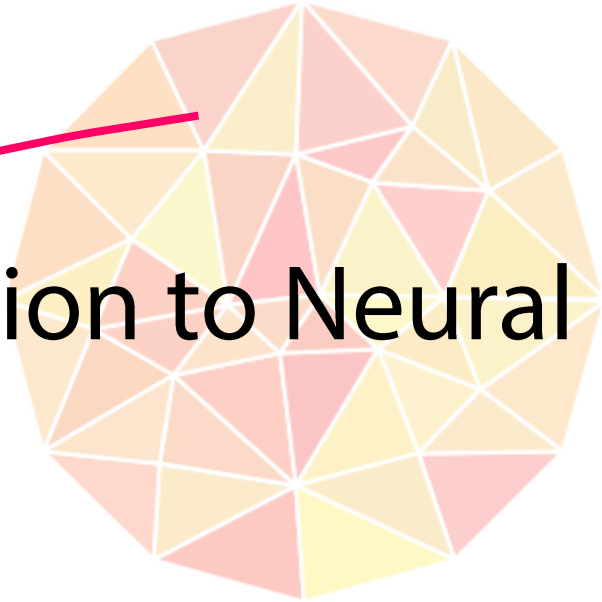
A Quick and Dirty



Introduction to Neural Networks

Stephen Baek

Motivation

- Typical machine learning tasks:

$$\min_{\boldsymbol{\theta}} \sum_i \left\| y^{(i)} - f\left(\mathbf{x}^{(i)} | \boldsymbol{\theta}\right) \right\|^2$$

  - where…
    - $\left(\mathbf{x}^{(i)}, y^{(i)}\right)$: i-th data point in a dataset.
    - $f(\cdot\,|\boldsymbol{\theta})$: machine learning model with parameters $\boldsymbol{\theta}$.

- Examples:
  - x=[particle size, void fraction, porosity, fluid viscosity, …], y=pearmeability (regression)
  - x=drone image, y=crack/no crack (classification)

So, how do we find $\boldsymbol{\theta}$?

Linear Models

- $\min_{\boldsymbol{\theta}} \sum_i \left\| y^{(i)} - f\left(\mathbf{x}^{(i)} \middle| \boldsymbol{\theta}\right) \right\|^2$ where $f(\mathbf{x}|\boldsymbol{\theta}) = \theta_0 + \theta_1 x_1 + \cdots + \theta_{d-1} x_{d-1} = \mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}$
- Let $\mathbf{X} \coloneqq \left[\left(\mathbf{x}^{(i)}\right)^{\mathrm{T}}\right]$, and $\mathbf{y} \coloneqq \left[y^{(i)}\right]$, then:

$$\mathcal{L}(\boldsymbol{\theta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|^2$$

Linear Models

- Solution:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \|\mathbf{y}-\mathbf{X}\boldsymbol{\theta}\|^2 = (\mathbf{y}-\mathbf{X}\boldsymbol{\theta})^{\mathrm{T}}(\mathbf{y}-\mathbf{X}\boldsymbol{\theta}) \\ &= \left(\mathbf{y}^{\mathrm{T}}-\boldsymbol{\theta}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\right)(\mathbf{y}-\mathbf{X}\boldsymbol{\theta}) \\ &= \mathbf{y}^{\mathrm{T}}\mathbf{y}-\boldsymbol{\theta}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{y}-\mathbf{y}^{\mathrm{T}}\mathbf{X}\boldsymbol{\theta}+\boldsymbol{\theta}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\theta} \\ &= \mathbf{y}^{\mathrm{T}}\mathbf{y}-2\mathbf{y}^{\mathrm{T}}\mathbf{X}\boldsymbol{\theta}+\boldsymbol{\theta}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\theta}\end{aligned}$$

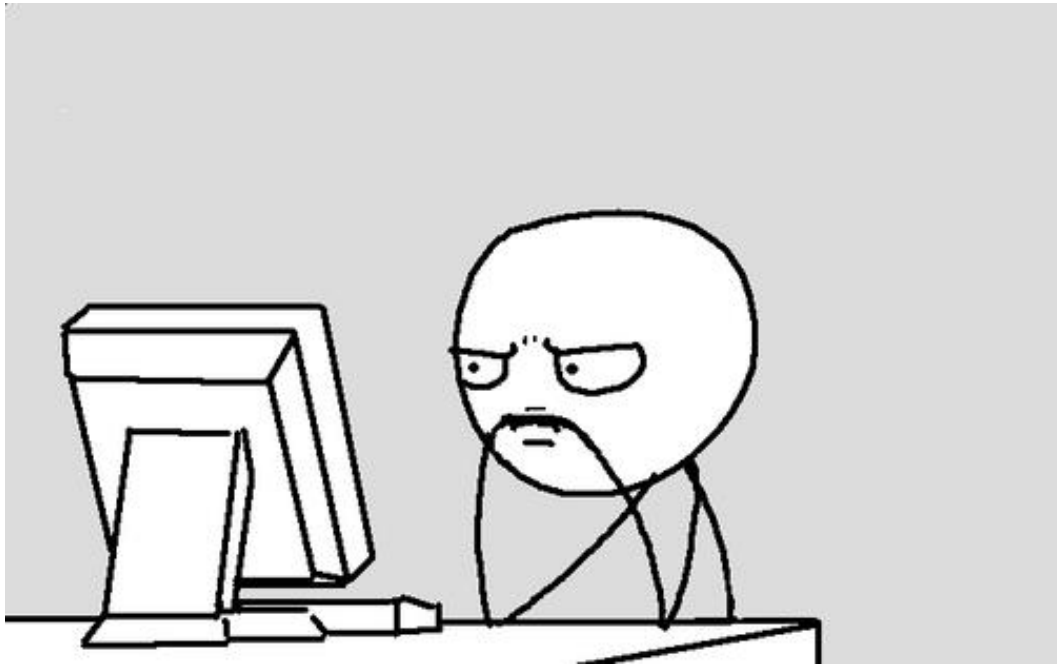
First order necessary condition: $\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}}(\boldsymbol{\theta}) = -2\mathbf{y}^{\mathrm{T}}\mathbf{X}+2\boldsymbol{\theta}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X} \equiv 0$

$$\begin{aligned}&\Leftrightarrow \boldsymbol{\theta}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X} = \mathbf{y}^{\mathrm{T}}\mathbf{X} \\ &\Leftrightarrow \mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\theta} = \mathbf{X}^{\mathrm{T}}\mathbf{y} \\ &\Leftrightarrow \boldsymbol{\theta} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}\end{aligned}$$

A bit of nonlinearity…

- Minimize $f(x) = (\cos x + \tan x)^2$, w.r.t. $x \in (-1,1)$
  - First order necessary condition:

$$\frac{\partial f}{\partial x} = 2(\cos x + \tan x)(-\sin x + \sec^2 x) \equiv 0$$

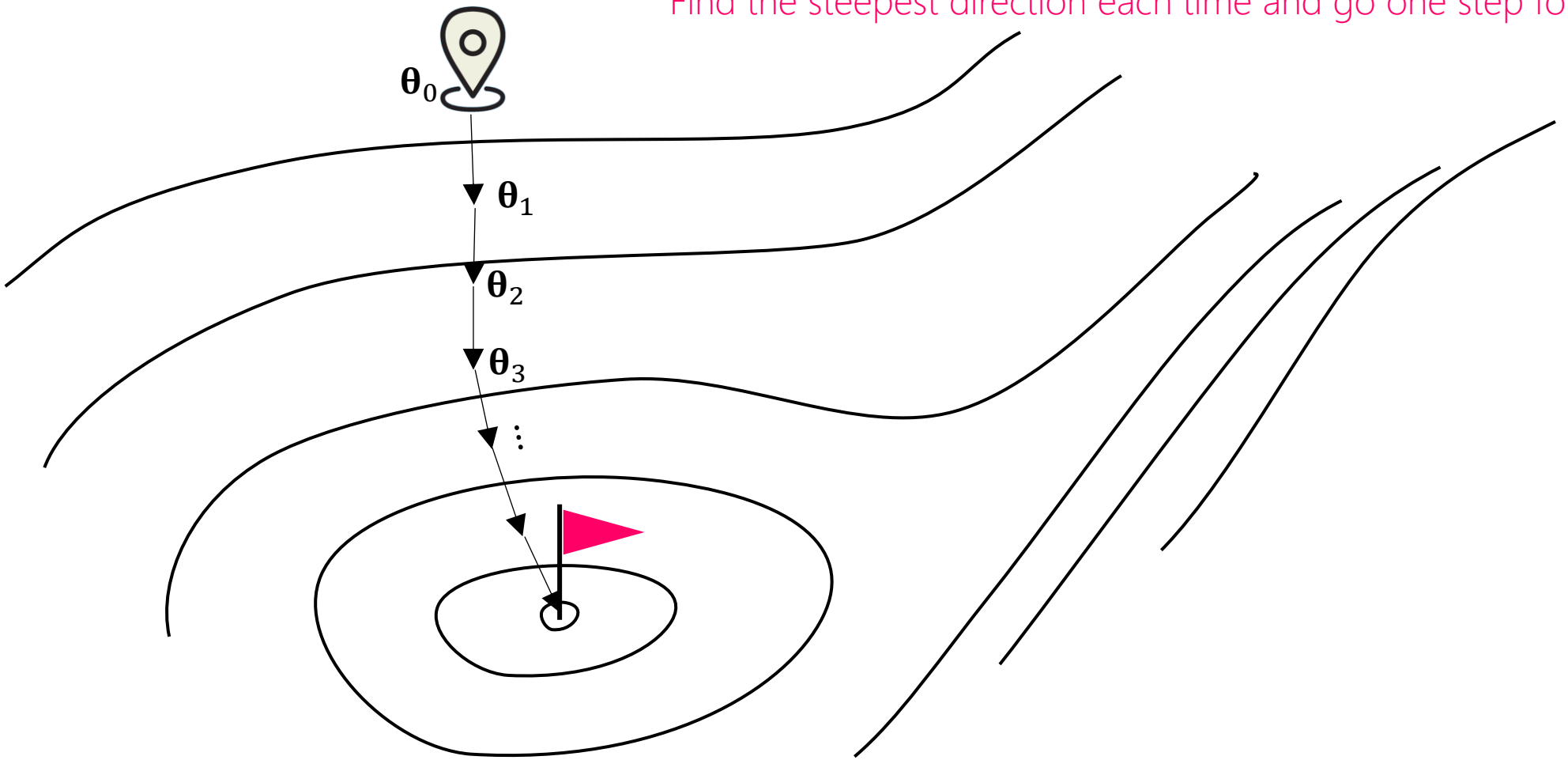


Climbing up a Mountain

- Q. Suppose you're an ***extremely*** near-sighted person (can only see things within, say 6 ft., of your periphery). What would be the best strategy to get to the peak?

