Artificial Neural Networks

Machine Learning and Pattern Recognition

Prof. Sandra Avila
Institute of Computing (IC/Unicamp)

MC886/MO444, September 15, 2017

What does an artificial neuron do?

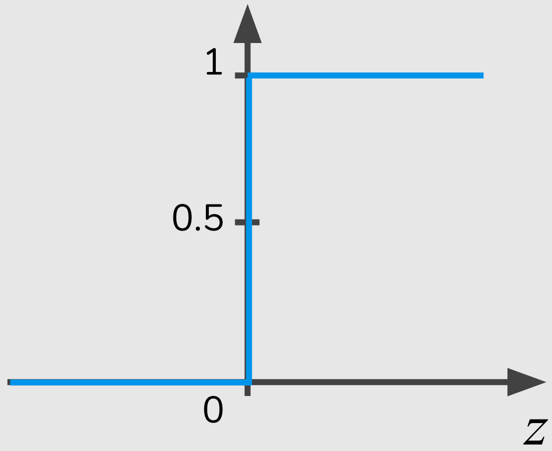
It calculates a “weighted sum” of its input, adds a bias and then decides whether it should be “fired” or not.

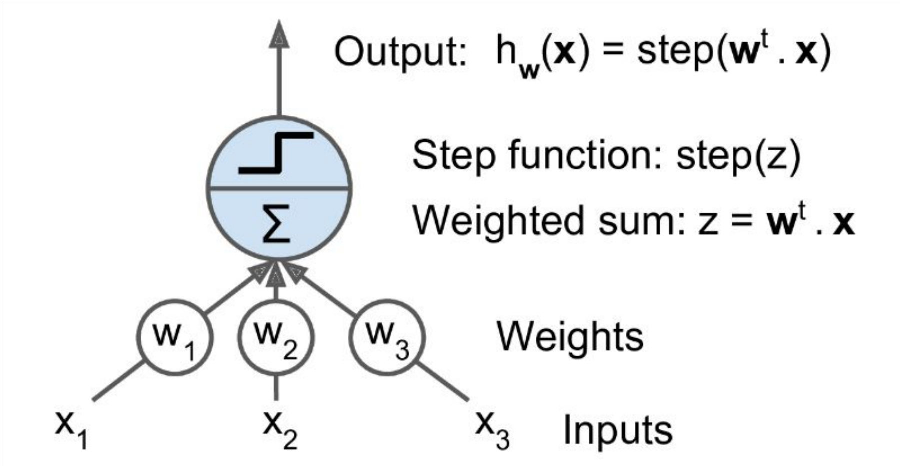
How do we decide whether the neuron should fire or not?

We decided to add “activation functions” for this purpose.

Step Function

Its output is **1 (activated)** when value > 0 (threshold) and outputs a **0 (not activated)** otherwise.





Step Function: **Problem?**

Step Function: **Problem?**

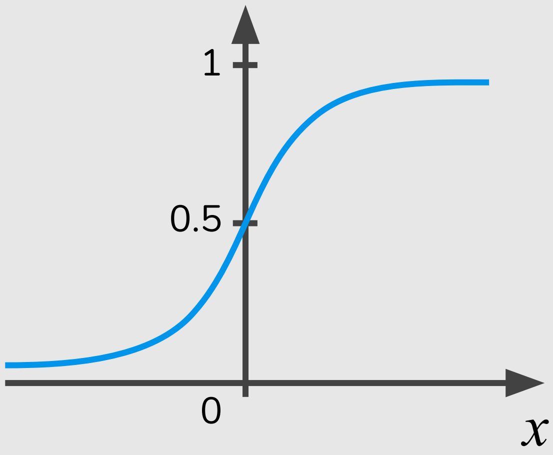
- Binary classifier (“yes” or “no”, activate or not activate). A Step function could do that for you!

Step Function: **Problem?**

- Binary classifier ("yes" or "no", activate or not activate). A Step function could do that for you!
- Multi classifier (class1, class2, class3, etc). What will happen if more than 1 neuron is "activated"?

Sigmoid Function

- The output of the activation function is always going to be in range **(0,1)**.
- It is nonlinear in nature.
- Combinations of this function are also nonlinear! Great!!



$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Sigmoid Function: **Problem?**

Sigmoid Function: **Problem?**

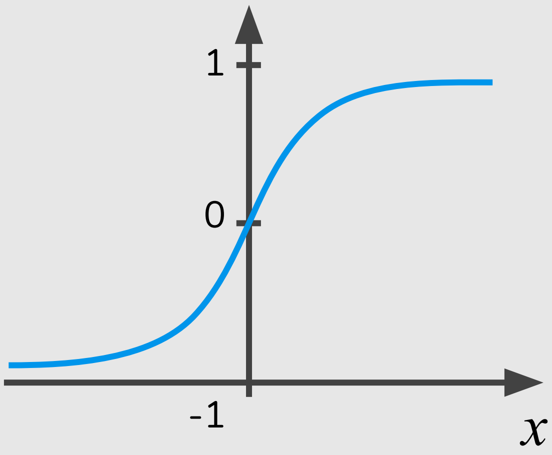
- Towards either end of the sigmoid function, the $\sigma(x)$ values tend to respond very less to changes in x.

Sigmoid Function: **Problem?**

- Towards either end of the sigmoid function, the $\sigma(x)$ values tend to respond very less to changes in x.
- The problem of “vanishing gradients”.
    - Cannot make significant change because of the extremely small value.

Tanh Function

- The output of the activation function is always going to be in range **(-1,1)**.
- It is nonlinear in nature.
- Combinations of this function are also nonlinear! Great!!



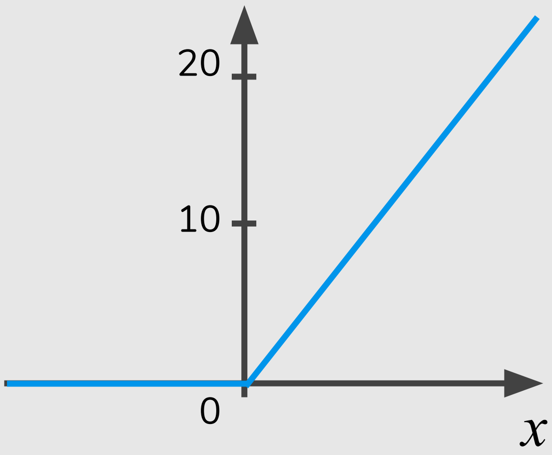
$$tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

Tanh Function: **Problem?**

- Like sigmoid, tanh also has the vanishing gradient problem.

ReLU Function

- It gives an output x if x is positive and 0 otherwise. The range is **[0, inf)**.
- It is nonlinear in nature. Combinations of this function are also nonlinear!
- Sparsity of the activation!



$$\mathrm{ReLU}(x) = max(0,x)$$

ReLU Function: **Problem?**

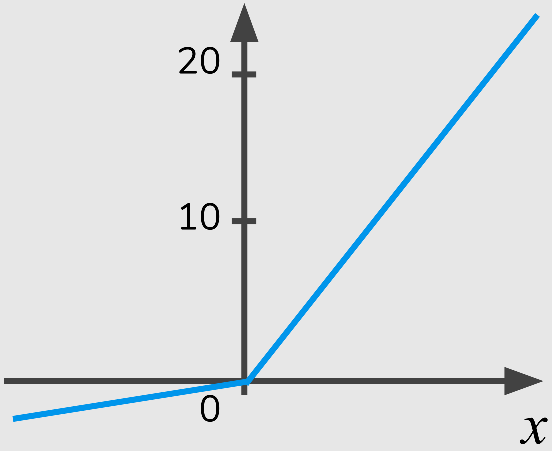
ReLU Function: **Problem?**

- Because of the horizontal line in ReLU(for negative x), the gradient can go towards 0.

- “Dying ReLU problem”: several neurons can just die and not respond making a substantial part of the network passive.

Leaky ReLU Function

- It gives an output x if x is positive and 0 otherwise. The range is **[0, inf)**.
- (Leaky) ReLU is less computationally expensive than *tanh* and *sigmoid* because it involves simpler mathematical operations.



$$\text{Leaky ReLU}(x) = \begin{cases} x \text{ if } x > 0 \\ 0.01x \text{ otherwise} \end{cases}$$

Ok! Which One Do We Use?

Ok! Which One Do We Use?

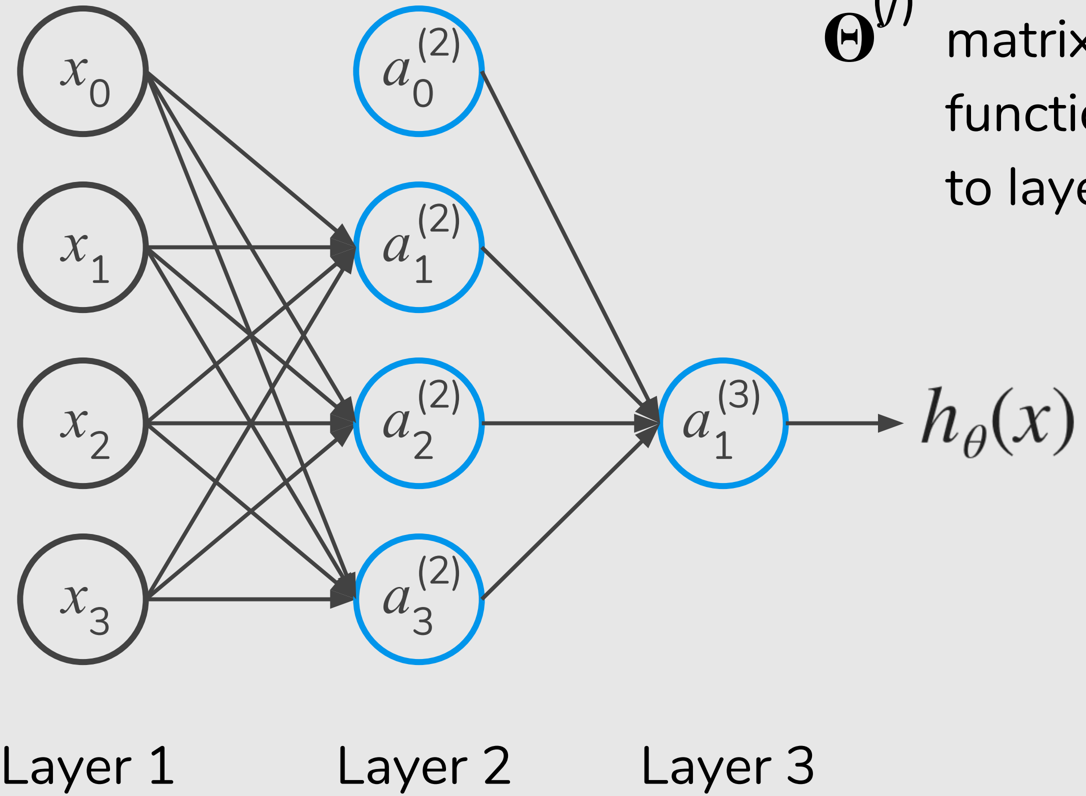
- If you don’t know the nature of the function you are trying to learn, start with ReLU.

Ok! Which One Do We Use?

- If you don’t know the nature of the function you are trying to learn, start with ReLU.
- You can use your own custom functions too!

