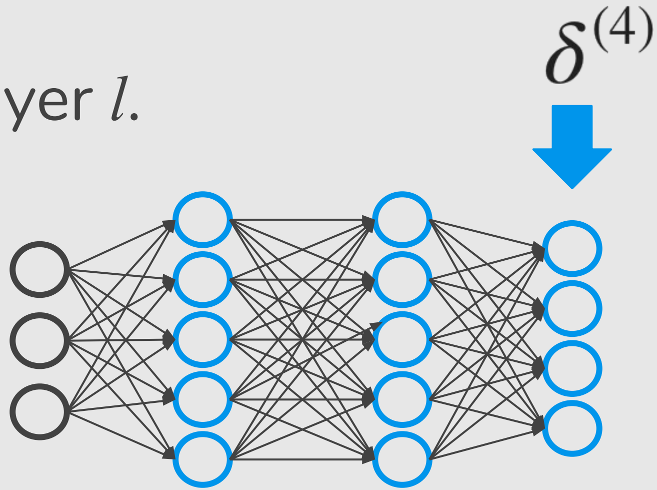


For each output unit (layer $L = 4$)

$$\delta_j^{(4)} = a_j^{(4)} - y_j$$

Gradient Computation: Backpropagation Algorithm

Intuition: $\delta_j^{(l)}$ = "error" of node j in layer l.