A Strategy: Steepest Ascent

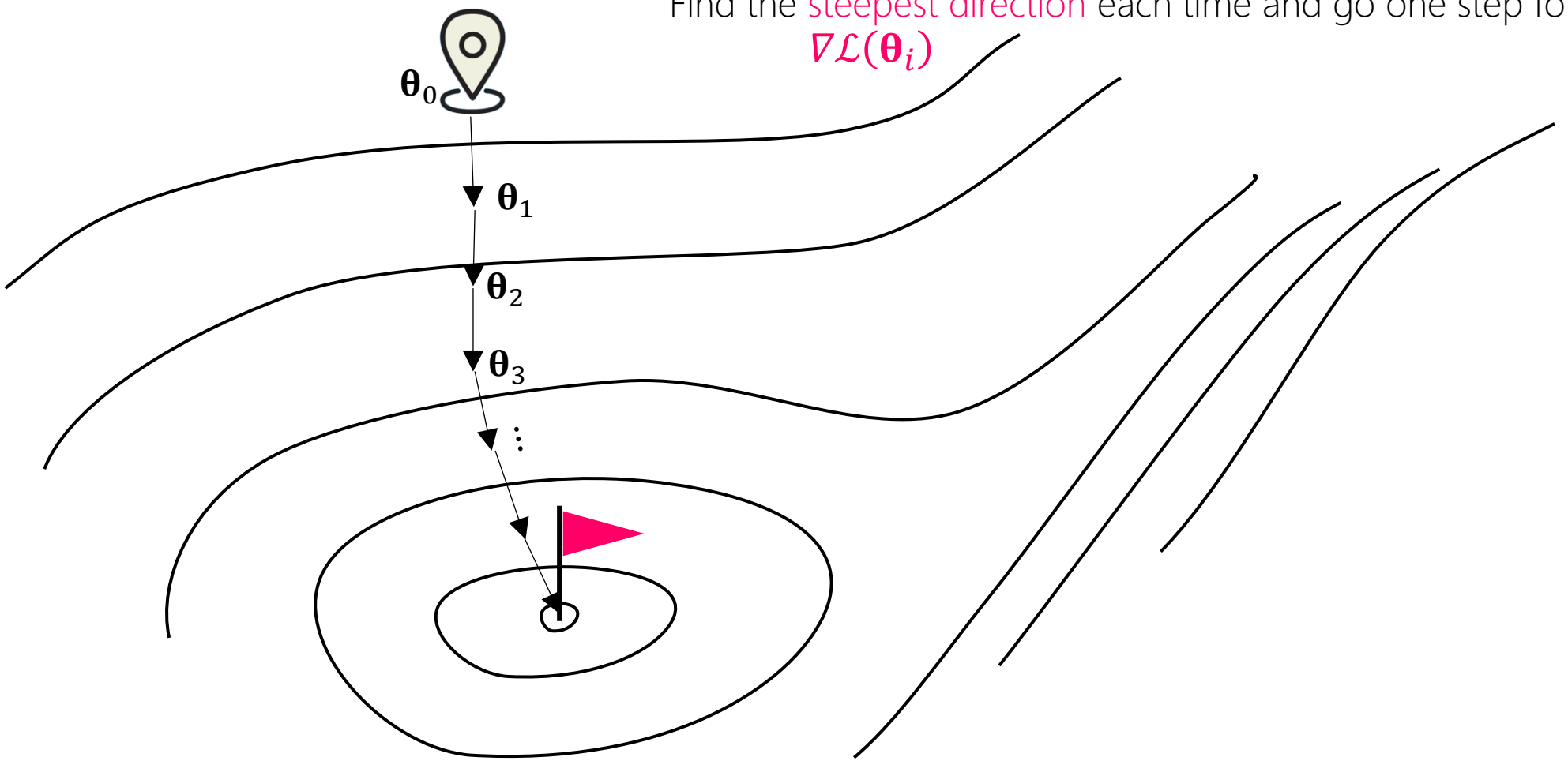
"Find the steepest direction each time and go one step forward."



A Strategy: Steepest Ascent

"Find the steepest direction each time and go one step forward."

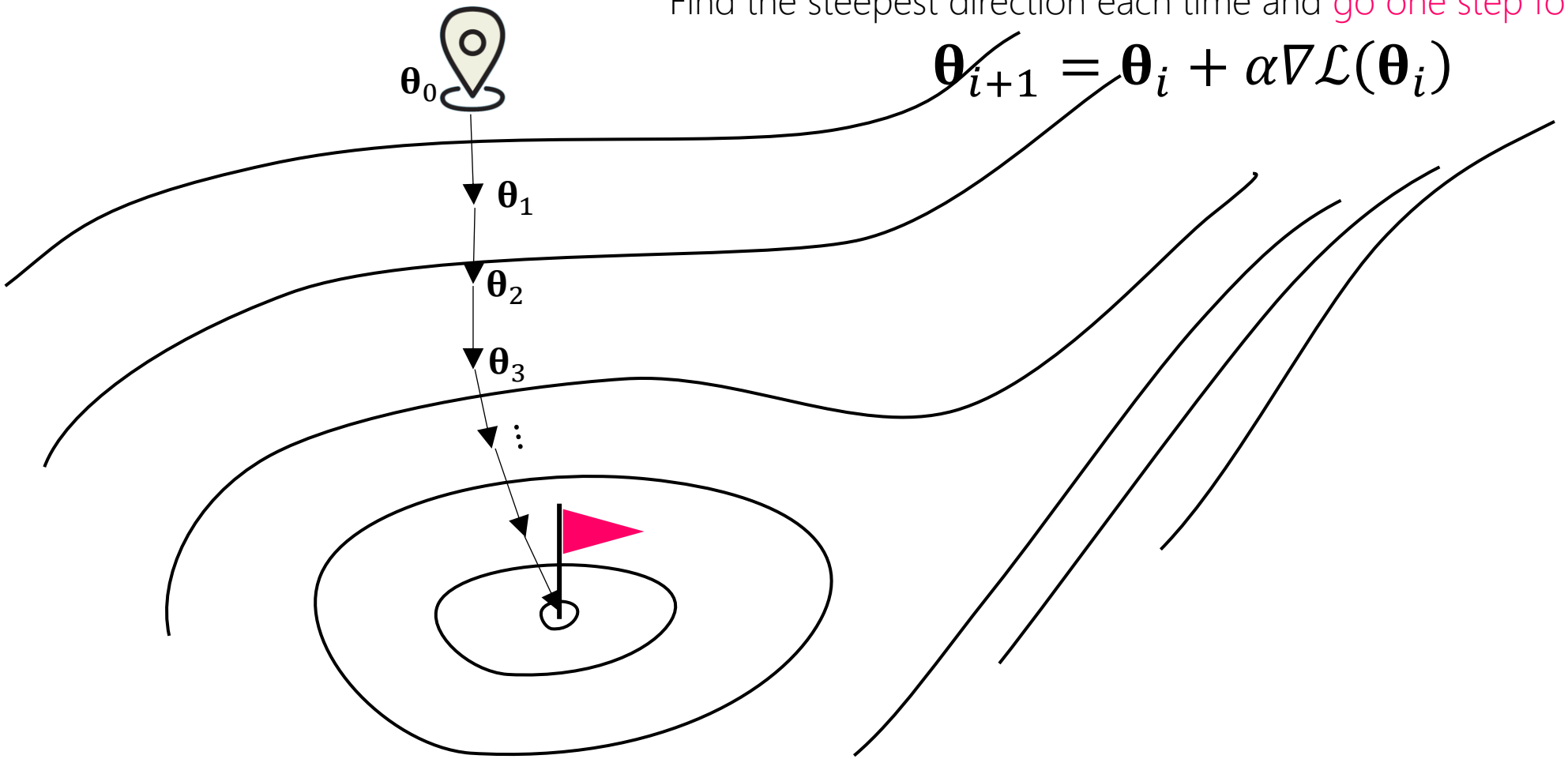
$\nabla \mathcal{L}(\boldsymbol{\theta}_i)$



A Strategy: Steepest Ascent

"Find the steepest direction each time and go one step forward."

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i + \alpha \nabla \mathcal{L}(\boldsymbol{\theta}_i)$$



Steepest Descent Algorithm (a.k.a. Gradient Descent)

- Given a differentiable function $\mathcal{L}$, the function value decreases the fastest if one goes in the direction of the negative gradient of $\mathcal{L}$.