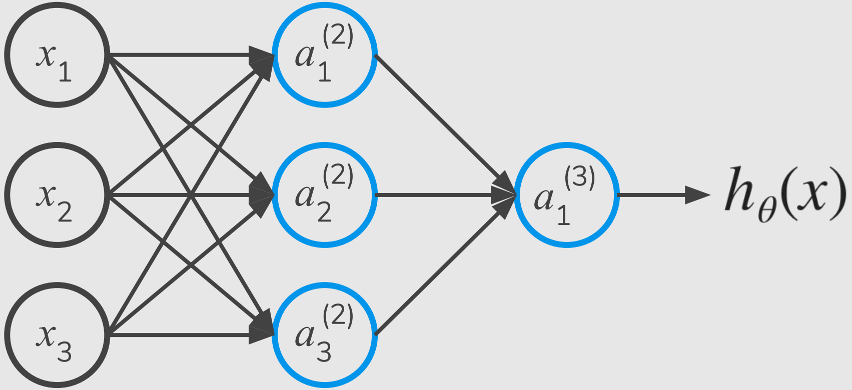
Neural Network Representation

Neural Network



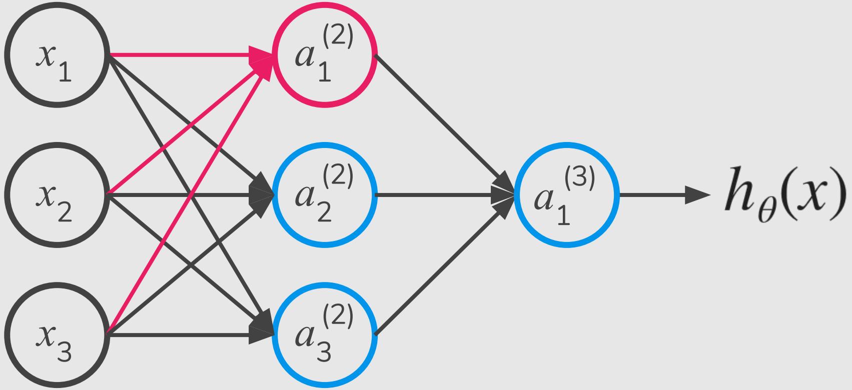
$a_i^{(j)}$ "activation" of unit i in layer j

$\mathbf{\Theta}^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$



$a_i^{(j)}$ "activation" of unit i in layer j

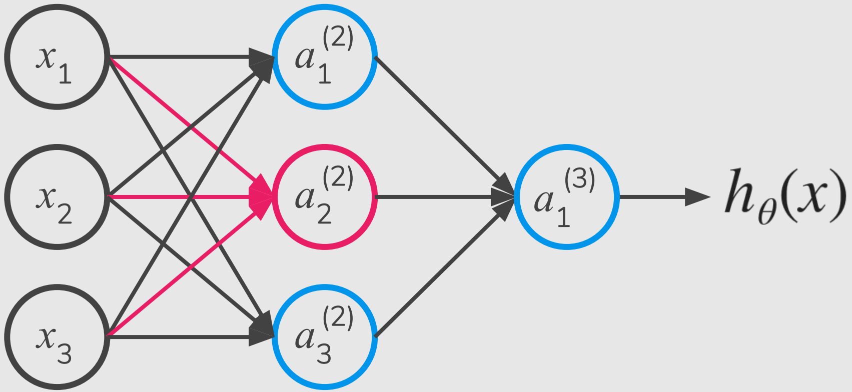
$\mathbf{\Theta}^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$



$a_i^{(j)}$ "activation" of unit i in layer j

$\mathbf{\Theta}^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

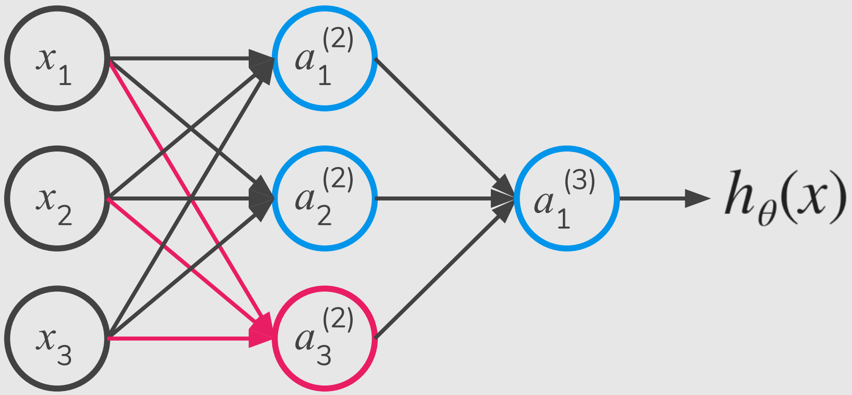


$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

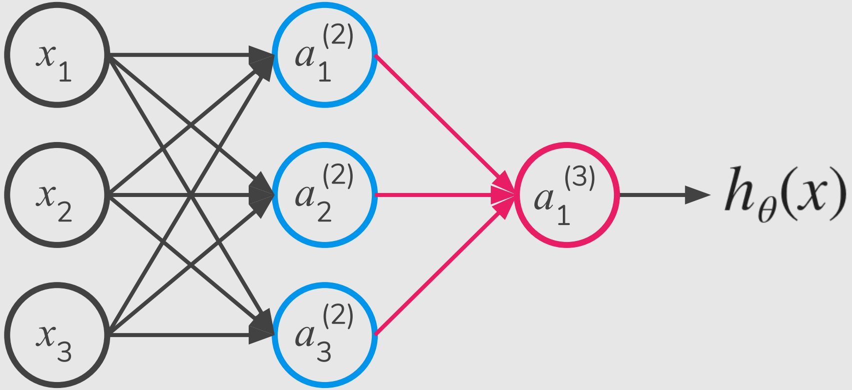
$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$



$a_i^{(j)}$ "activation" of unit i in layer j

$\boldsymbol{\Theta}^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$



$a_i^{(j)}$ "activation" of unit i in layer j

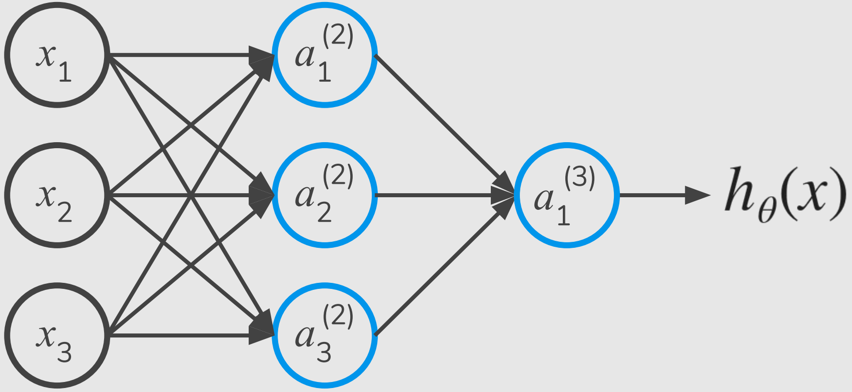
$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

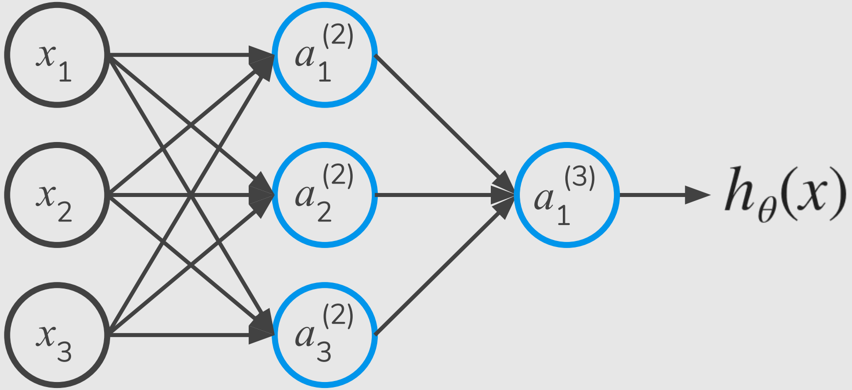


$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

Feedforward Neural Network (forward propagating)

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

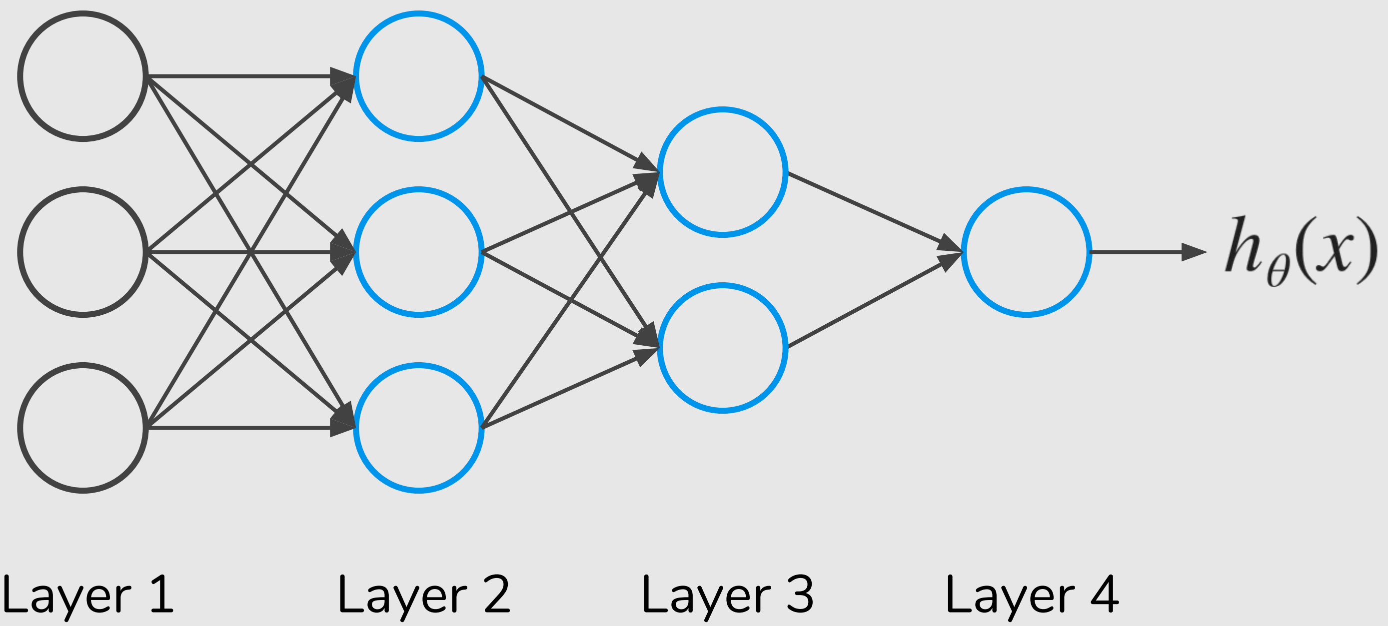


$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

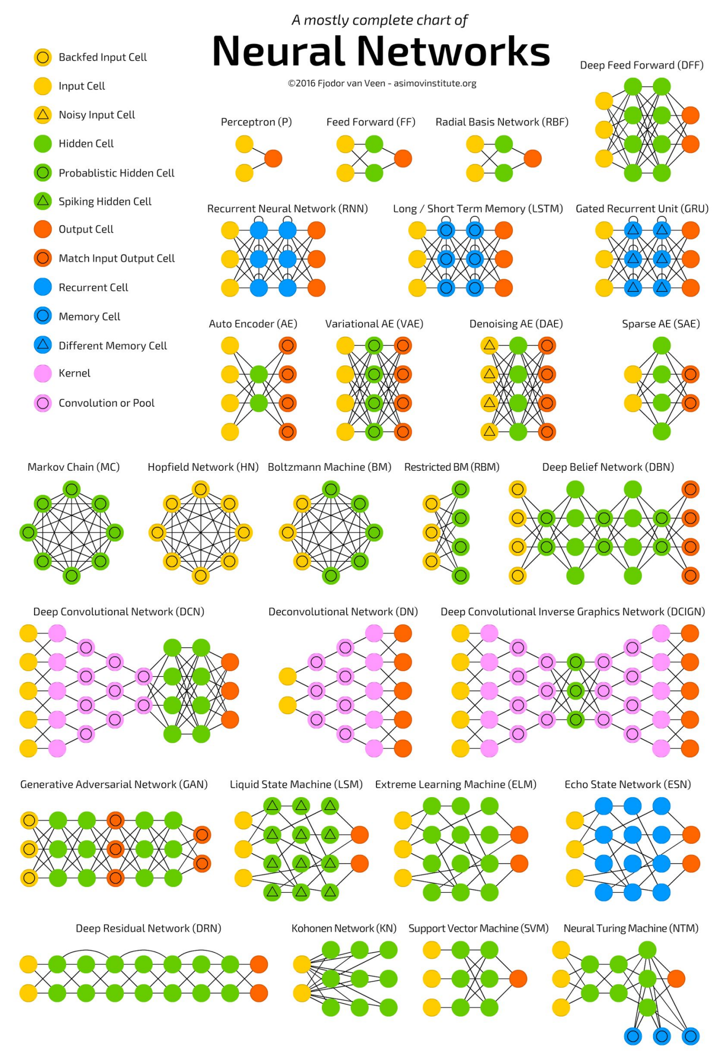
If network has s_j units in layer j, s_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be of dimension $s_{j+1} \times (s_j + 1)$.