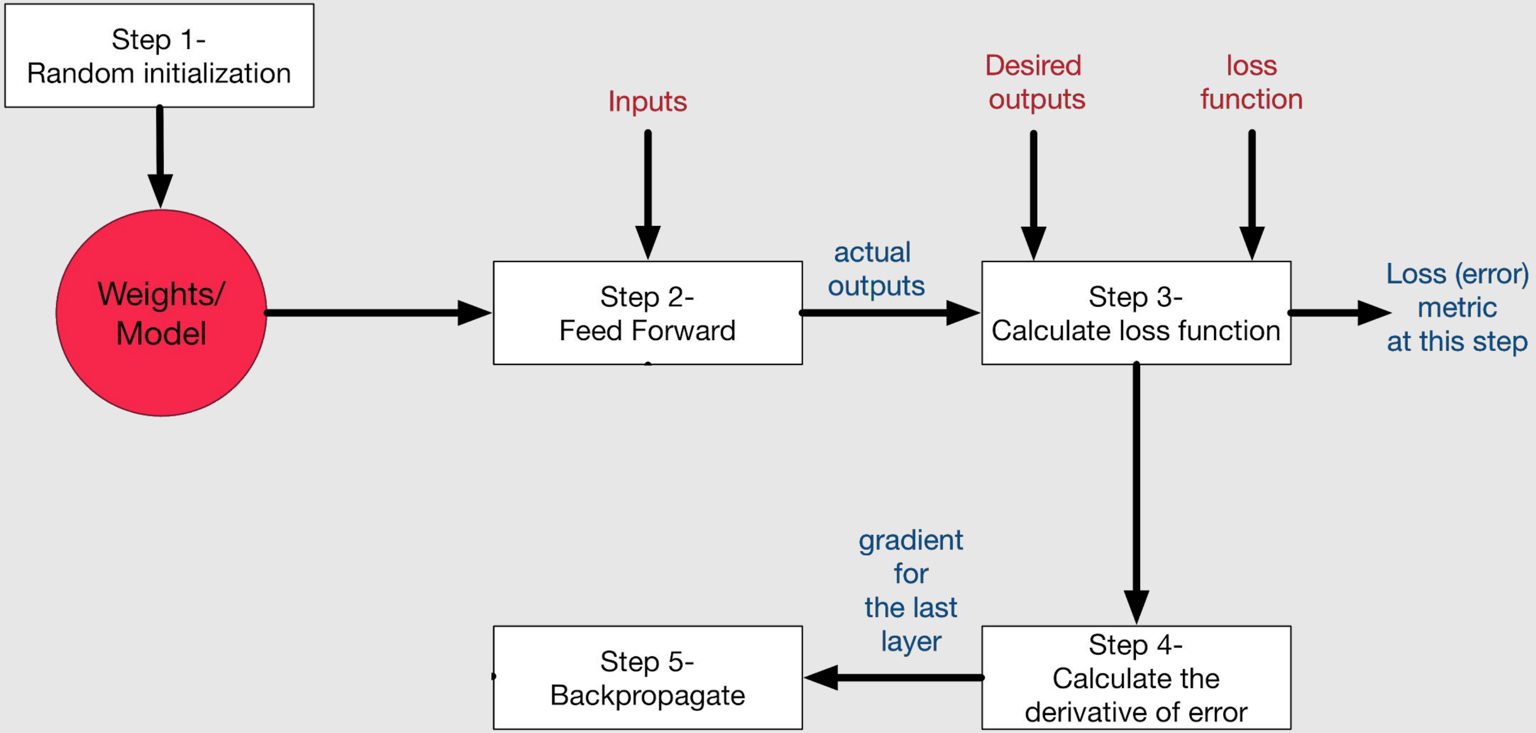


For each output unit (layer L = 4)

$$\delta_j^{(4)} = a_j^{(4)} - y_j$$

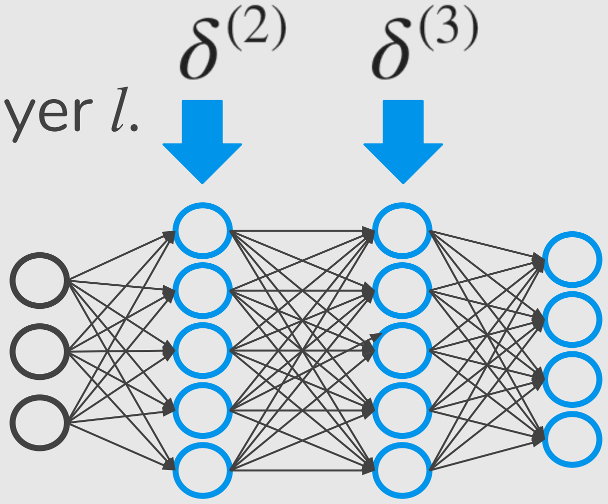
$$(h_\Theta(x))_j$$

Training a Neural Network



Gradient Computation: Backpropagation Algorithm

Intuition: $\delta_j^{(l)}$ = "error" of node j in layer l.



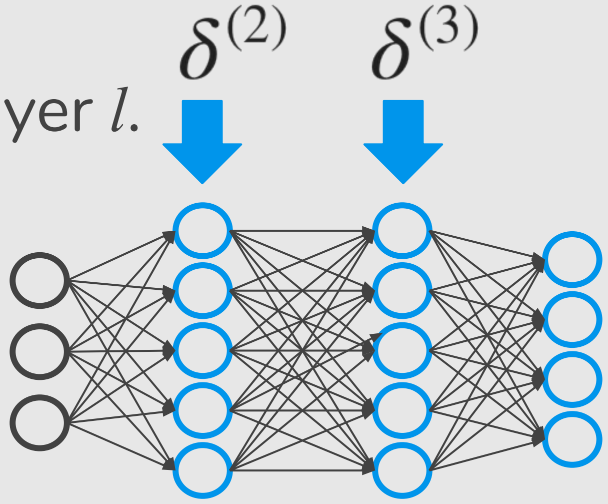
For each output unit (layer $L = 4$)

$$\delta_j^{(4)} = a_j^{(4)} - y_j$$

For each hidden unit

Gradient Computation: Backpropagation Algorithm

Intuition: $\delta_j^{(l)}$ = "error" of node j in layer l.



For each output unit (layer $L = 4$)

$$\delta_j^{(4)} = a_j^{(4)} - y_j$$

For each hidden unit

$$\delta^{(3)} = (\Theta^{(3)})^T \delta^{(4)} .* g'(z^{(3)})$$

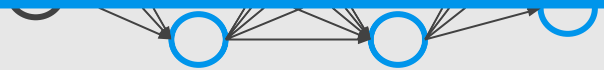
$$\delta^{(2)} = (\Theta^{(2)})^T \delta^{(3)} .* g'(z^{(2)})$$

$$\boxed{\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = a_j^{(l)} \delta_i^{(l+1)}}$$

Gradient Computation: Backpropagation Algorithm

Intuition: $\delta_j^{(l)}$ = "error" of node j in layer l.