- It follows that, for small enough scalar value α, if

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \alpha \nabla \mathcal{L}(\boldsymbol{\theta}_i)$$

  then $\mathcal{L}(\boldsymbol{\theta}_{i+1}) \leq \mathcal{L}(\boldsymbol{\theta}_i)$. (Proof: 1st order Taylor approximation)

Machine Learning with Gradient Descent

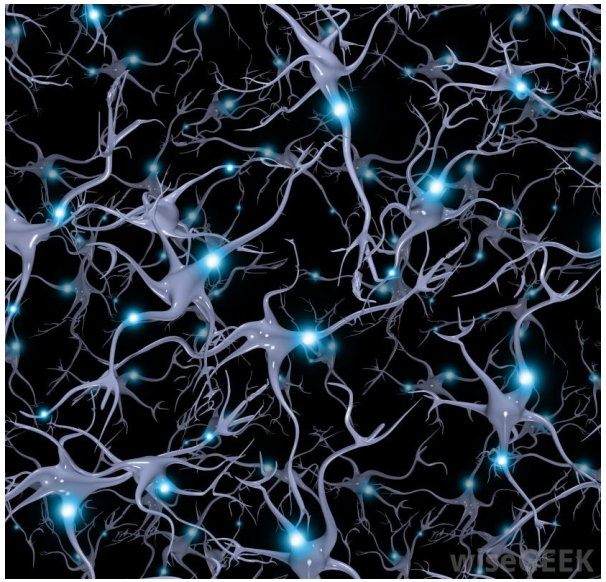
1. Design your model $f(\mathbf{x}|\boldsymbol{\theta})$, and the learning objective $\mathcal{L}(\boldsymbol{\theta}|\mathbf{x}, y)$.
2. Initialize the model parameters $\boldsymbol{\theta}$ (usually with random numbers).
3. Evaluate the gradient $\nabla\mathcal{L}$ using the current model parameters $\boldsymbol{\theta}$ and training dataset $\{\mathbf{x}^{(i)}, y^{(i)}\}$.
4. Improve the model parameters with a given learning rate α and the update strategy: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\nabla\mathcal{L}$.
5. Repeat 3~4 until converges

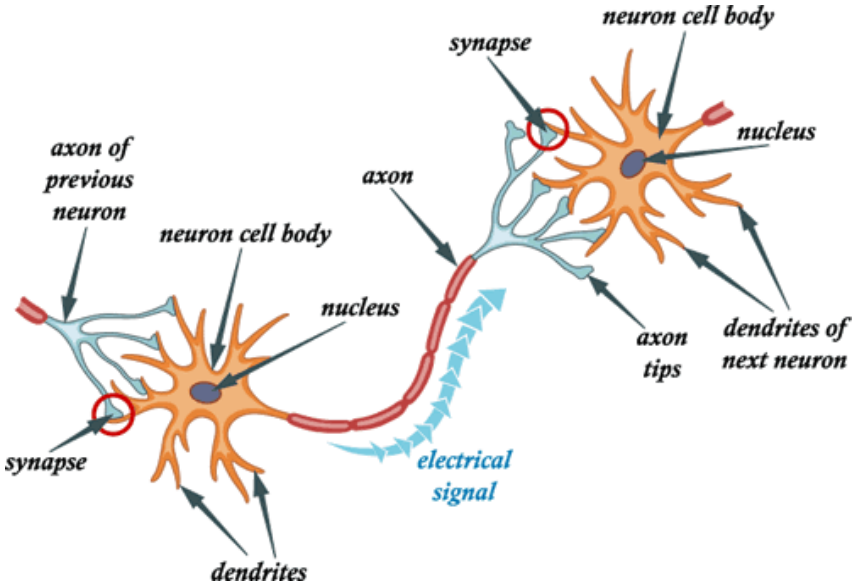
Machine Learning with Gradient Descent

1. Design your model $f(\mathbf{x}|\boldsymbol{\theta})$, and the learning objective $\mathcal{L}(\boldsymbol{\theta}|\mathbf{x}, y)$.
2. Initialize the model parameters $\boldsymbol{\theta}$ (usually with random numbers).
3. Evaluate the gradient $\nabla\mathcal{L}$ using the current model parameters $\boldsymbol{\theta}$ and training dataset $\{\mathbf{x}^{(i)}, y^{(i)}\}$.
4. Improve the model parameters with a given learning rate α and the update strategy: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\nabla\mathcal{L}$.
5. Repeat 3~4 until converges

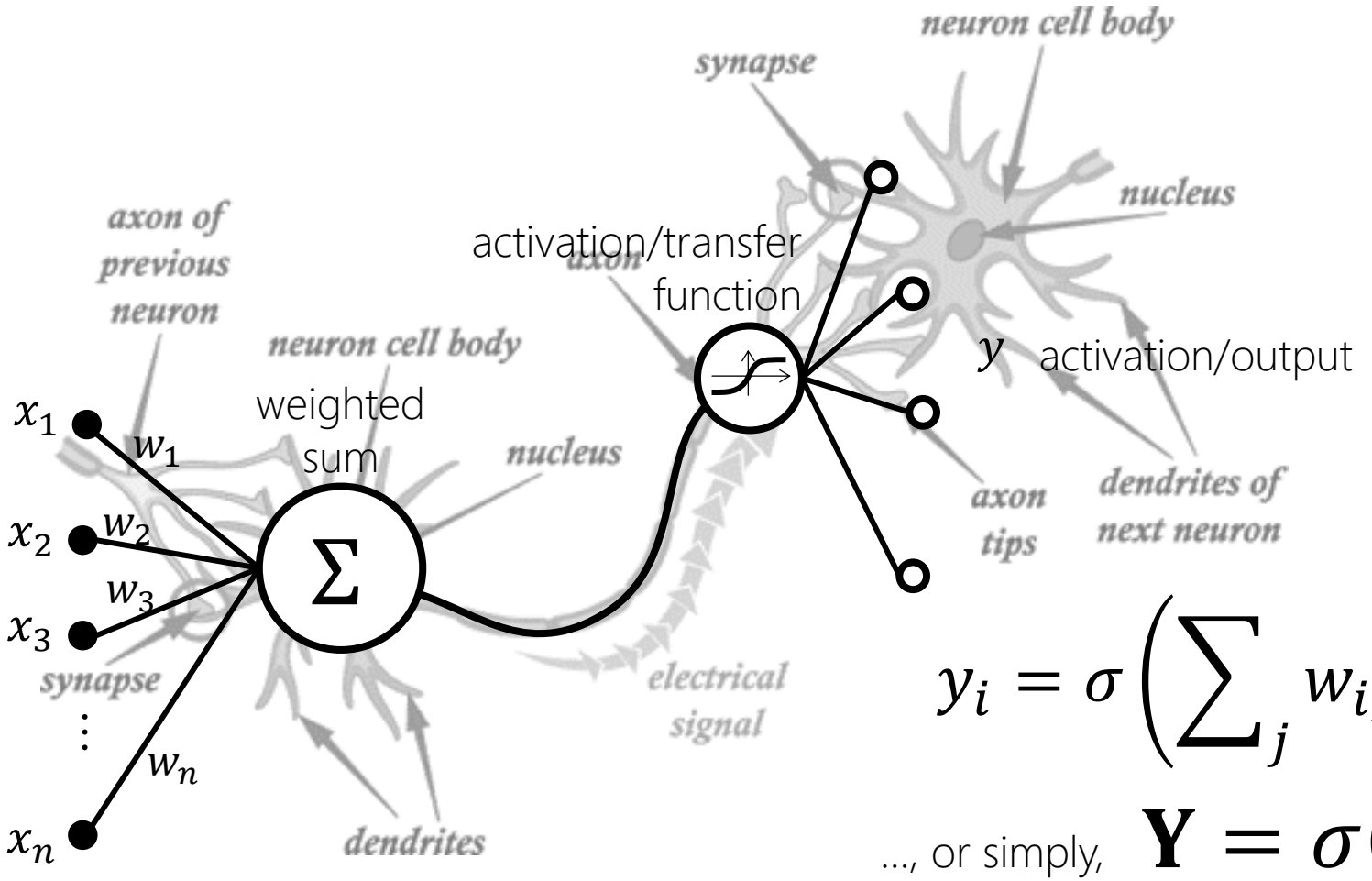
Neural Networks

Biological Neural Networks





Artificial Neural Networks (ANN)



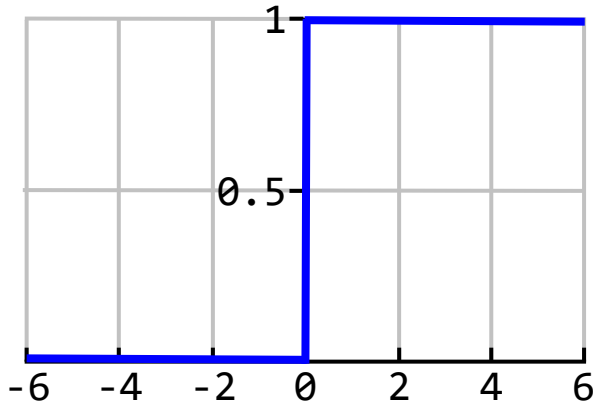
$$y_i = \sigma\left(\sum_j w_{i,j} x_j\right)$$

..., or simply, $\mathbf{Y} = \sigma(\mathbf{XW})$

Activation/Transfer Functions

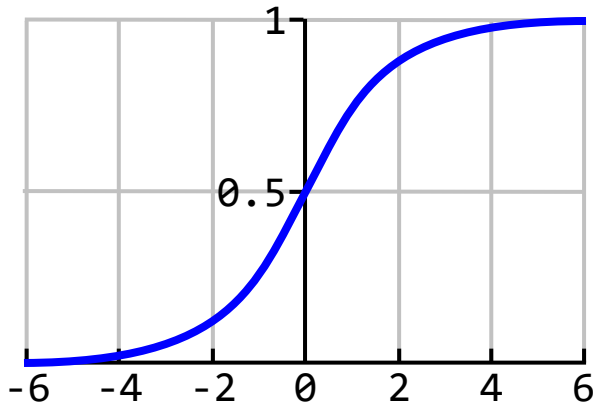
- "Threshold" in biological systems
- Step Function:

$$y = \sigma(z) = \begin{cases} 1 & \text{if } u \geq 0 \\ 0 & \text{otherwise} \end{cases}$$