Other Network Architectures

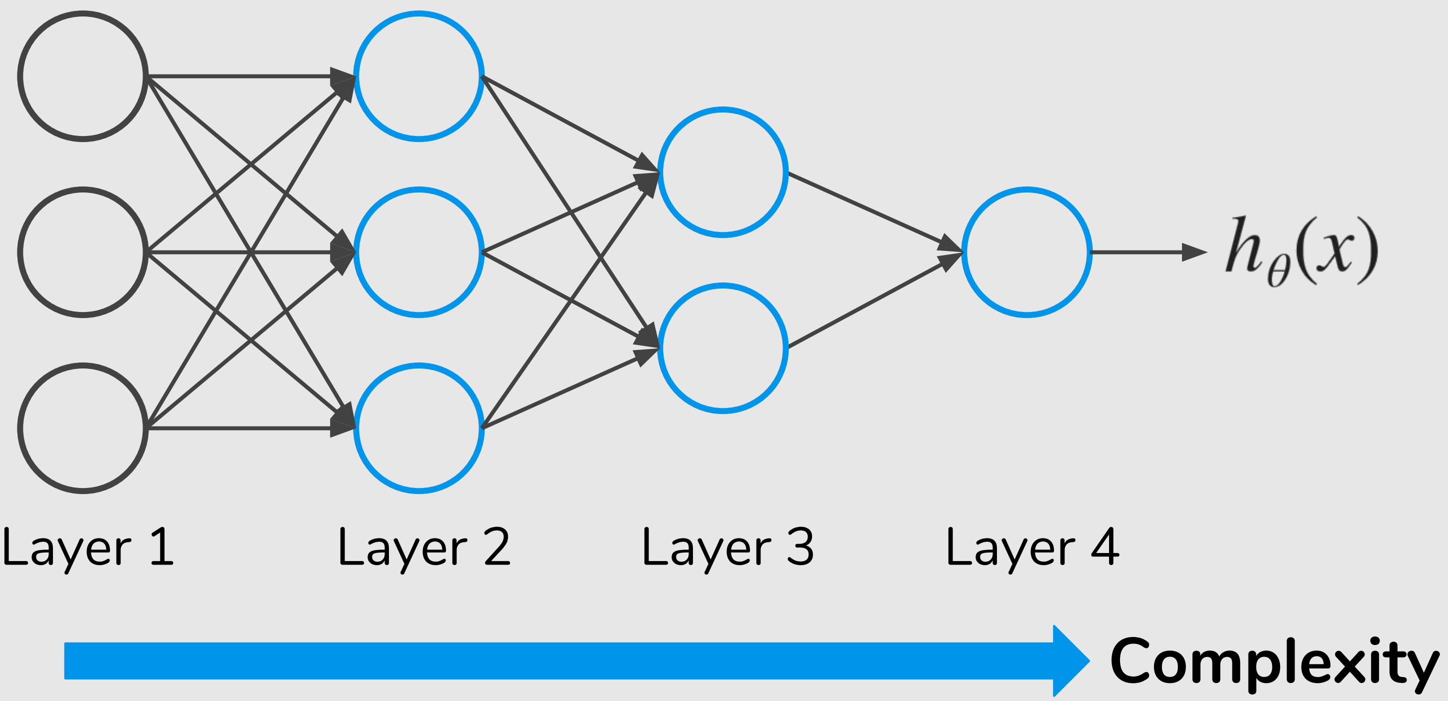


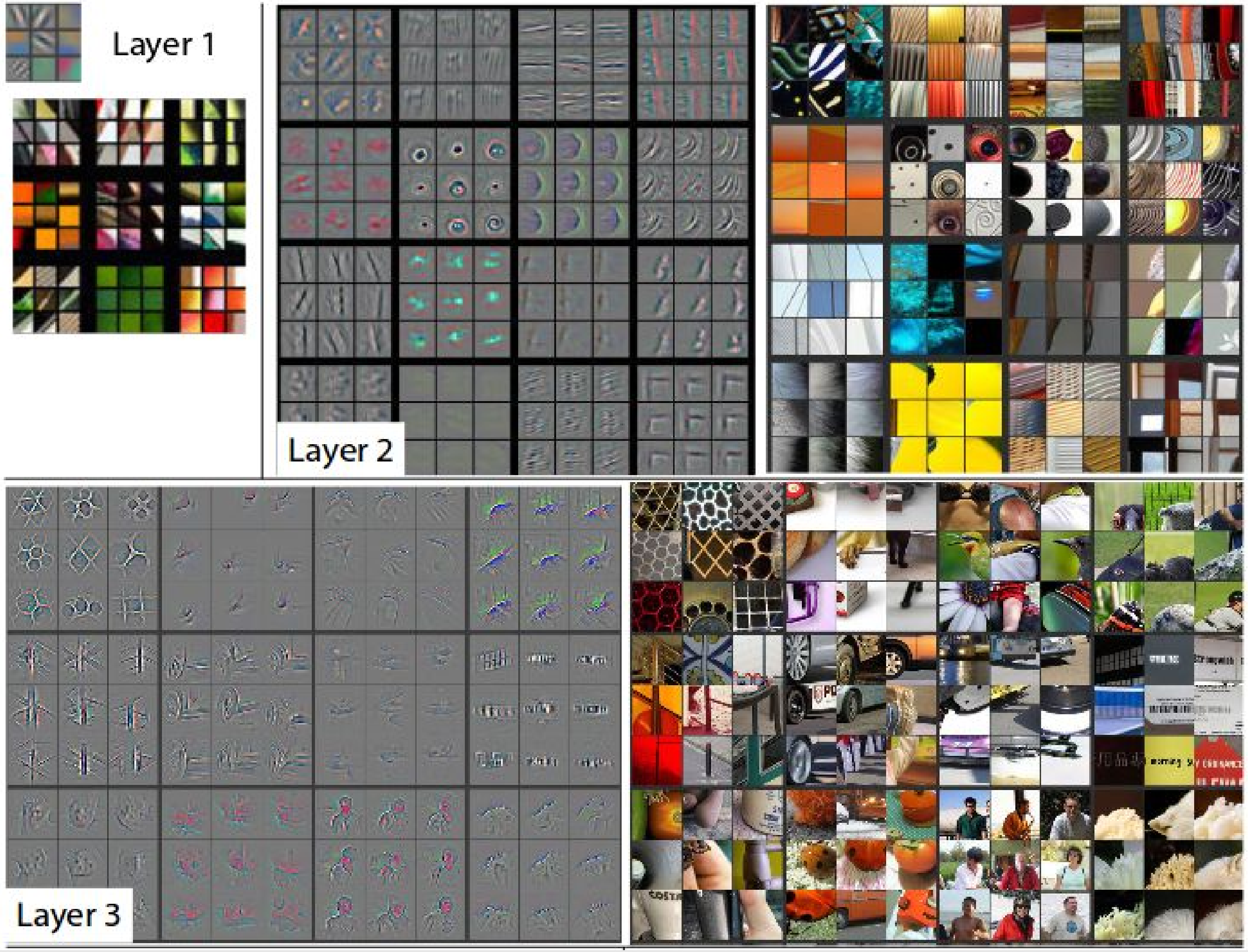
Neural Network Zoo

http://www.asimovinstitute.org/neural-network-zoo/

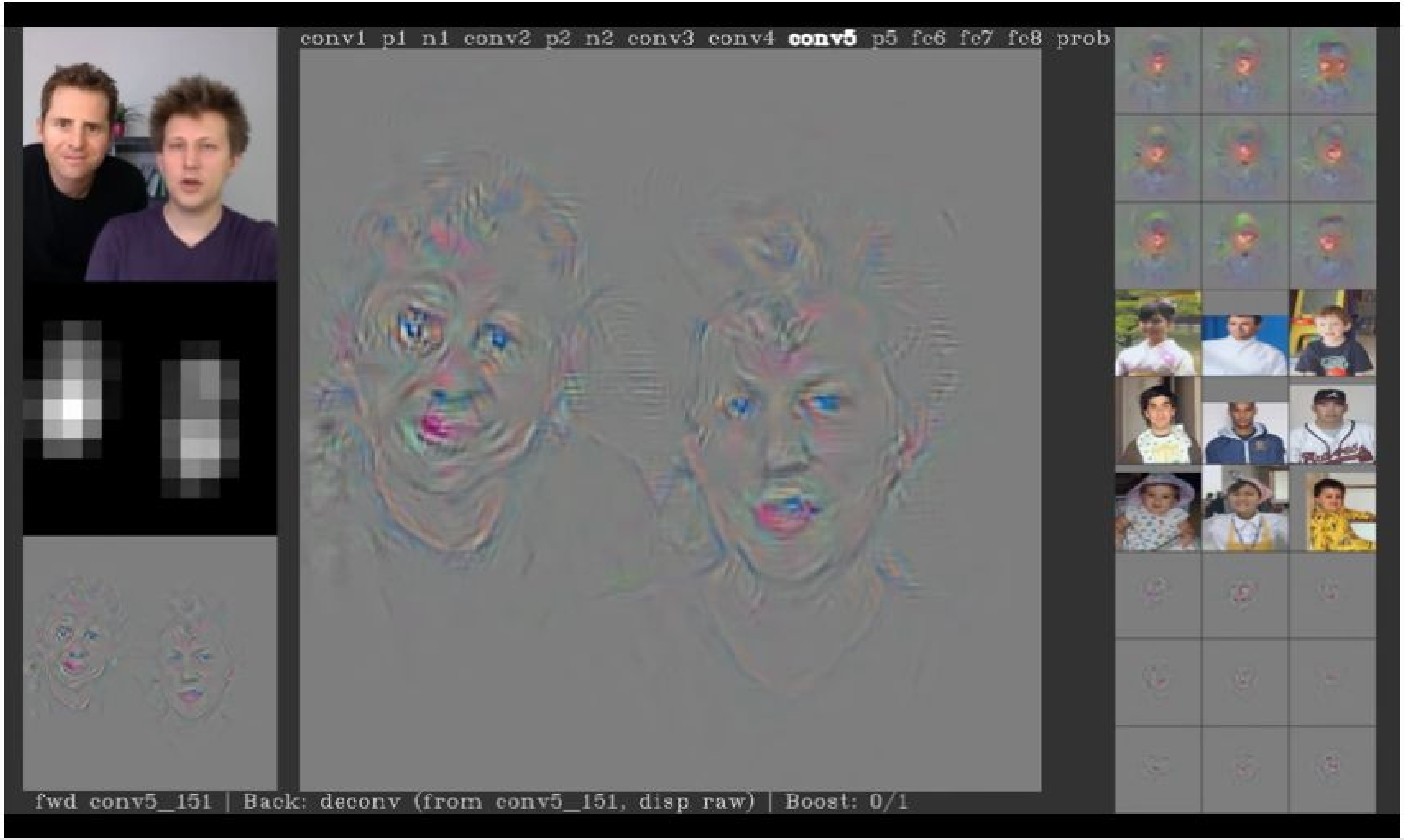


Neural Network Intuition





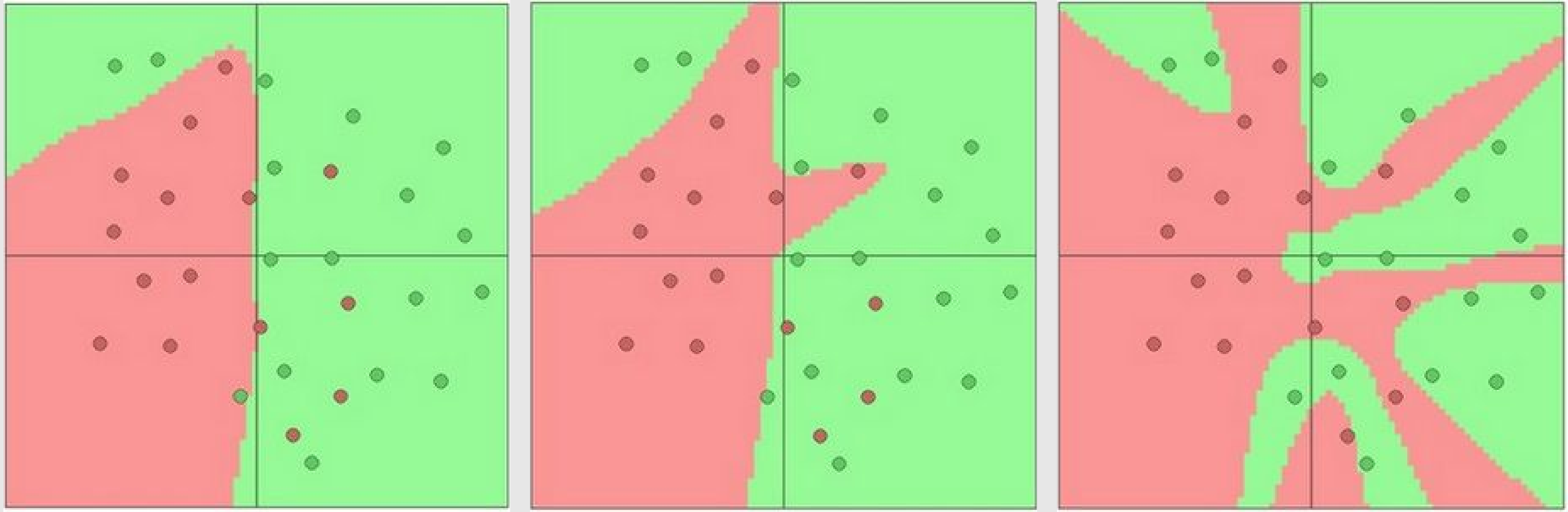
https://larseidnes.files.wordpress.com/2015/12/screenshot-from-2015-12-15-213302.png?w=1008 http://yosinski.com/deepvis https://youtu.be/AgkfIQ4IGaM



Neural Network Intuition

Toy 2d classification with 2-layer neural network

http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html



3 hidden neurons 6 hidden neurons 20 hidden neurons

http://cs231n.github.io/assets/nn1/layer_sizes.jpeg

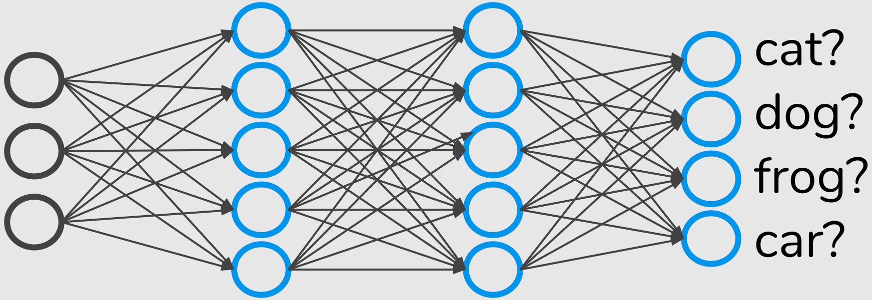
Multi-class Classification


Cat


Dog


Frog


Car

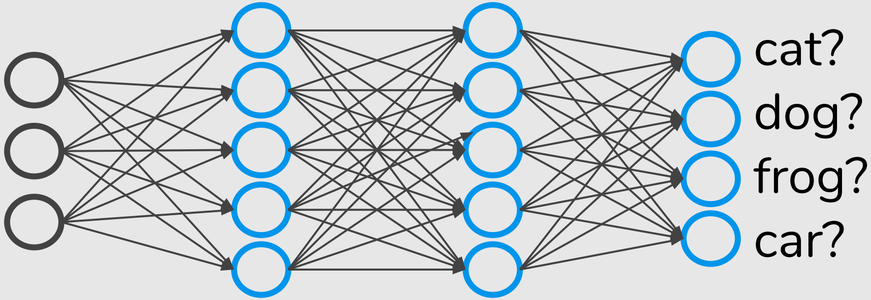


Cat

Dog

Frog

Car