Proof: https://theclevermachine.wordpress.com/2014/09/06/derivation-error-backpropagation-gradient-descent-for-neural-networks



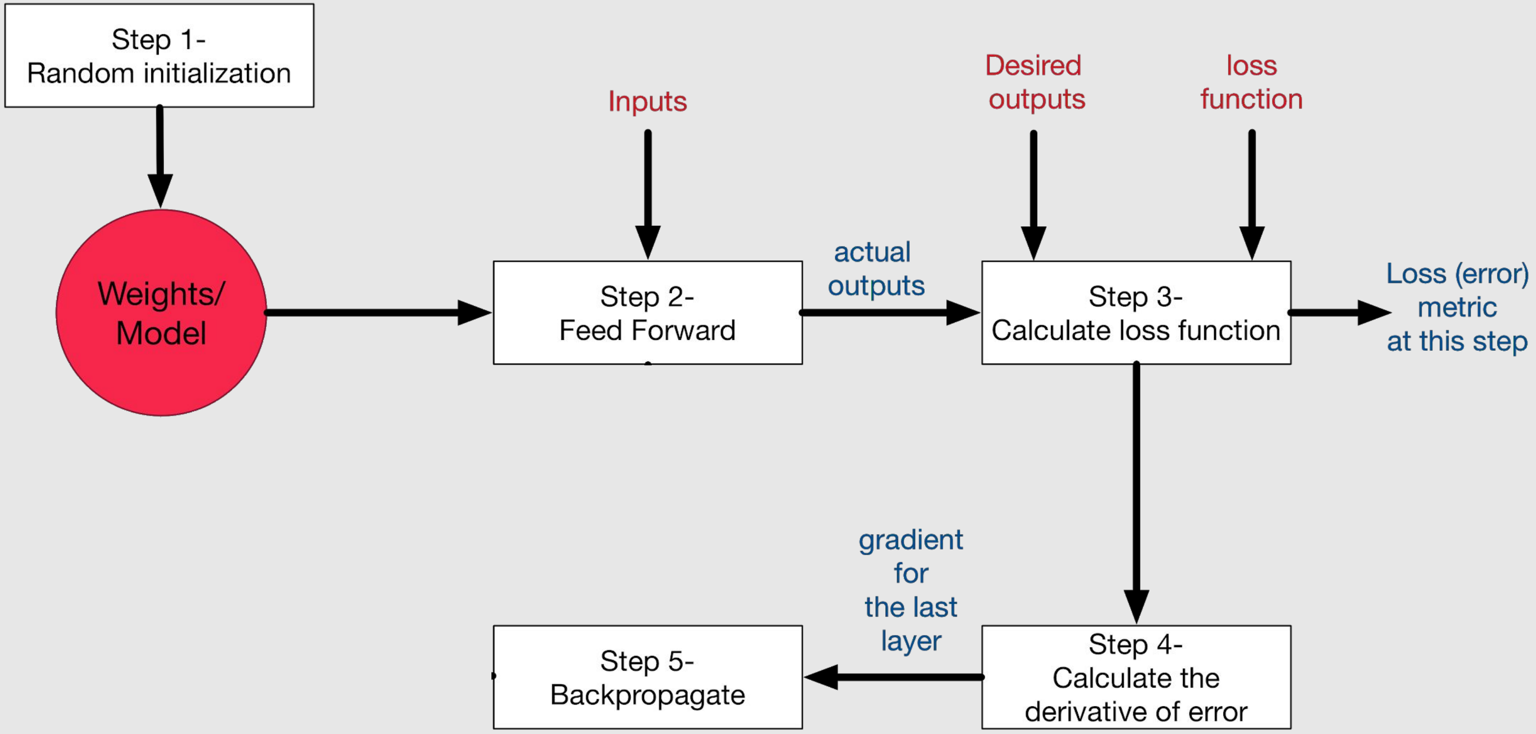
For each hidden unit

$$\delta^{(3)} = (\Theta^{(3)})^{T}\delta^{(4)} .* g'(z^{(3)})$$

$$\delta^{(2)} = (\Theta^{(2)})^{T}\delta^{(3)} .* g'(z^{(2)})$$

$$\boxed{\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = a_j^{(l)} \delta_i^{(l+1)}}$$