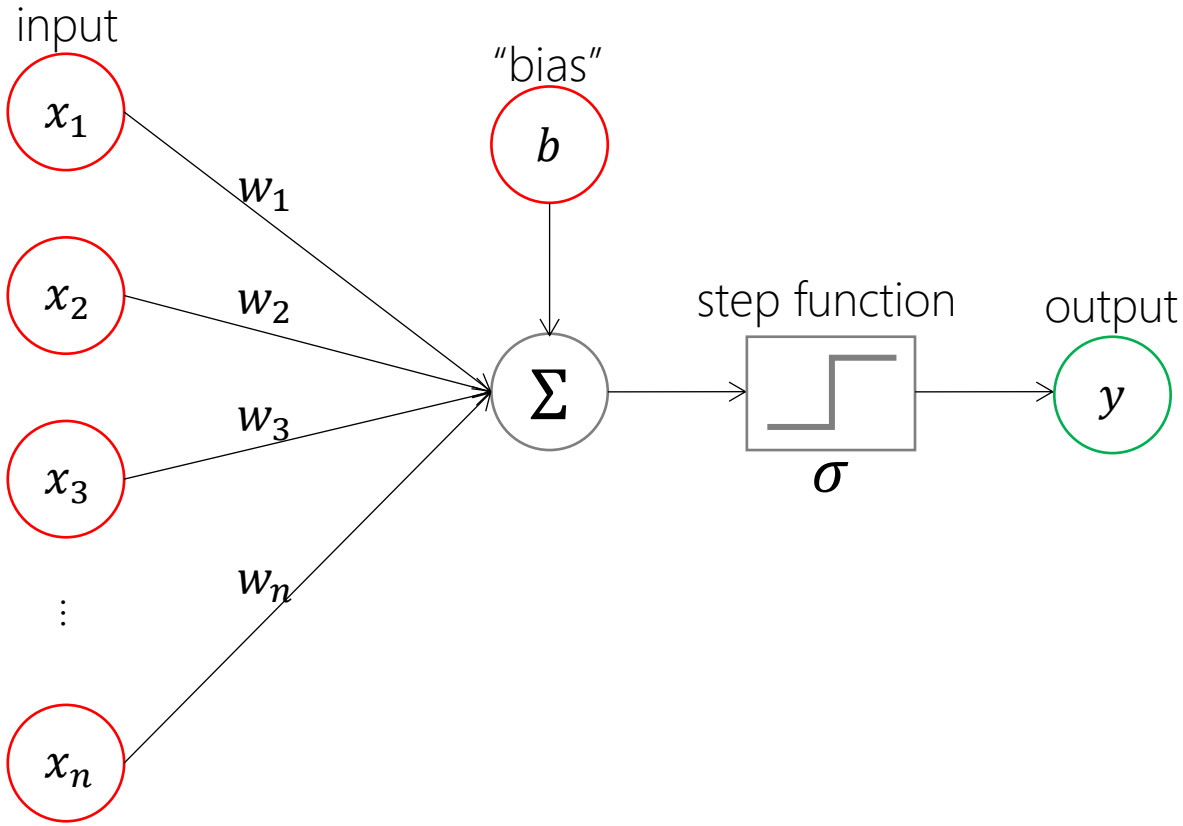
- Sigmoid Function:

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



The simplest model: the "Perceptron"

- Frank Rosenblatt (1957)



$$y = \sigma\left(\sum_j w_j x_j + b\right)$$

For each example $\left(x^{(t)}, \hat{y}^{(t)}\right)$, try to minimize:

$$\mathcal{L} = \frac{1}{2}\left(y^{(t)} - \hat{y}^{(t)}\right)^2$$

$$\frac{\partial \mathcal{L}}{\partial w_j} = \left(y^{(t)} - \hat{y}^{(t)}\right)\frac{\partial y^{(t)}}{\partial w_j}$$

$$= \left(y^{(t)} - \hat{y}^{(t)}\right)x_j^{(t)}$$

Weight update scheme:

$$w(t+1) = w(t) - r\left(y^{(t)} - \hat{y}^{(t)}\right)x_j^{(t)}$$

learning rate

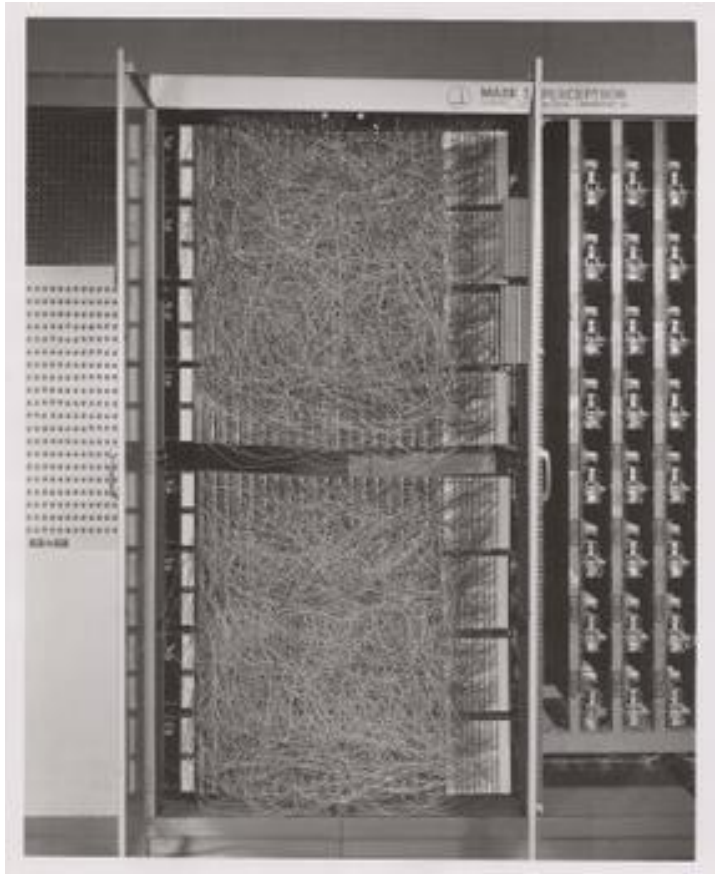
The simplest model: the “Perceptron”

- Frank Rosenblatt (1957)
  - The **Mark I Perceptron** machine was the first implementation of the perceptron algorithm.
  - Designed for image recognition.
  - Connected to a camera that used a 20x20 cadmium sulfide photocell array to produce a 400-pixel image.

http://www.rutherfordjournal.org/article040101.html

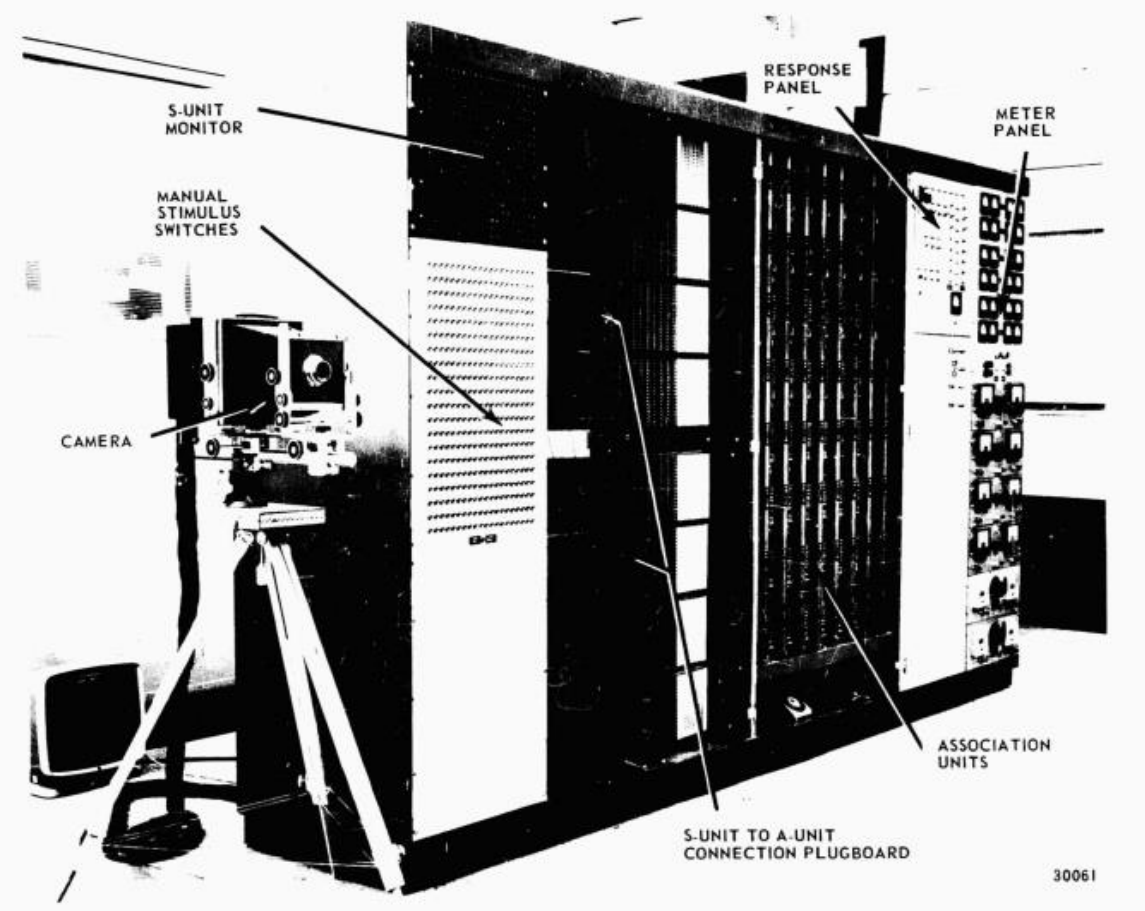
The simplest model: the “Perceptron”