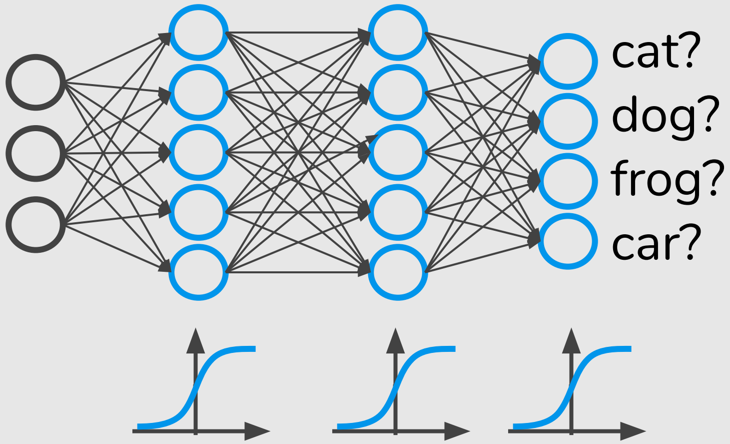


Want $h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

when cat | when dog | when frog | when car

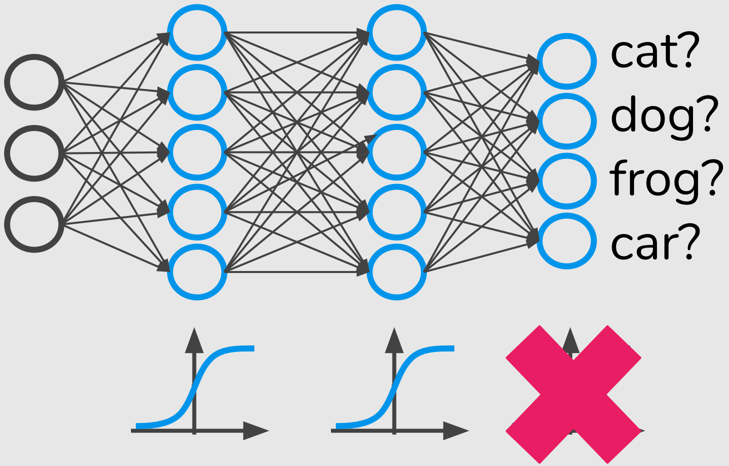
Softmax Classification

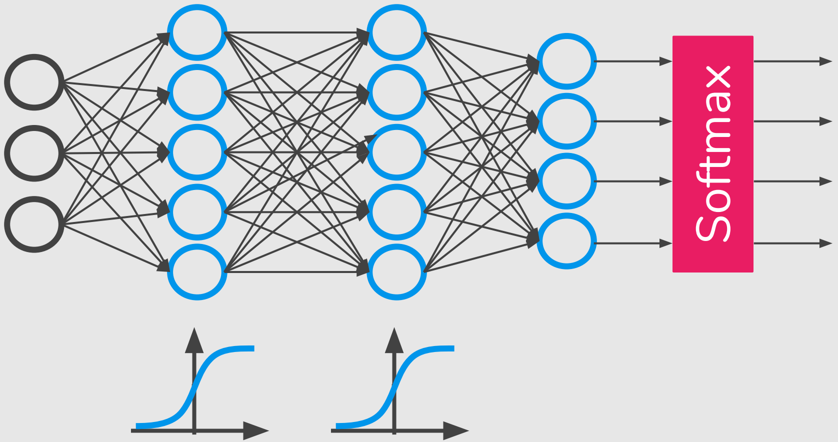
The **output layer** is typically modified **by replacing** the individual activation functions **by a shared softmax** function.



Softmax Classification

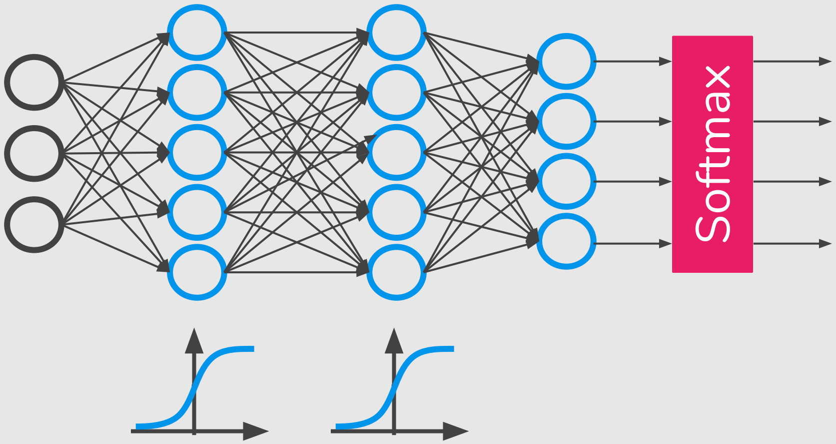
The **output layer** is typically modified **by replacing** the individual activation functions **by a shared softmax** function.





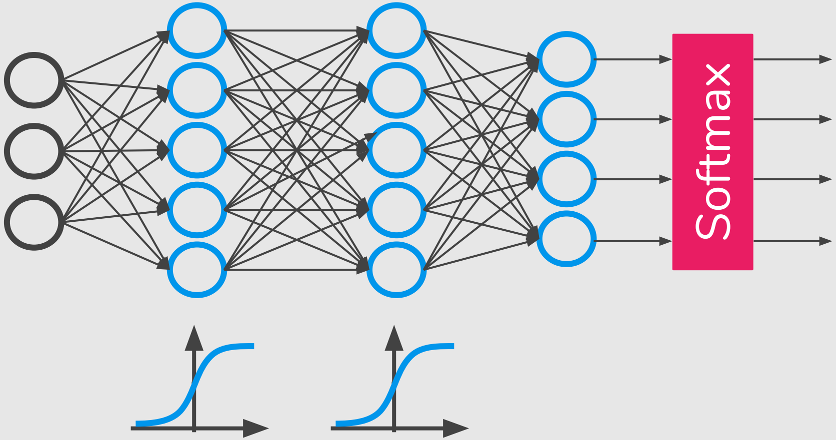
Softmax Classification

The **output layer** is typically modified **by replacing** the individual activation functions **by a shared softmax** function.



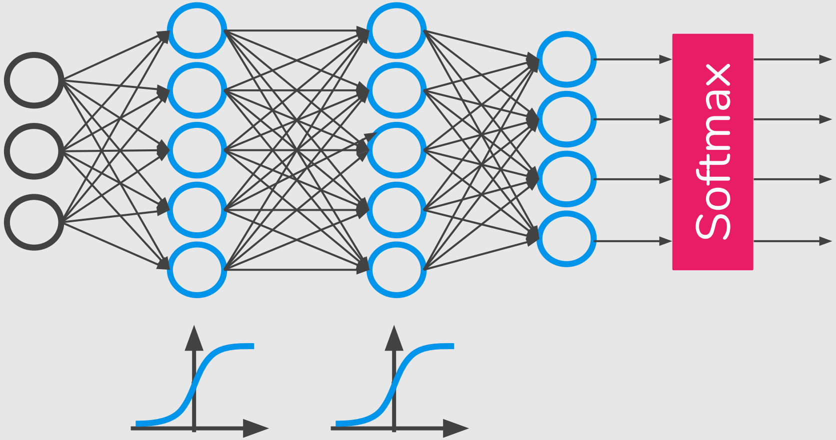
$$f(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$$