Training a Neural Network



Backpropagation Algorithm

Training Set: $(x^{(1)}, y^{(1)}),\ (x^{(2)}, y^{(2)}),\ \ldots,\ (x^{(m)}, y^{(m)})$

Backpropagation Algorithm

Training Set: $(x^{(1)},y^{(1)}),\ (x^{(2)},y^{(2)}),\ \ldots,\ (x^{(m)},y^{(m)})$

Set $\Delta_{ij}^{(l)} = 0$ (for all l, i, j)

For $i = 1$ to m

Set $a^{(1)} = x^{(i)}$

Performed forward propagation to compute $a^{(l)}$ for $l = 2, 3, \ldots, L$

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$

Compute $\delta^{(L-1)},\delta^{(L-2)},\ldots,\delta^{(2)}$

$$\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)}\delta_i^{(l+1)}$$

$$D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)}$$

Backpropagation Algorithm

Training Set: $(x^{(1)}, y^{(1)}),\ (x^{(2)}, y^{(2)}),\ \ldots,\ (x^{(m)}, y^{(m)})$

Set $\Delta_{ij}^{(l)} = 0$ (for all l, i, j)

For i = 1 to m

Set $a^{(1)} = x^{(i)}$

Performed forward propagation to compute $a^{(l)}$ for $l = 2, 3, \ldots, L$

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$

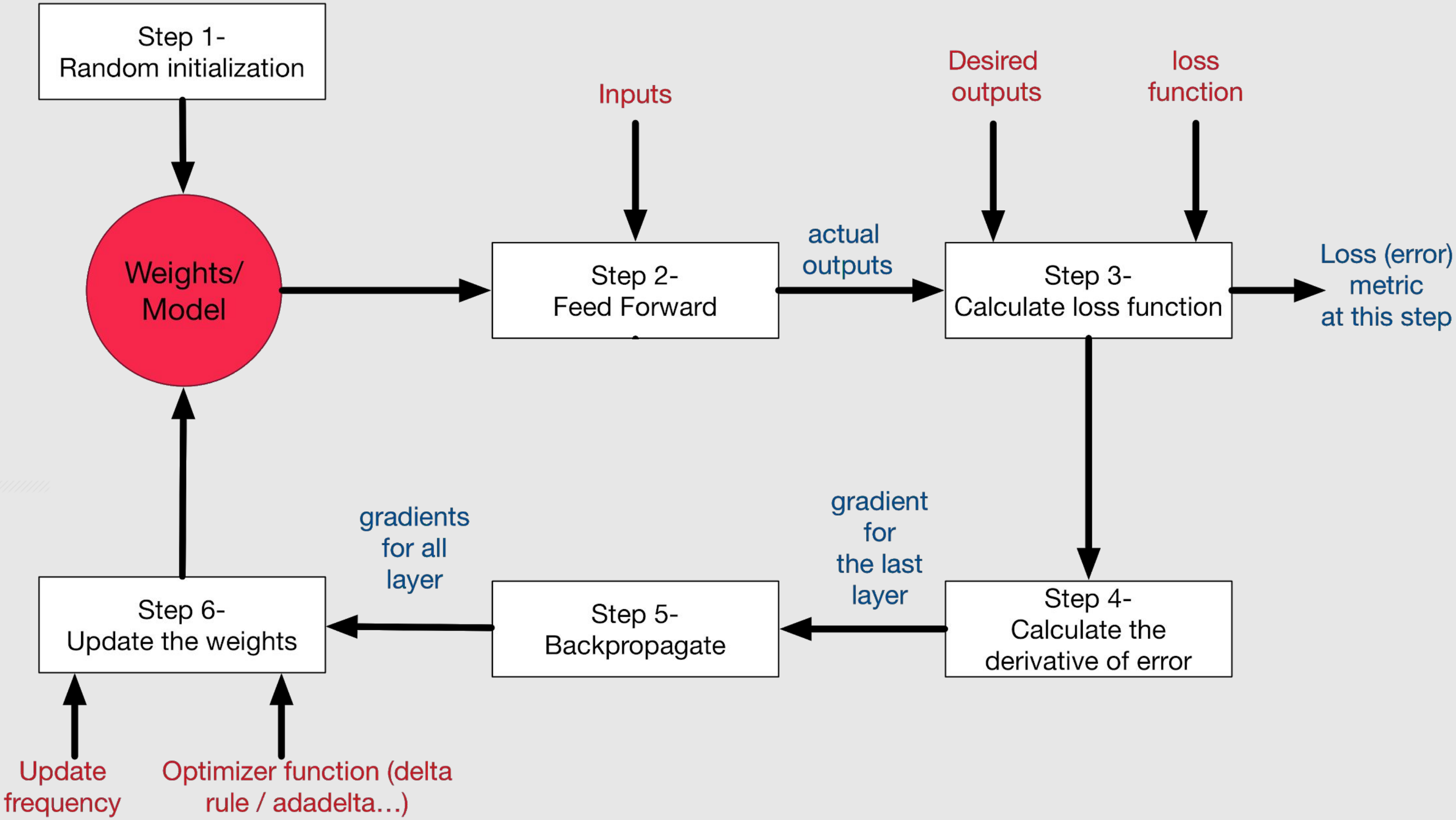
Compute $\delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(2)}$