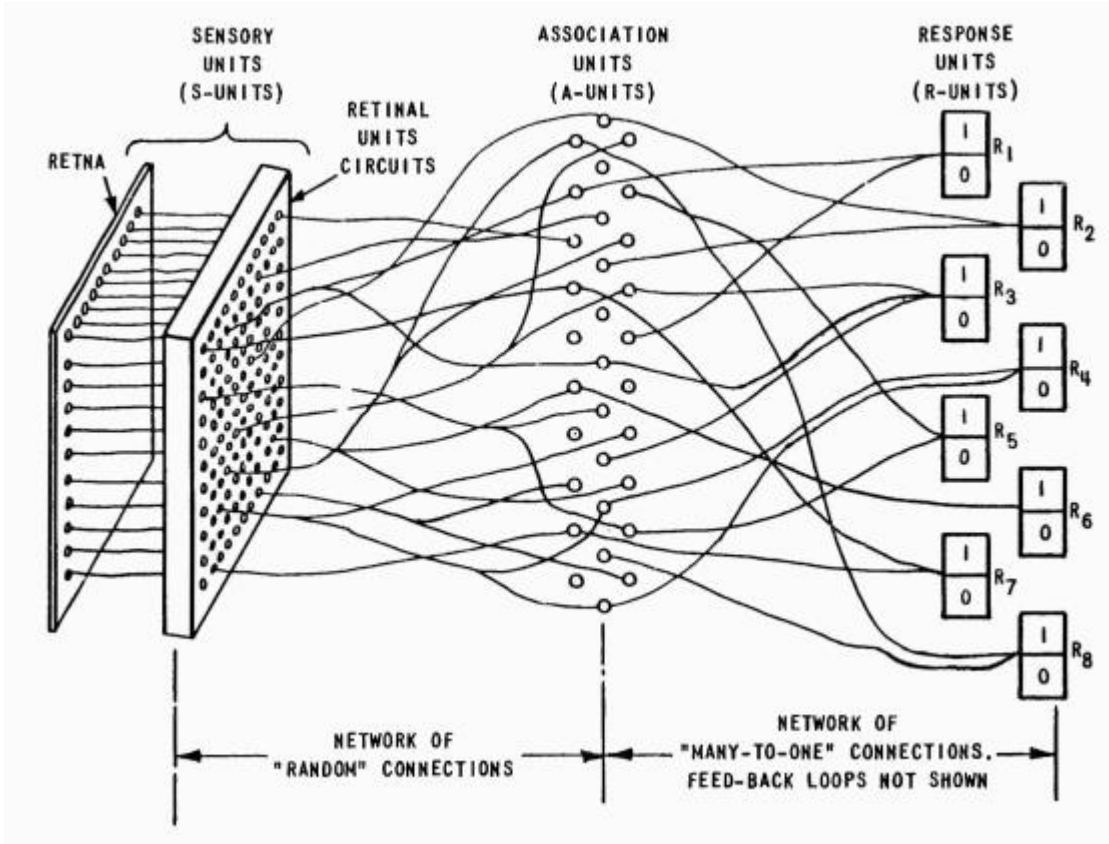
- Frank Rosenblatt (1957)
  - The **Mark I Perceptron** machine was the first implementation of the perceptron algorithm.
  - Designed for image recognition.
  - Connected to a camera that used a 20x20 cadmium sulfide photocell array to produce a 400-pixel image.
  - Weights were encoded in potentiometers.
  - Weight updates were performed by electric motors.



Wikipedia – Perceptron

The simplest model: the “Perceptron”





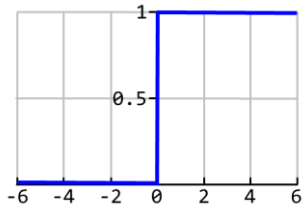
Mark 1 Perceptron Operators’ Manual (http://www.dtic.mil/dtic/tr/fulltext/u2/236965.pdf)

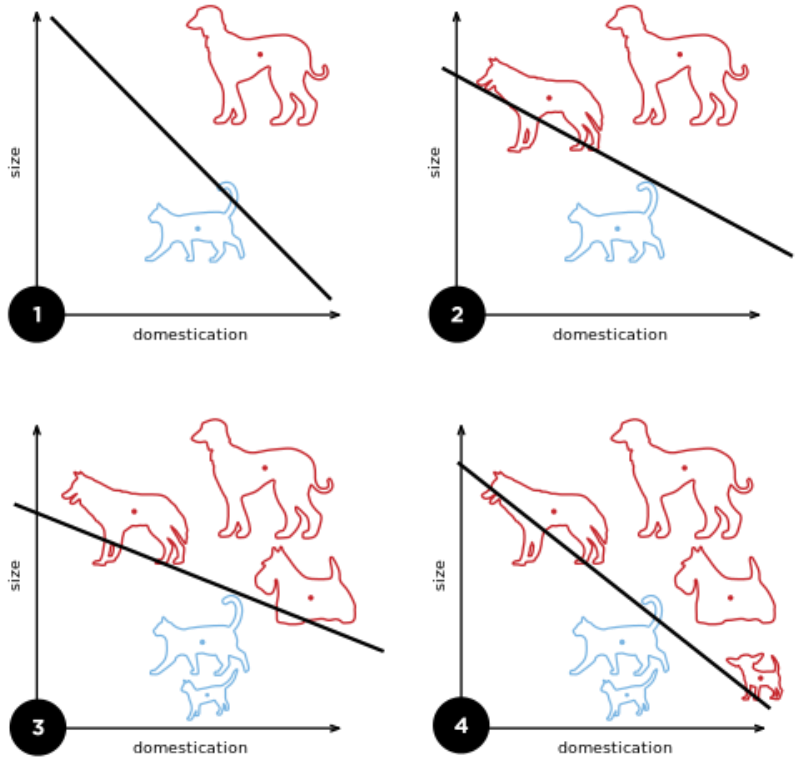
Perceptron as Linear Binary Classification

- Perceptron returns 1 iff. $\sum_j w_j x_j + b \geq 0$ and 0 otherwise.
- Essentially, this is equivalent to linear binary classification problem

$$z = \sum_j w_j x_j + b$$

$$y = \sigma(z) = \begin{cases} 1 & \text{if } u \geq 0 \\ 0 & \text{otherwise} \end{cases}$$





Wikipedia – Perceptron

The XOR Problem

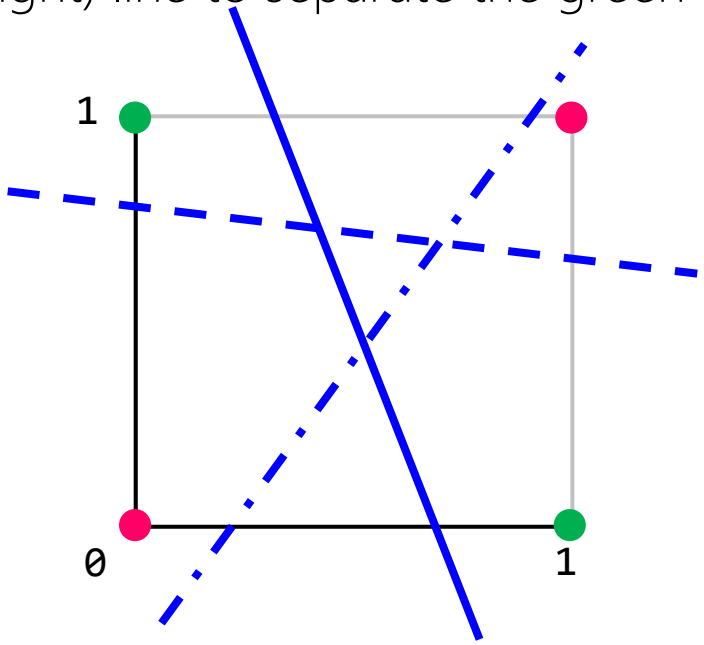
- XOR: Exclusive OR:

x_1	x_2	y
0	0	0
0	1	1
1	1	0
1	0	1

- Q. Can you implement the XOR gate with the perceptron model?