Softmax Classification



$$f(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$$

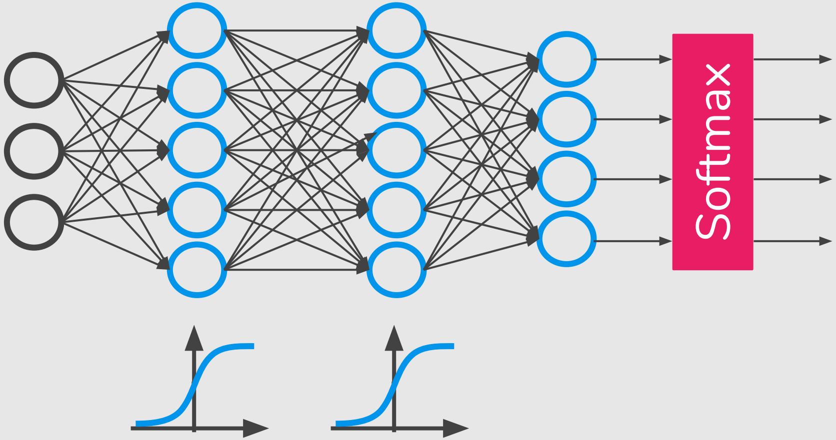
Softmax Classification



Cat	5.1
Dog	3.2
Frog	-1.7
Car	-2.0

$$\boxed{f(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}}$$

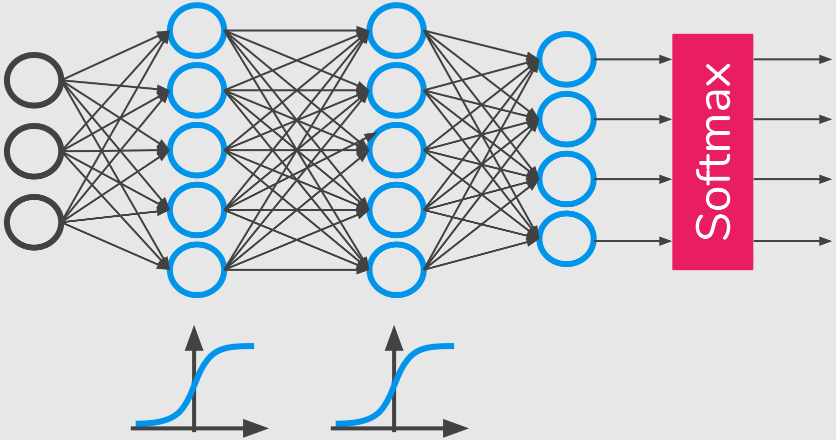
Softmax Classification



Cat	5.1	164.0
Dog	3.2	24.5
Frog	-1.7	0.18
Car	-2.0	0.13

$$f(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$$

Softmax Classification



Cat	5.1	164.0	0.87
Dog	3.2	24.5	0.13
Frog	-1.7	0.18	0.00
Car	-2.0	0.13	0.00

$$f(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$$

Cost Function

Cost Function

Let's first define a few variables that we will need to use:

- L = total number of **layers** in the network
- s_l = number of **units** (not counting bias unit) in layer l
- K = number of **output** units/classes

Cost Function

Let's first define a few variables that we will need to use:

- L = total number of **layers** in the network
- s_l = number of **units** (not counting bias unit) in layer l
- K = number of **output** units/classes

Our cost function for neural networks is going to be a generalization of the one we used for **logistic regression**.

Logistic Regression:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))\right]$$

Logistic Regression:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))\right]$$

Neural Network: $h_\Theta(x) \in \mathbb{R}^K$ $\quad (h_\Theta(x))_i = i^{th}$ output

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} \sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1-y_k^{(i)}) \log(1-(h_\Theta(x^{(i)}))_k)\right]$$

Logistic Regression:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta^2$$

Neural Network: $h_\Theta(x) \in \mathbb{R}^K \quad (h_\Theta(x))_i = i^{th}$ output

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1-y_k^{(i)}) \log(1-(h_\Theta(x^{(i)}))_k)\right]$$

$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\Theta_{ji}^{(l)})^2$$

Backpropagation

A Simple Example

Backpropagation: A Simple Example

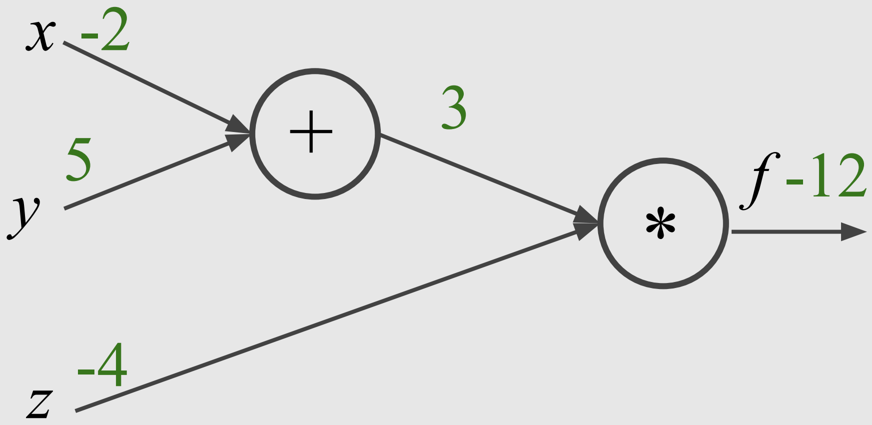
$f(x, y, z) = (x + y)z$

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

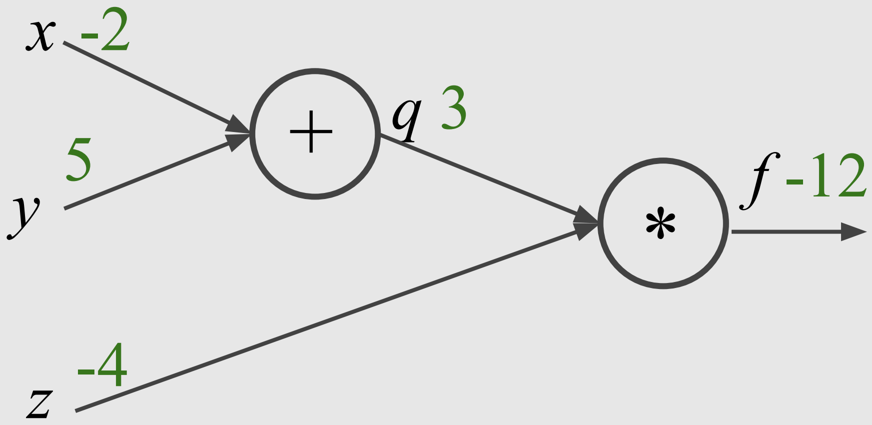
$$f(x, y, z) = (x + y)z$$

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

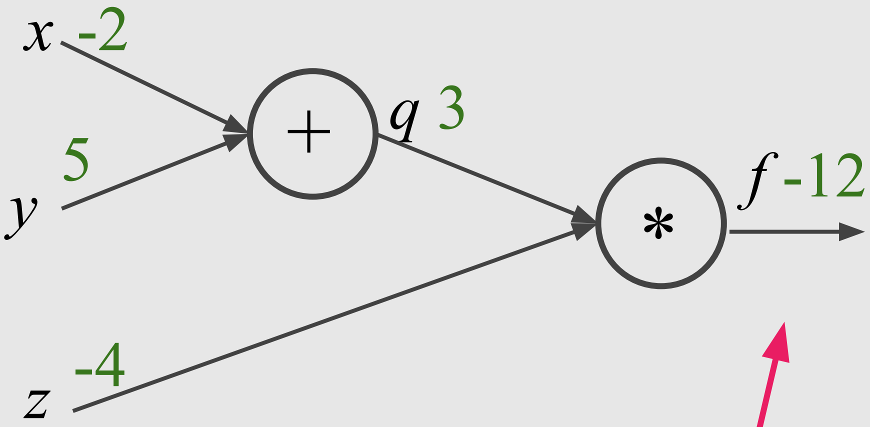
$$f(x, y, z) = (x + y)z$$

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

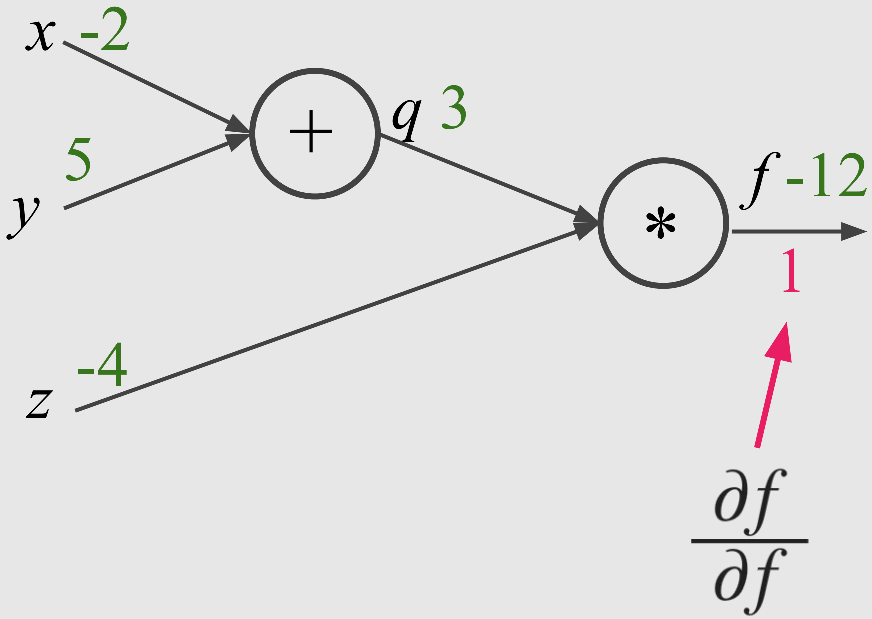
$$f(x, y, z) = (x + y)z$$

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

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

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

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



Backpropagation: A Simple Example

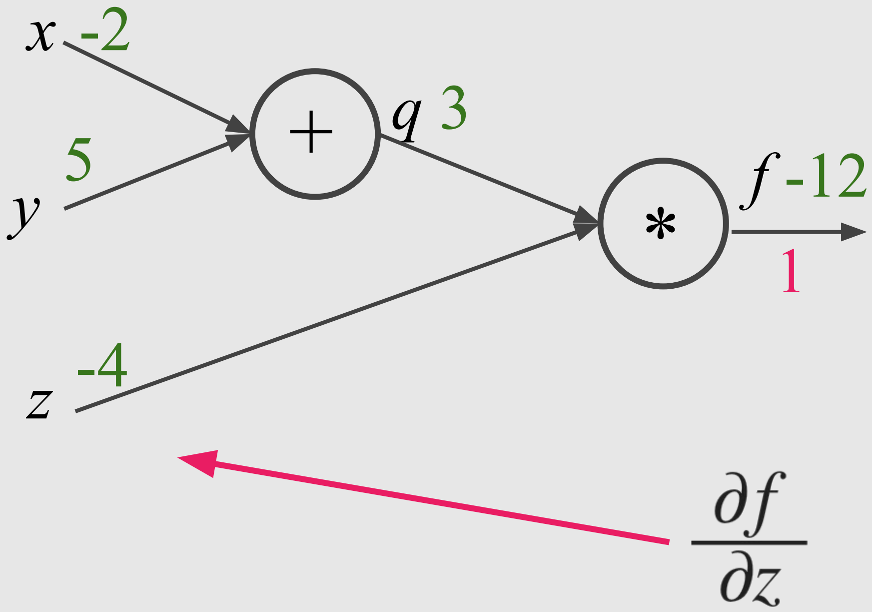
$$f(x, y, z) = (x + y)z$$

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: $\dfrac{\partial f}{\partial x}, \dfrac{\partial f}{\partial y}, \dfrac{\partial f}{\partial z}$