$$\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$$

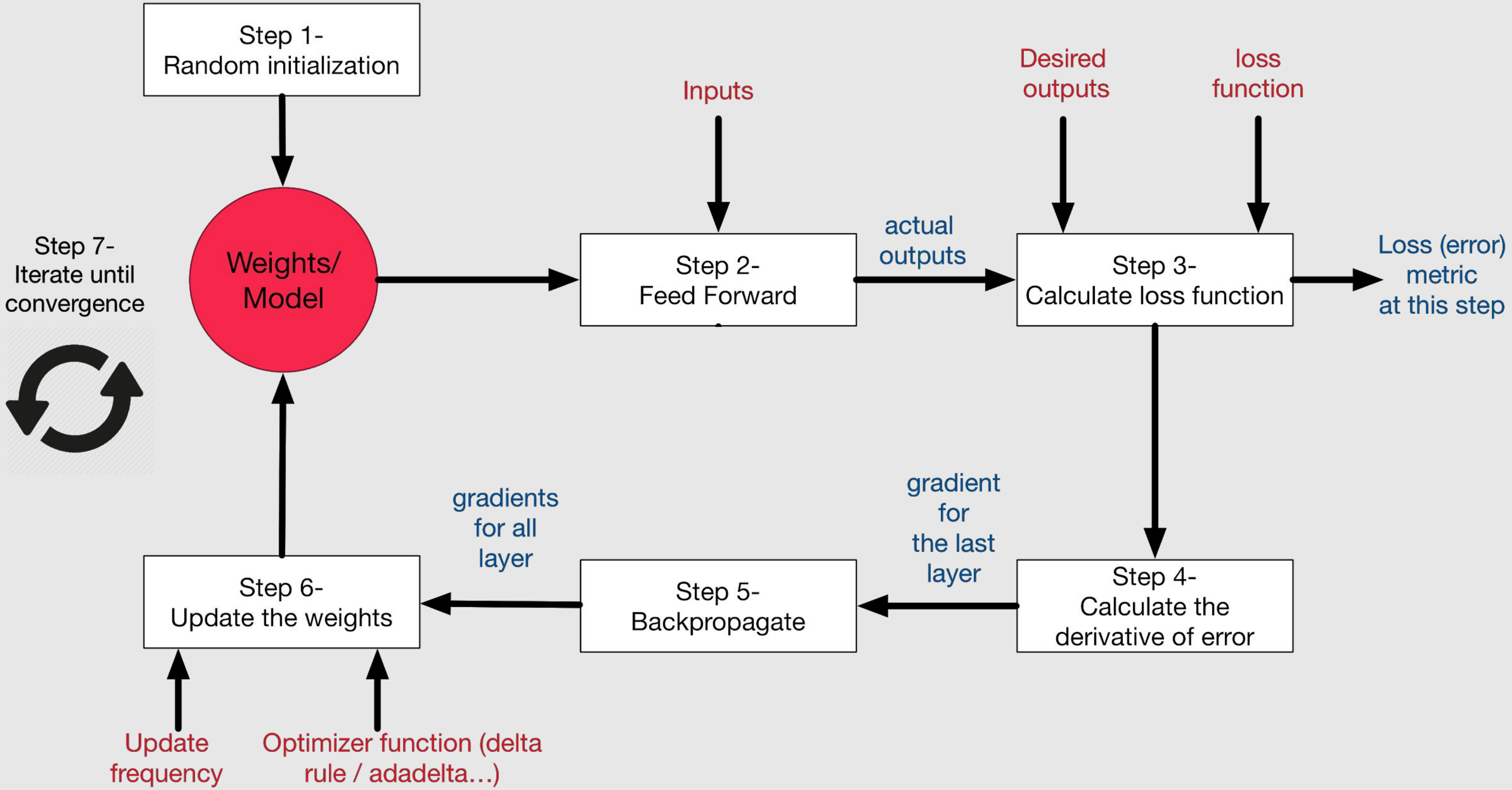
$$\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta) = D_{ij}^{(l)}$$