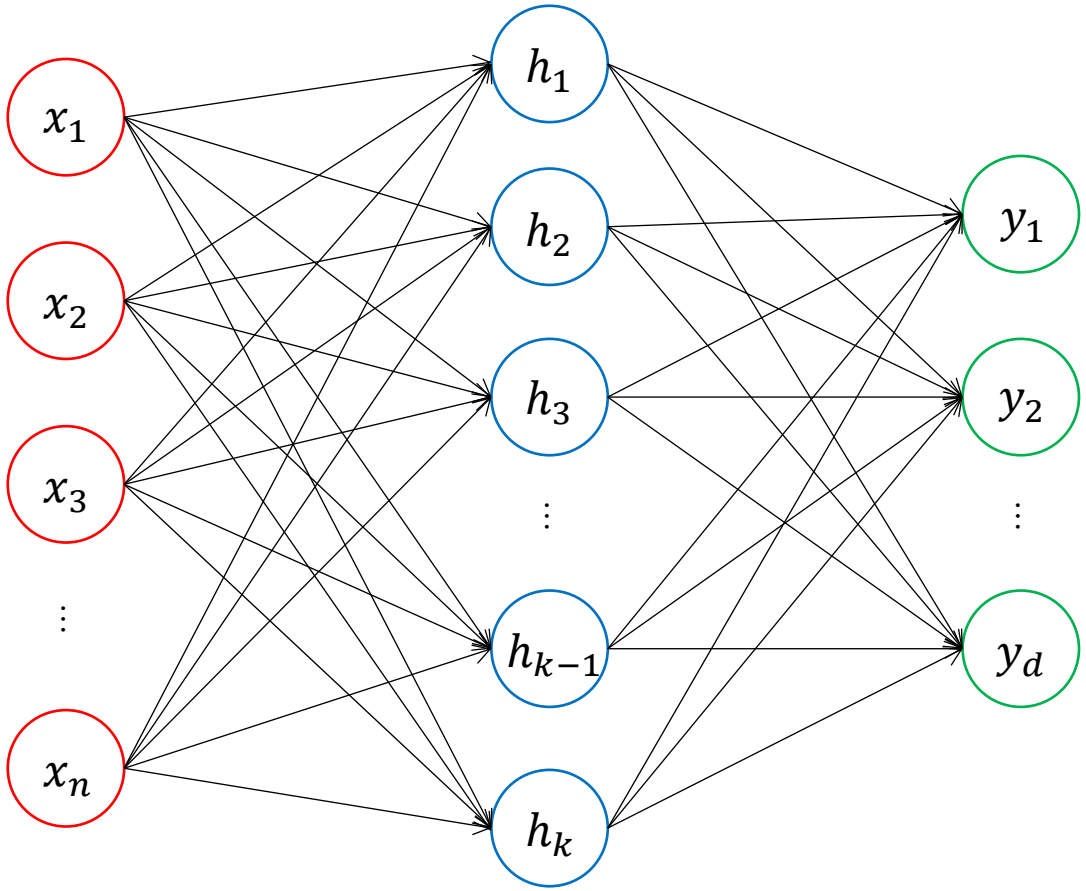
The XOR Problem

- The quick answer is NO.
- Why?
  - a perceptron is a linear classifier.
  - Can you draw a single (straight) line to separate the green and red dots?



Multi-Layer Perceptron (MLP)

- Minsky & Papert. (1969)

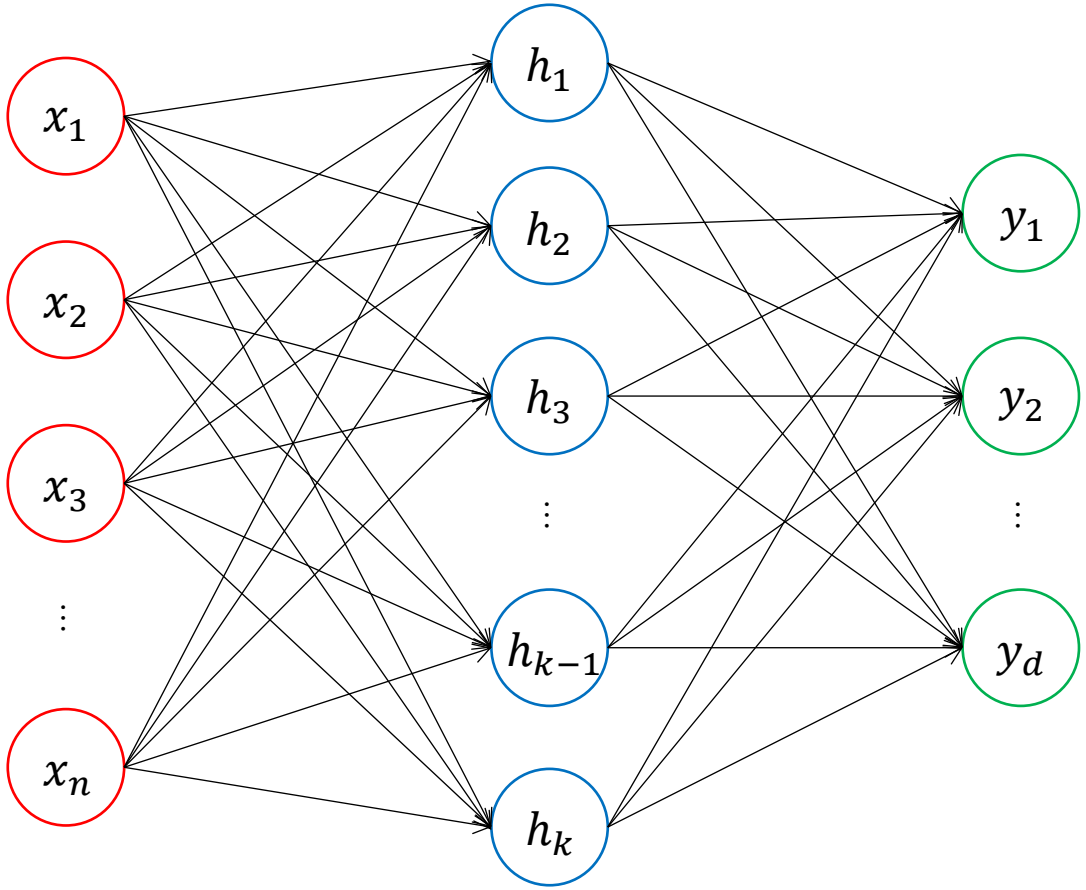


The perceptron has shown itself worthy of study despite (and even because of!) its severe limitations. It has many features to attract attention: its linearity; its intriguing learning theorem; its clear paradigmatic simplicity as a kind of parallel computation. There is no reason to suppose that any of these virtues carry over to the many-layered version. Nevertheless, we consider it to be an important research problem to elucidate (or reject) our intuitive judgement that the extension to multilayer systems is sterile.

Minsky & Papert (1969, pp. 231-232)

Multi-Layer Perceptron (MLP)

- Minsky & Papert. (1969)



$$h_i = \sigma_h\left(\sum_j w_{h,i,j}x_j + b_h\right) \quad y_k = \sigma_o\left(\sum_i w_{o,k,i}h_i + b_o\right)$$

$$\mathcal{L} = \frac{1}{2}\left\|\mathbf{y}^{(t)} - \hat{\mathbf{y}}^{(t)}\right\|^2$$

$$\frac{\partial \mathcal{L}}{\partial w} = ?$$

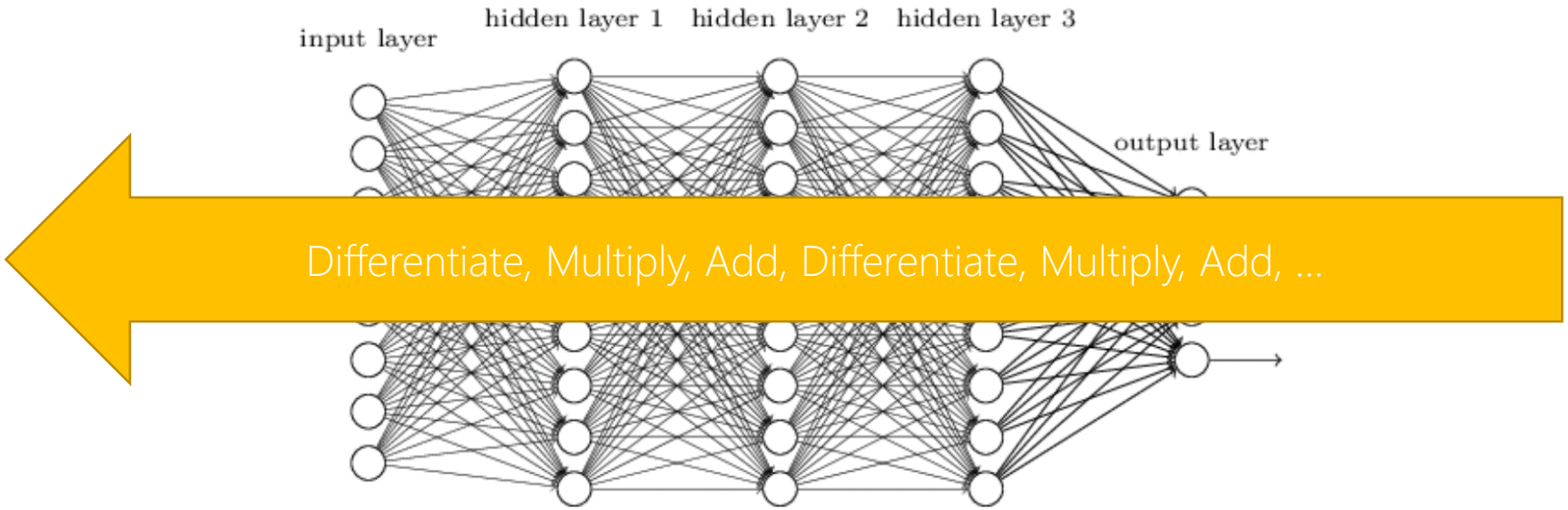
- M&P's criticism on (single-layer) perceptrons
- No practical way to train MLPs

→ The First AI Winter (1970s)