Backpropagation: A Simple Example

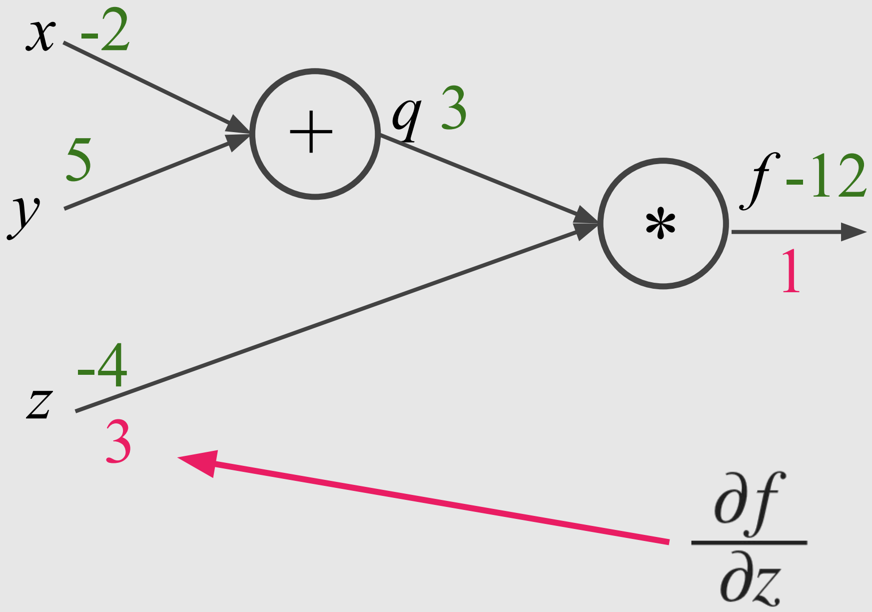
$$f(x, y, z) = (x + y)z$$

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

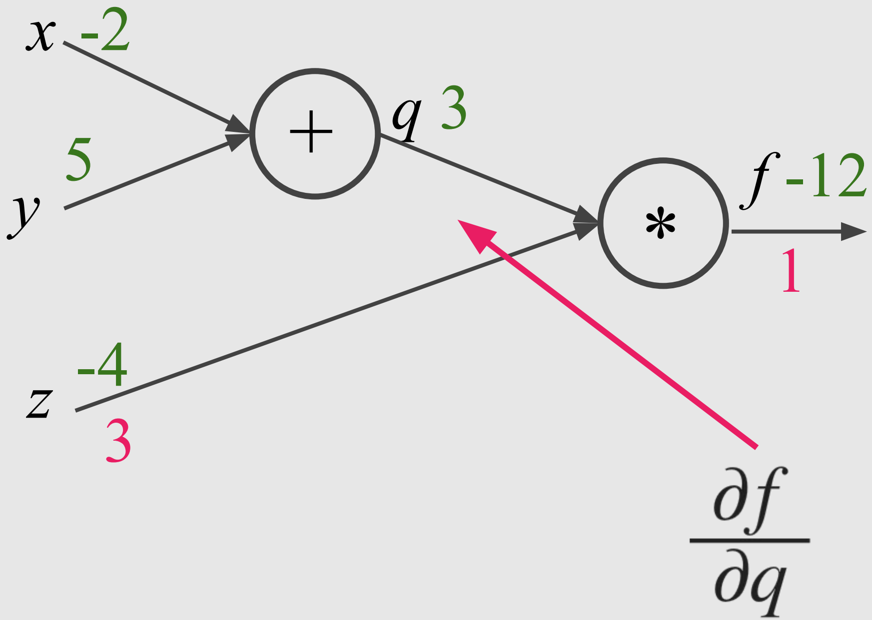
$$f(x, y, z) = (x + y)z$$

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

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

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

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



Backpropagation: A Simple Example

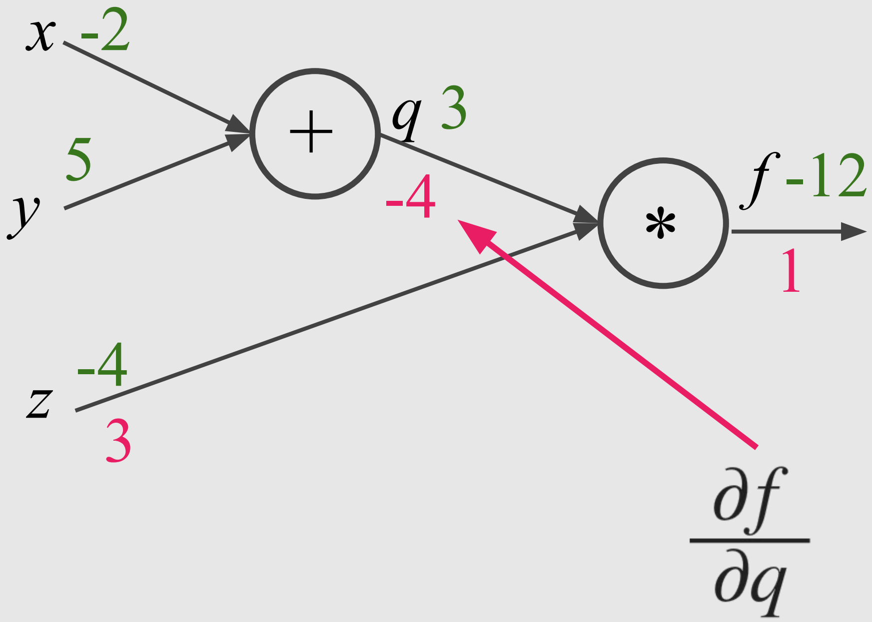
$$f(x, y, z) = (x + y)z$$

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

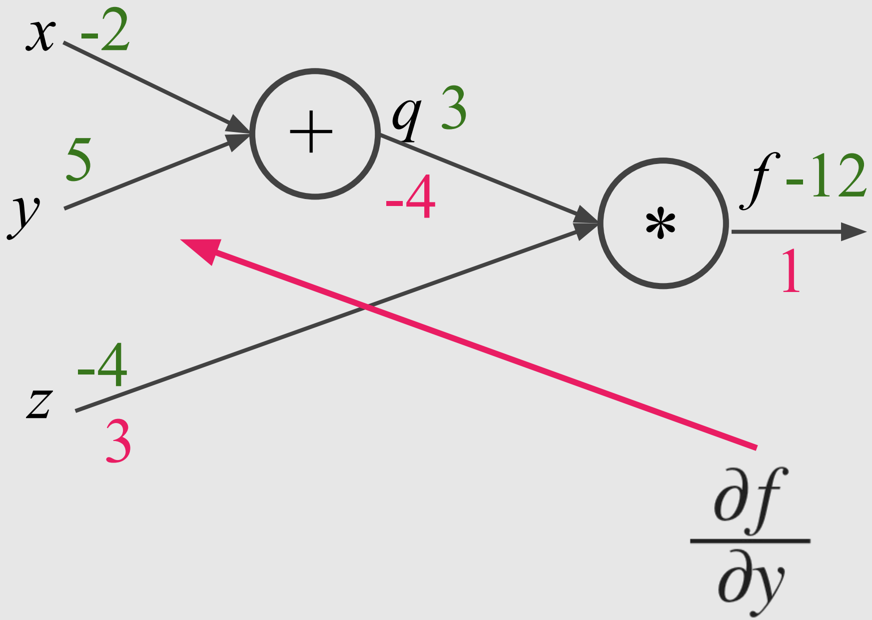
$$f(x, y, z) = (x + y)z$$

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

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

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

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



Backpropagation: A Simple Example

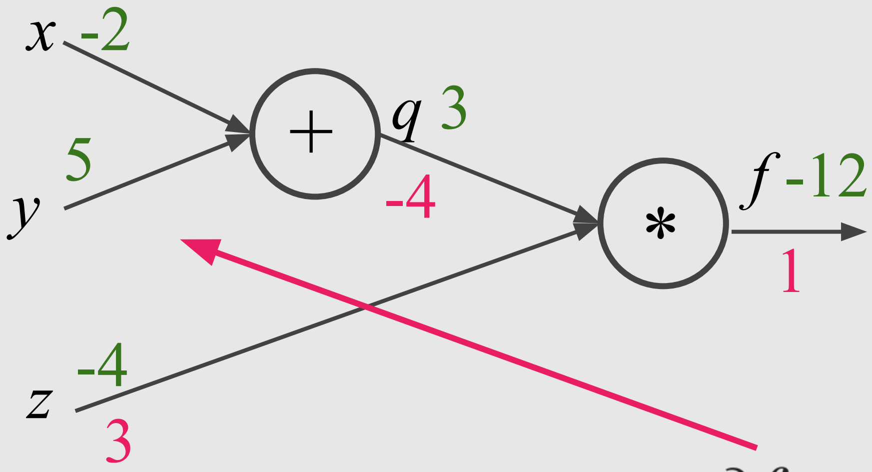
$$f(x, y, z) = (x + y)z$$

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

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

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

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