$$D_{ij}^{(l)} := \frac{1}{m} \Delta_{ij}^{(l)}$$

Training a Neural Network

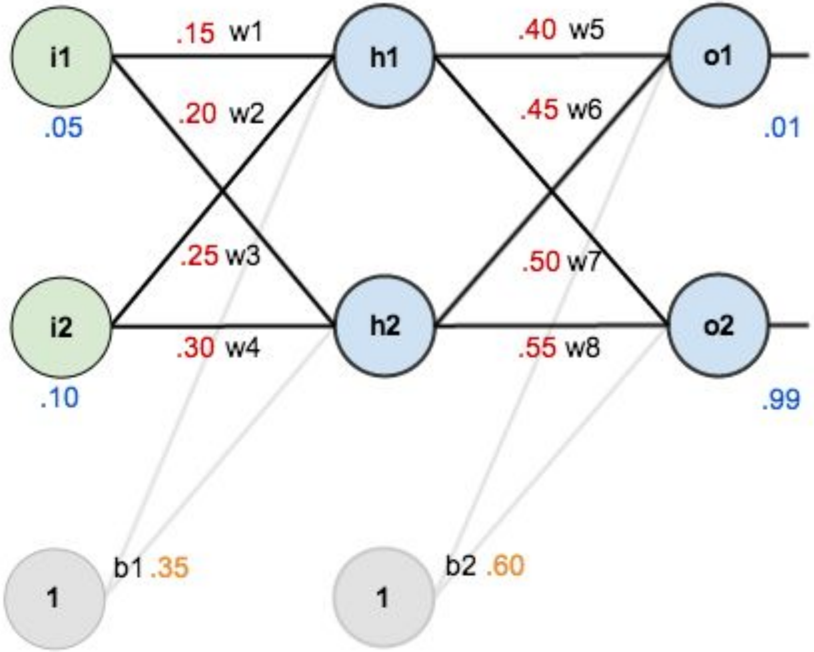


Training a Neural Network



A Step by Step Backpropagation Example

Given inputs 0.05 and 0.10,
we want the neural network to output 0.01 and 0.99.



Initial weights, the biases, and training inputs/outputs.

Neural Networks and Deep Learning: http://neuralnetworksanddeeplearning.com

CHAPTER 3

Improving the way neural networks learn

When a golf player is first learning to play golf, they usually spend most of their time developing a basic swing. Only gradually do they develop other shots, learning to chip, draw and fade the ball, building on and modifying their basic swing. In a similar way, up to now we've focused on understanding the backpropagation algorithm. It's our "basic swing", the foundation for learning in most work on neural networks. In this chapter I explain a suite of techniques which can be used to improve on our vanilla implementation of backpropagation, and so improve the way our networks learn.

The techniques we'll develop in this chapter include: a better choice of cost function, known as the cross-entropy cost function; four so-called "regularization" methods (L1 and L2 regularization, dropout, and artificial expansion of the training data), which make our networks better at generalizing beyond the training data; a better method for initializing the weights in the network; and a set of heuristics to help choose good hyper-parameters for the network. I'll also overview several other techniques in less depth

References

Machine Learning Books

- Hands-On Machine Learning with Scikit-Learn and TensorFlow, Chap. 10
- Pattern Recognition and Machine Learning, Chap. 5
- Pattern Classification, Chap. 6
- Free online book: **http://neuralnetworksanddeeplearning.com**

Machine Learning Courses

- https://www.coursera.org/learn/machine-learning, Week 4 & 5
- https://www.coursera.org/learn/neural-networks