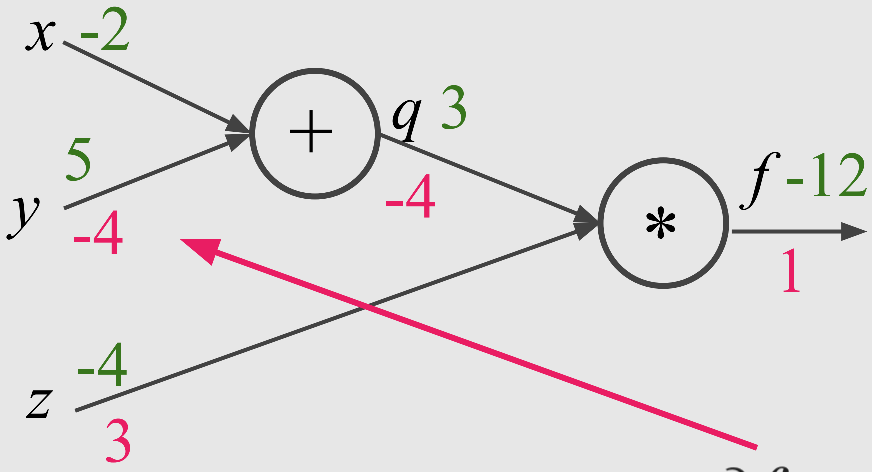
$$f(x, y, z) = (x + y)z$$

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

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

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

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



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

Chain rule

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

Backpropagation: A Simple Example

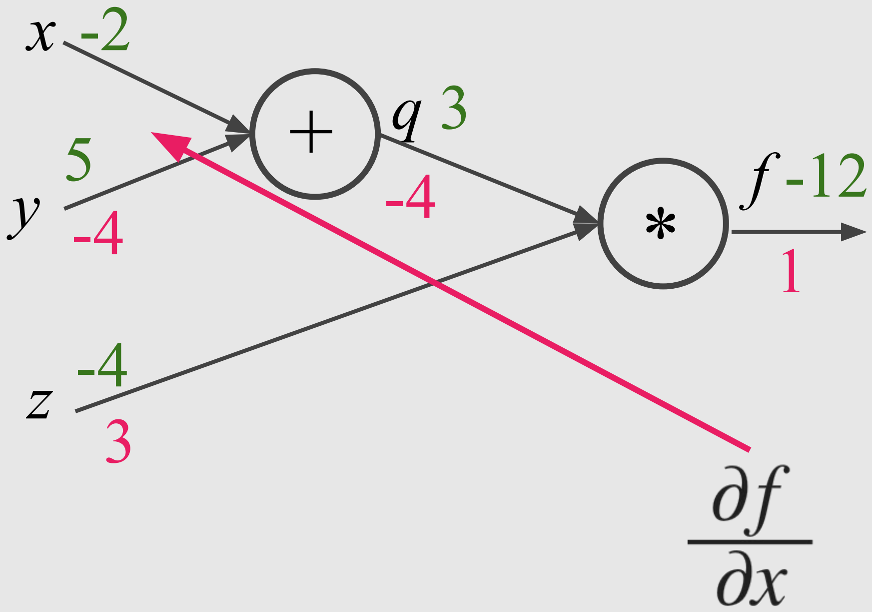
$$f(x, y, z) = (x + y)z$$

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

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

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

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



Backpropagation: A Simple Example

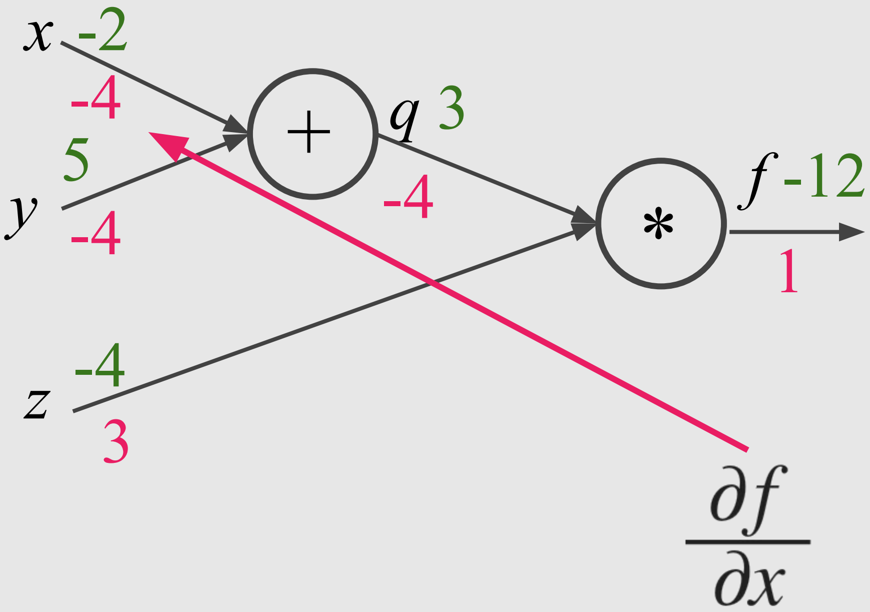
$$f(x, y, z) = (x + y)z$$

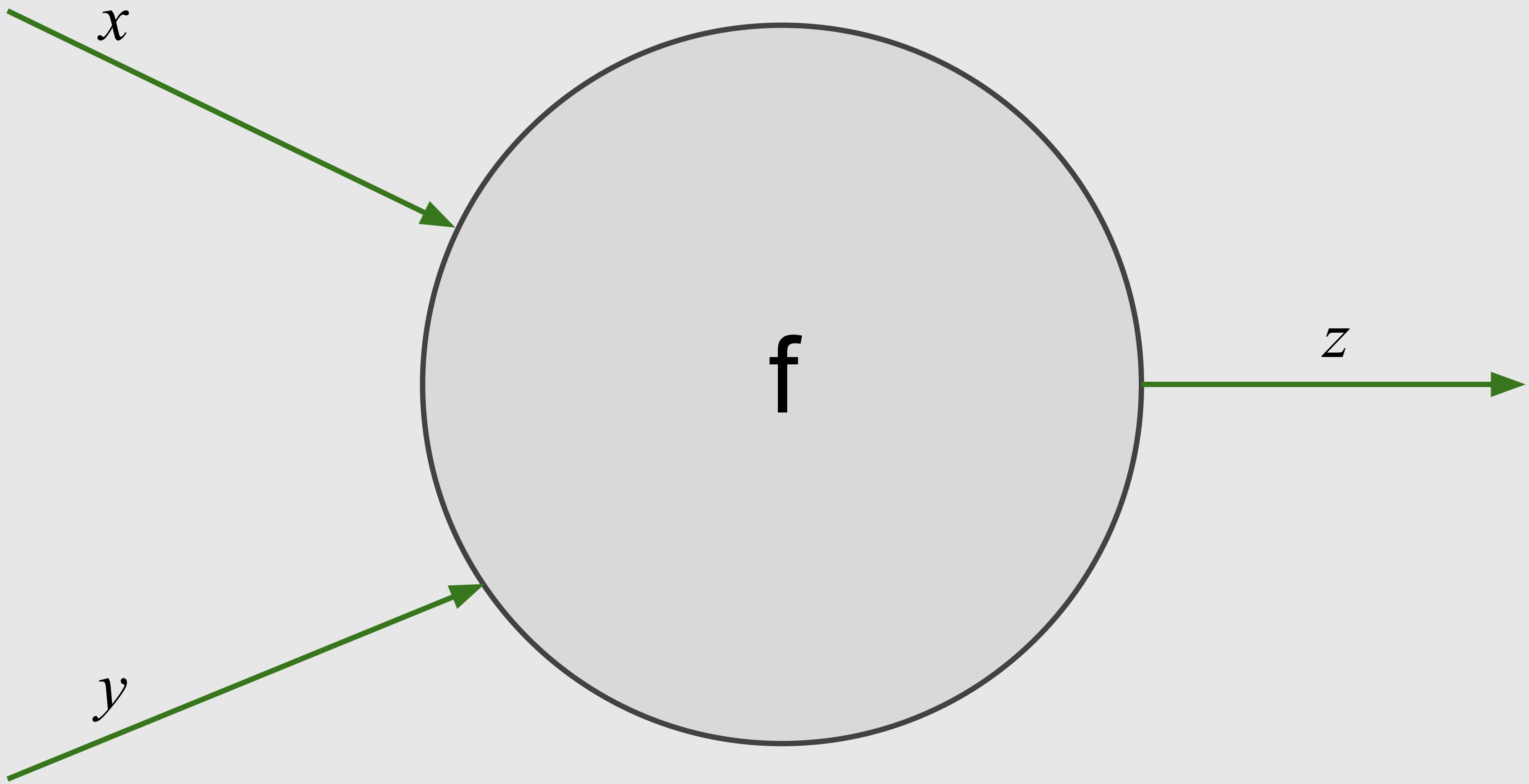
e.g., $x = -2$, $y = 5$, $z = -4$

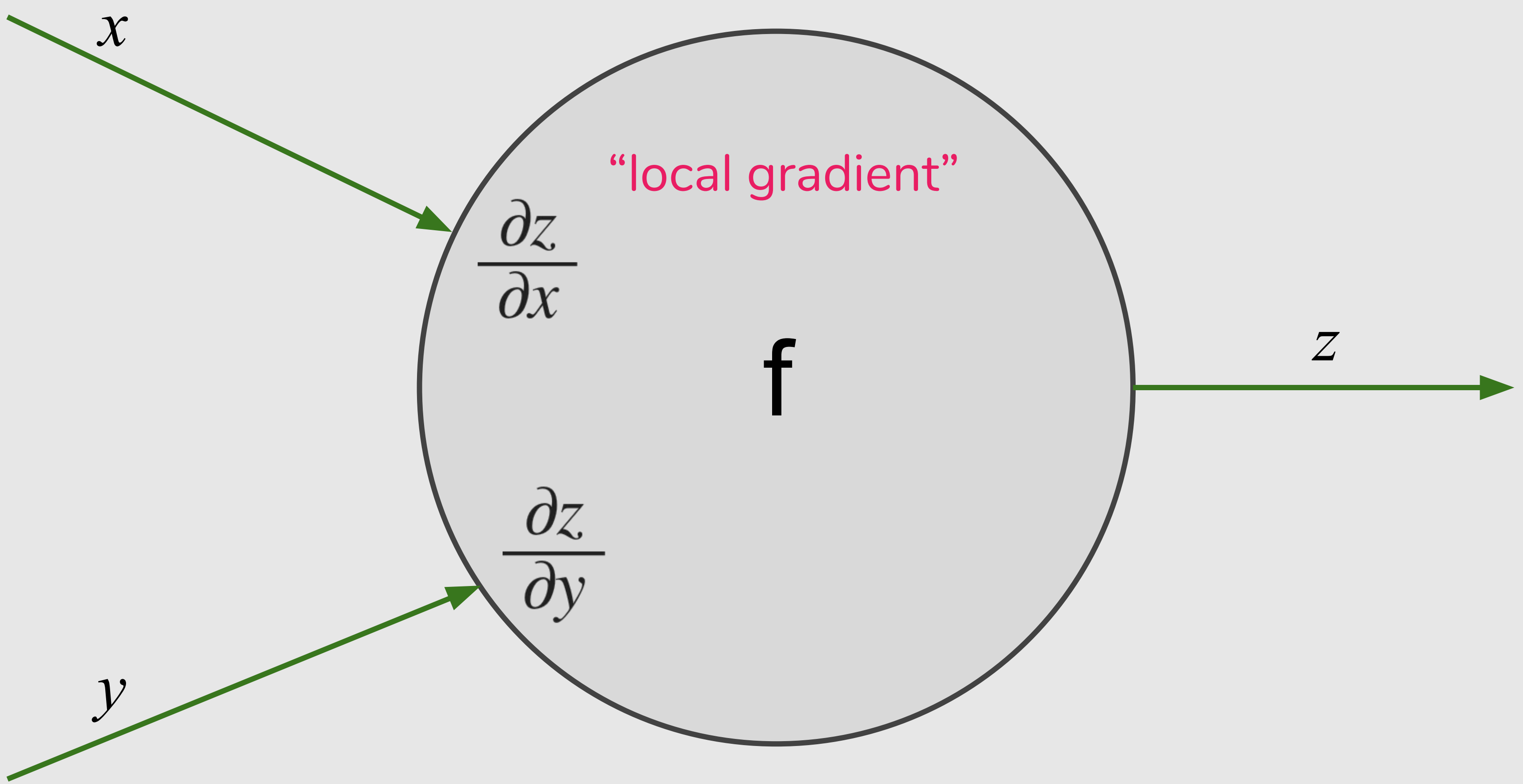
$$q = x + y \qquad \frac{\partial q}{\partial x} = 1 \qquad \frac{\partial q}{\partial y} = 1$$

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

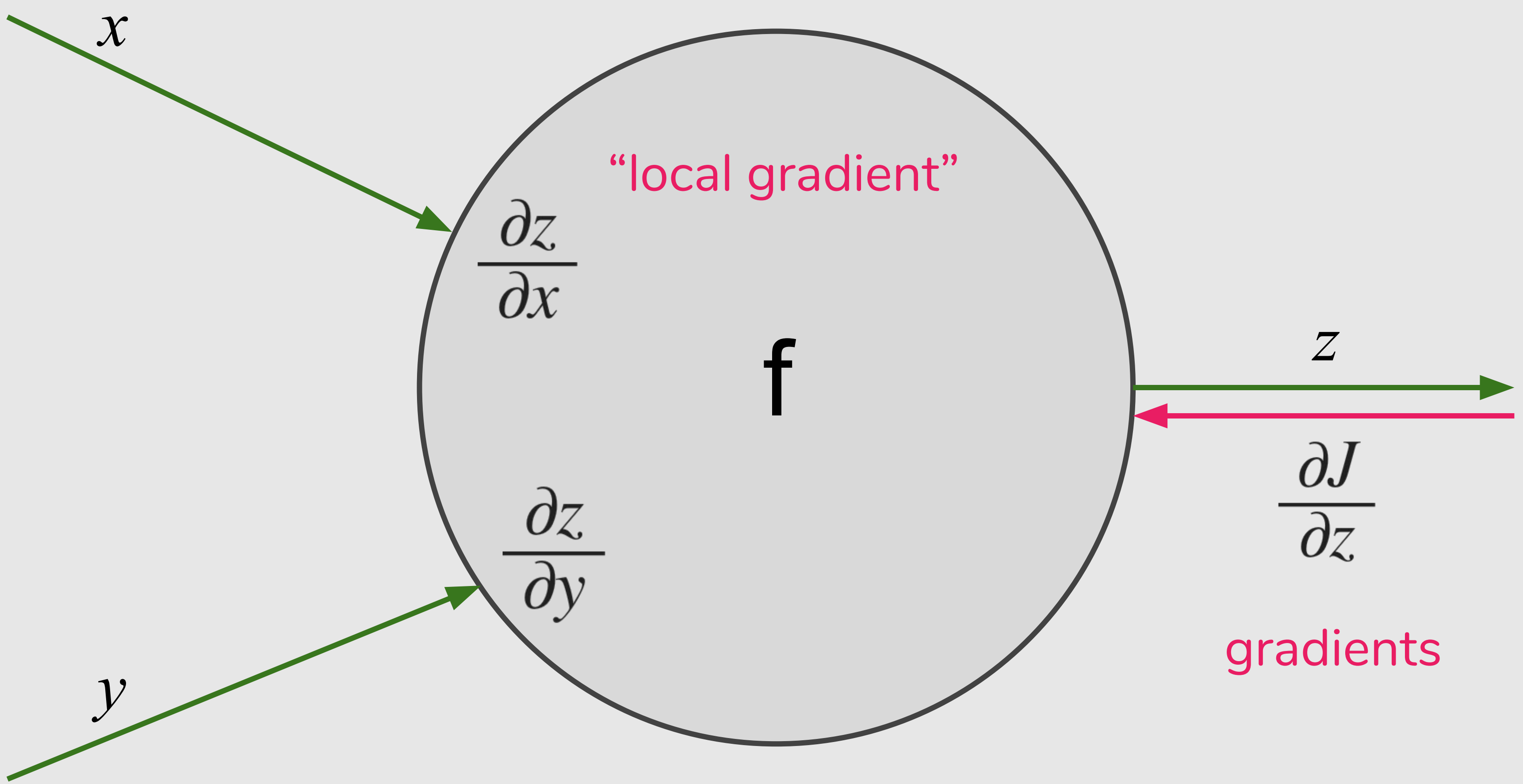
Want: $\dfrac{\partial f}{\partial x}, \dfrac{\partial f}{\partial y}, \dfrac{\partial f}{\partial z}$



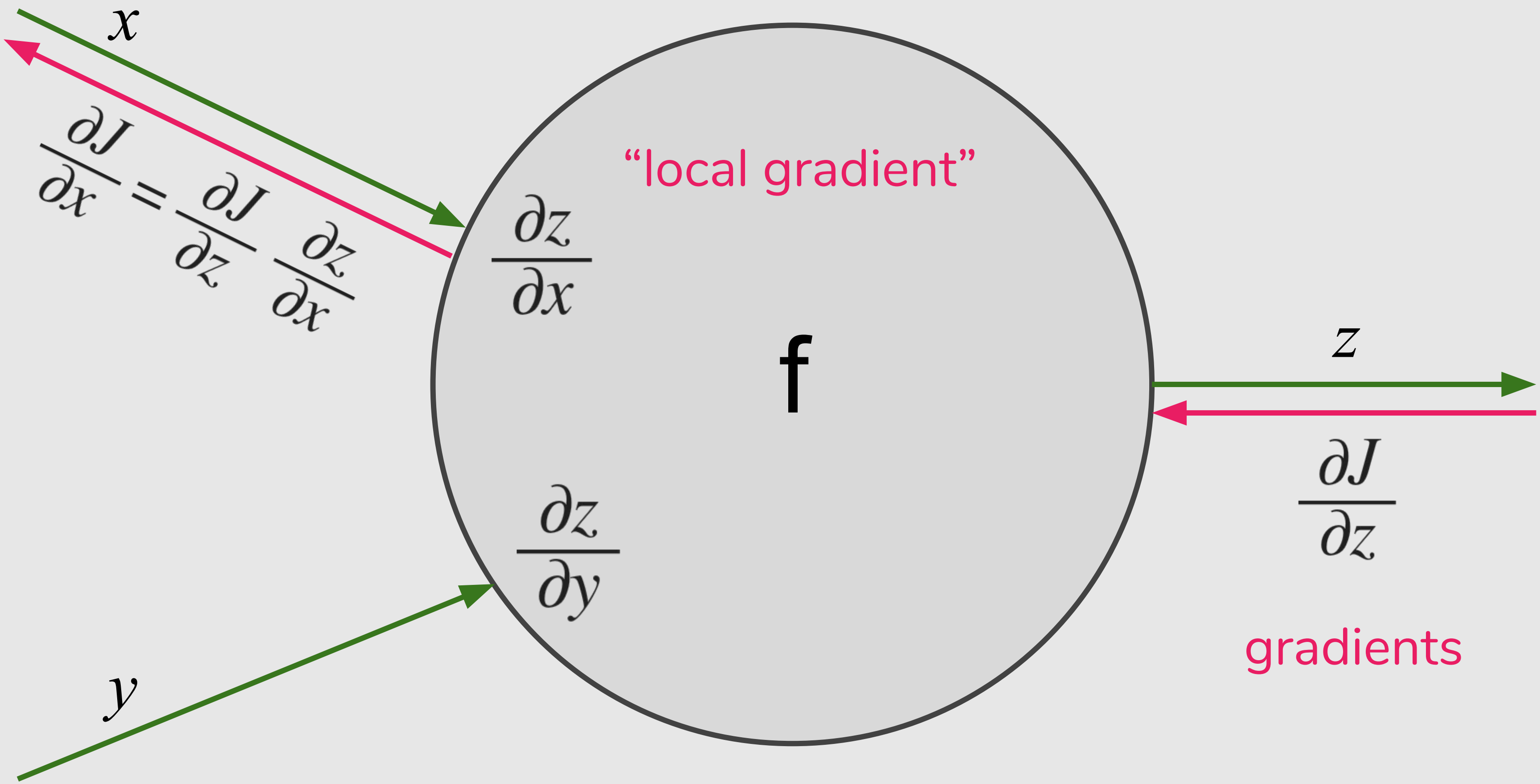

x
y
f
z



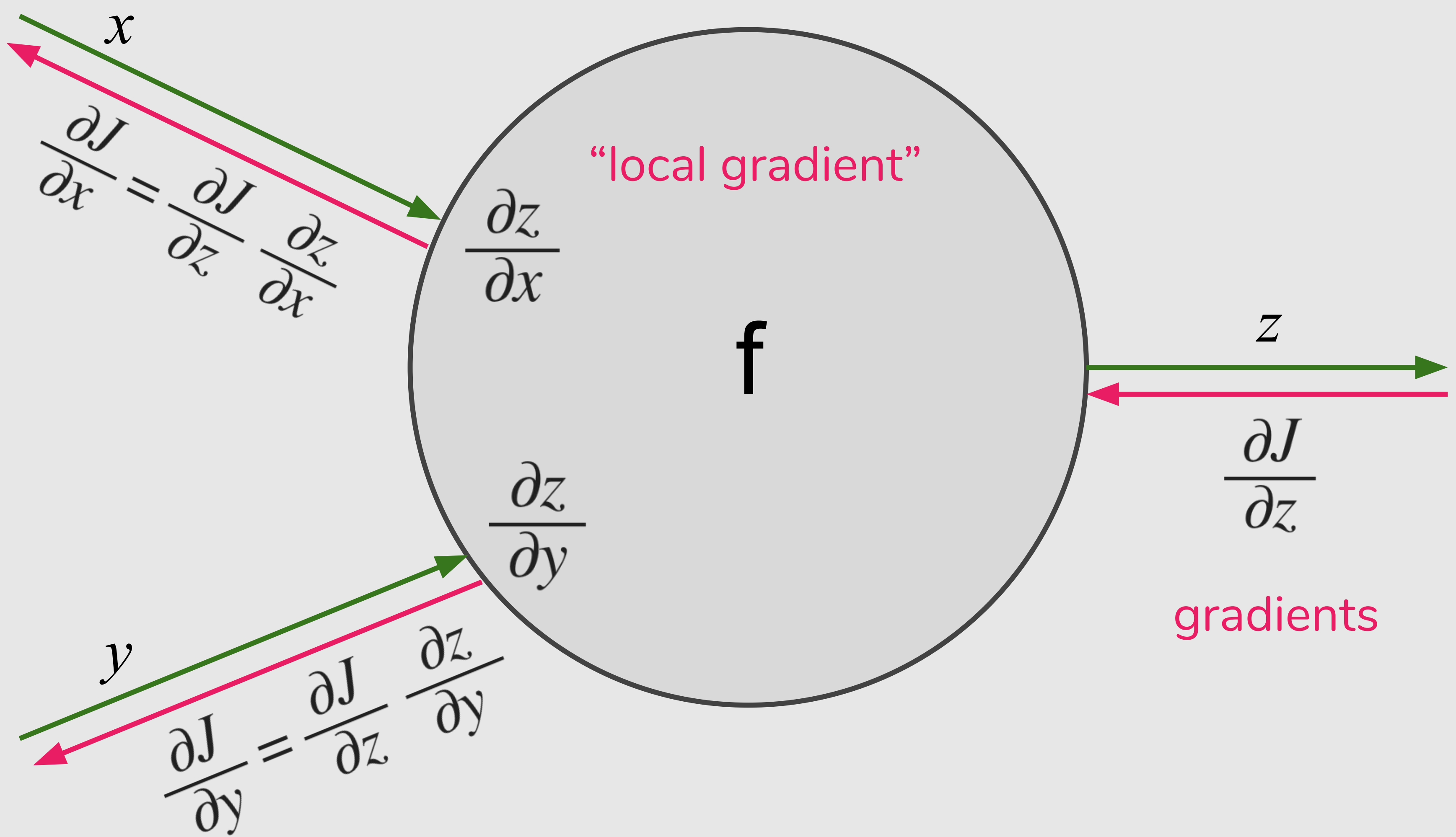
x
y
"local gradient"
$\frac{\partial z}{\partial x}$
$\frac{\partial z}{\partial y}$
f
z

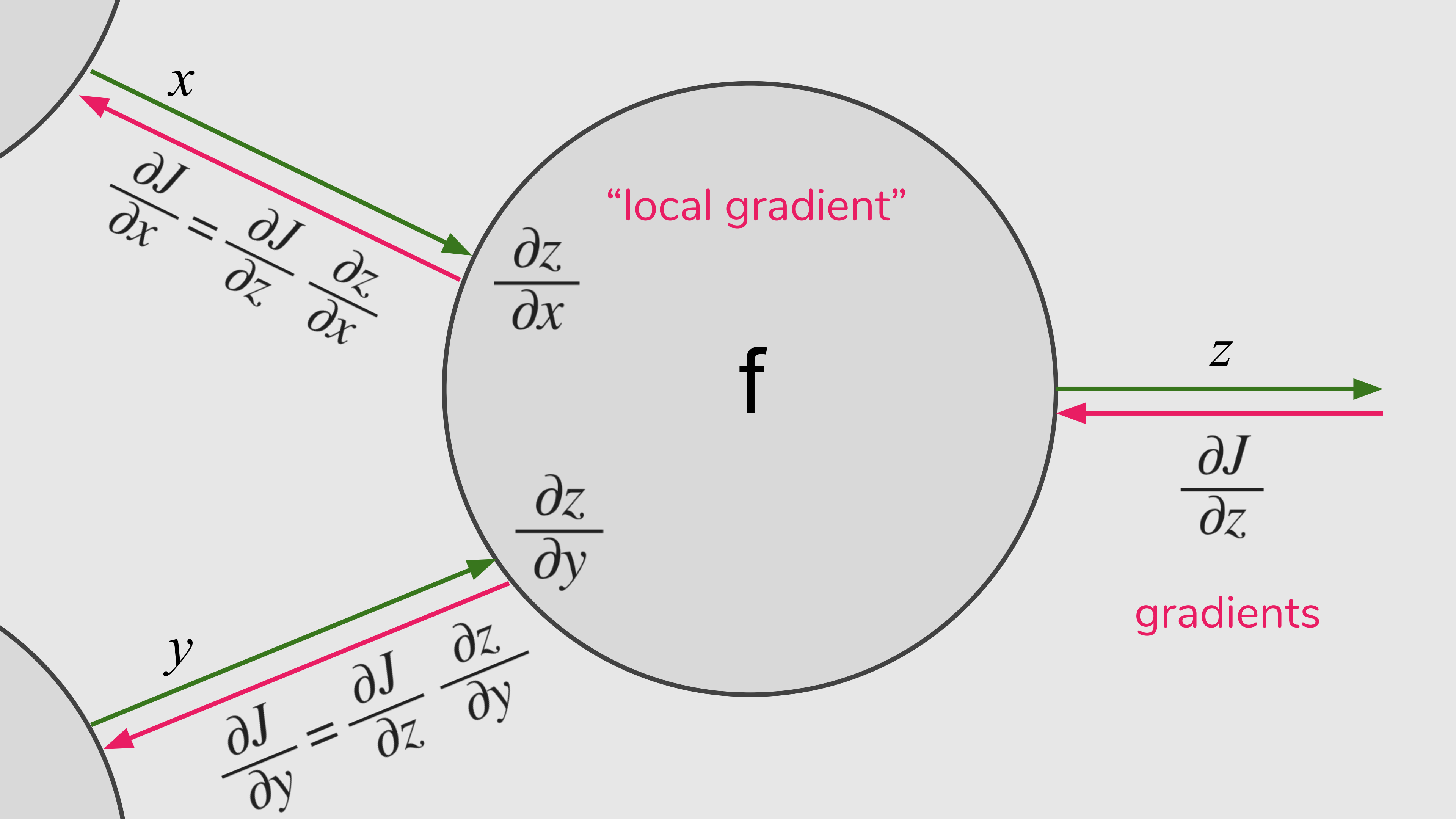


x
y
“local gradient”
$\frac{\partial z}{\partial x}$
$\frac{\partial z}{\partial y}$
f
z
$\frac{\partial J}{\partial z}$
gradients



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
$\frac{\partial J}{\partial z}$
$\frac{\partial z}{\partial y}$
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
$\frac{\partial J}{\partial z}$
$\frac{\partial z}{\partial y}$
y
$\frac{\partial J}{\partial y} = \frac{\partial J}{\partial z}\frac{\partial z}{\partial y}$
gradients

To be continued ...

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