A FEW
MOMENTS LATER

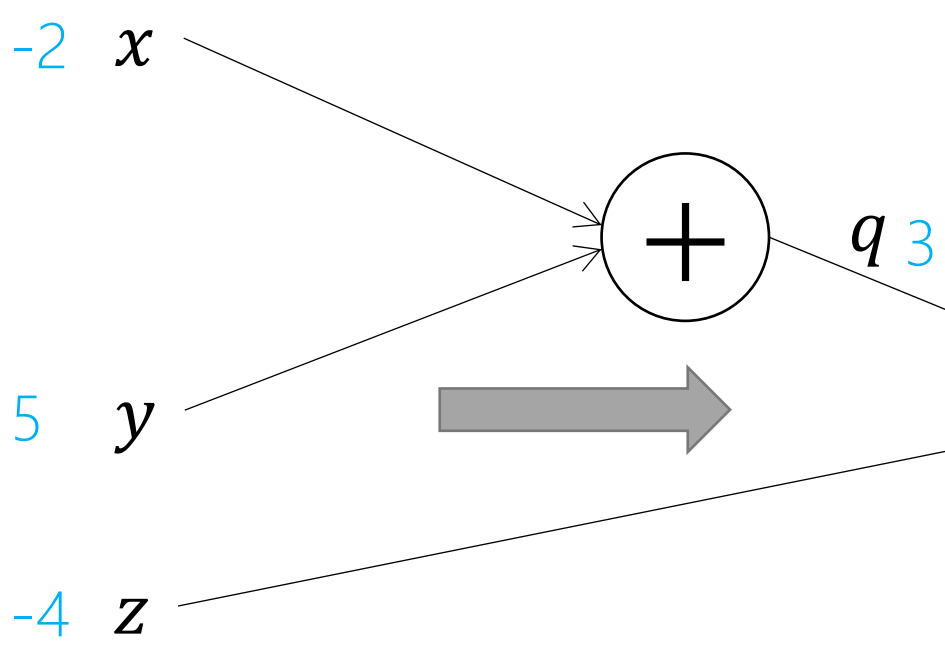
Backpropagation

- Rumelhart, Hinton, and Williams. (1986).
- A popular training method for neural nets
- Propagate what? “the gradient of the current error”



Backpropagation

- A simple example: $f(x, y, z) = (x + y)z$
  - Computational graph:



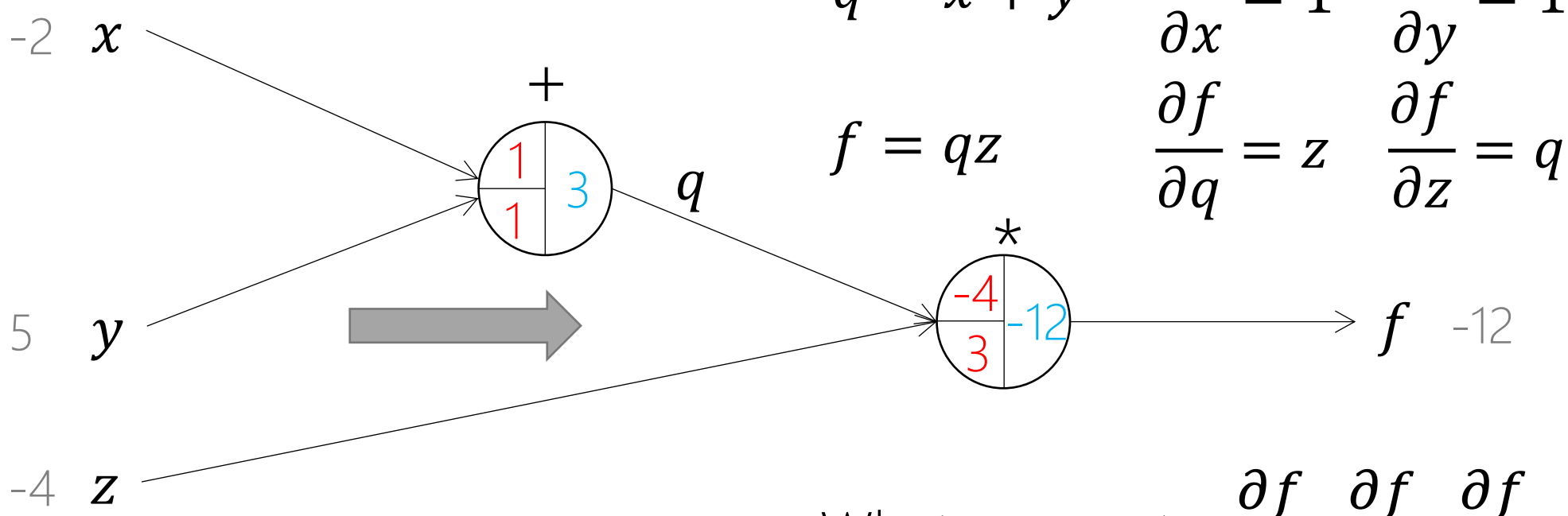
$$q = x + y \quad \frac{\partial q}{\partial x} = 1 \quad \frac{\partial q}{\partial y} = 1$$

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

What we want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

Backpropagation

- A simple example: $f(x, y, z) = (x + y)z$
  - Computational graph:



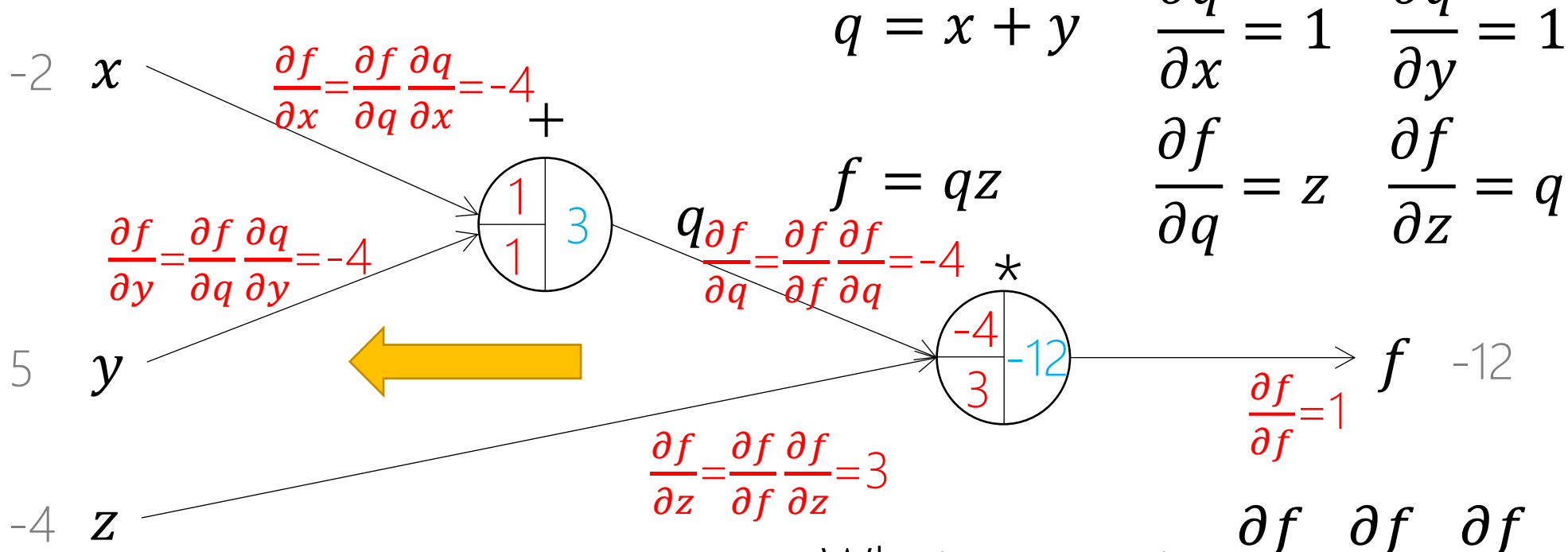
$q = x + y$ $\quad \frac{\partial q}{\partial x} = 1$ $\quad \frac{\partial q}{\partial y} = 1$

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

What we want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

Backpropagation

- A simple example: $f(x, y, z) = (x + y)z$
  - Computational graph:

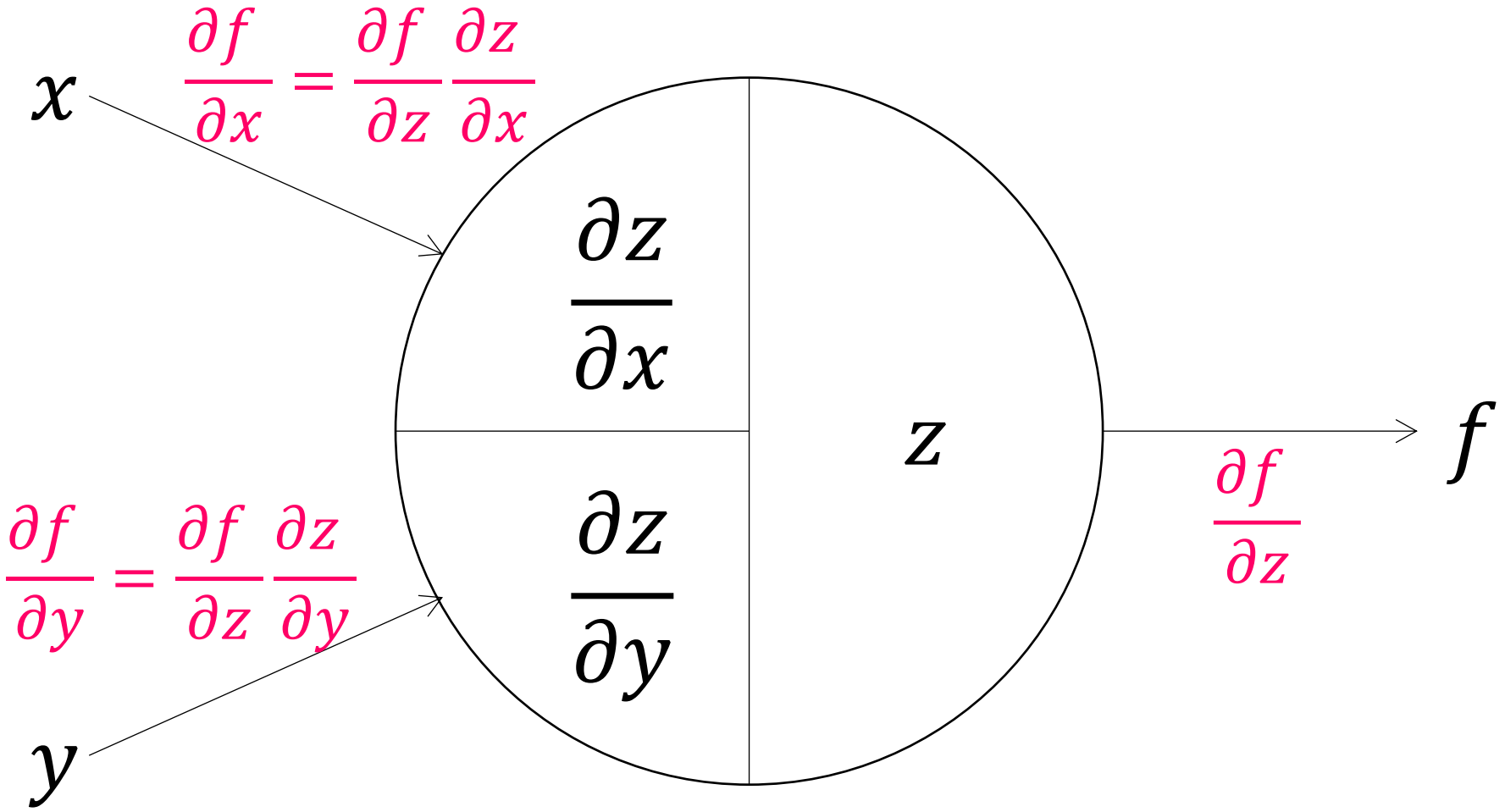


$q = x + y$ $\quad \frac{\partial q}{\partial x} = 1$ $\quad \frac{\partial q}{\partial y} = 1$

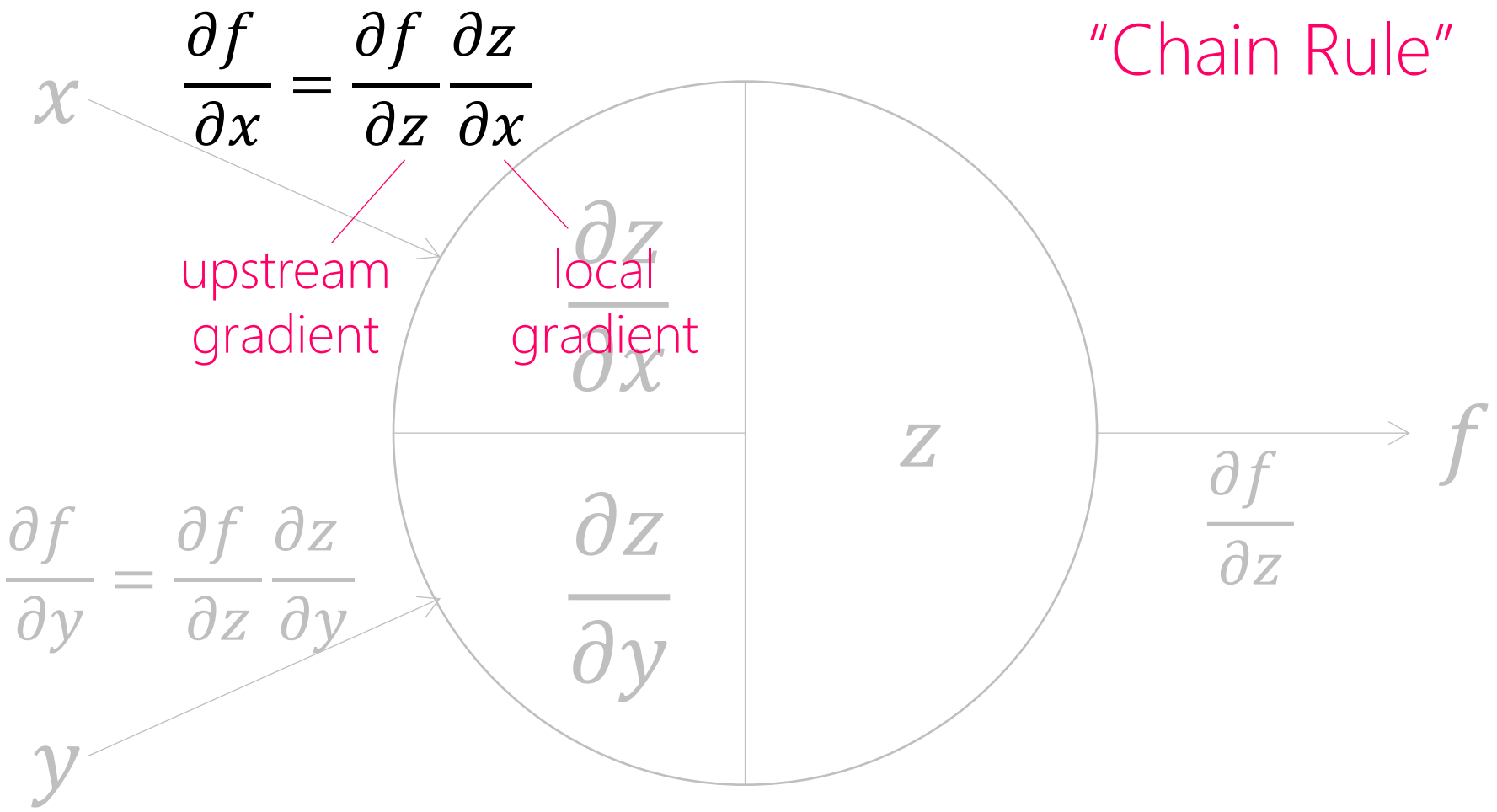
$f = qz$ $\quad \frac{\partial f}{\partial q} = z$ $\quad \frac{\partial f}{\partial z} = q$

What we want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

Backpropagation



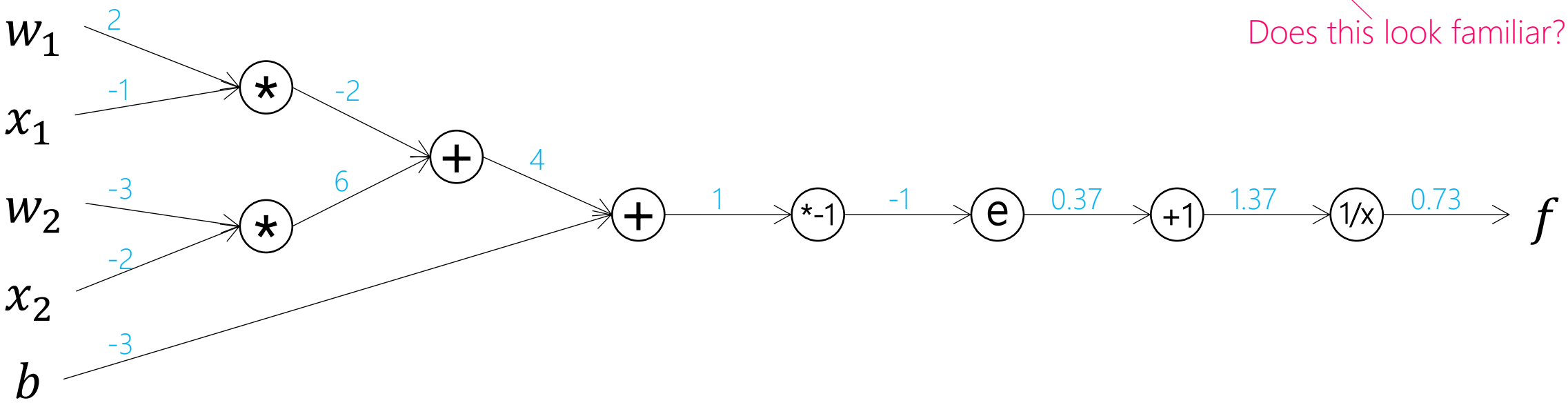
Backpropagation



Another Example

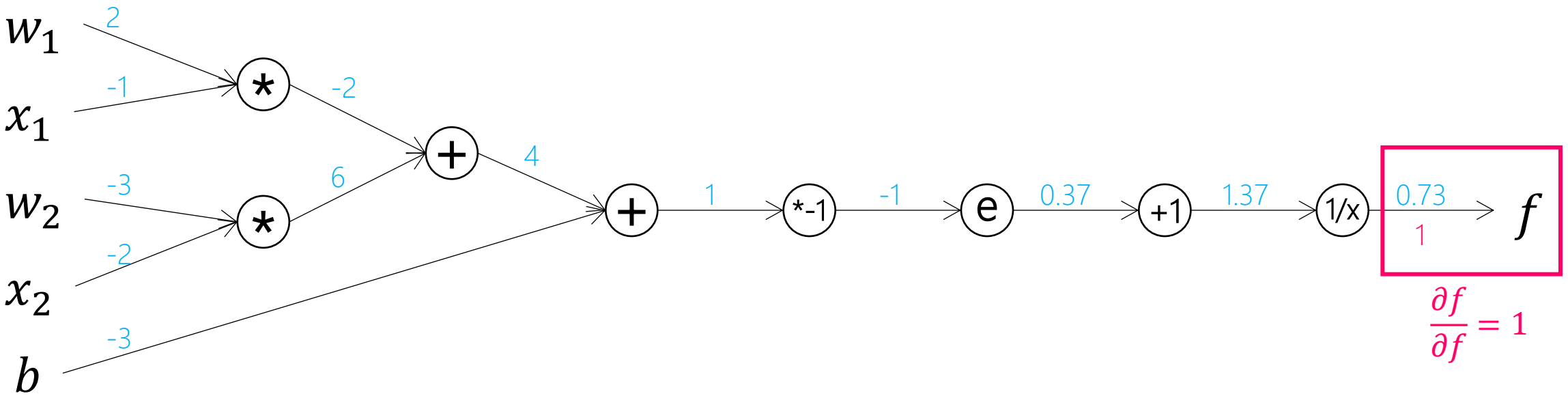
$$f(w,x) = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$

Does this look familiar?



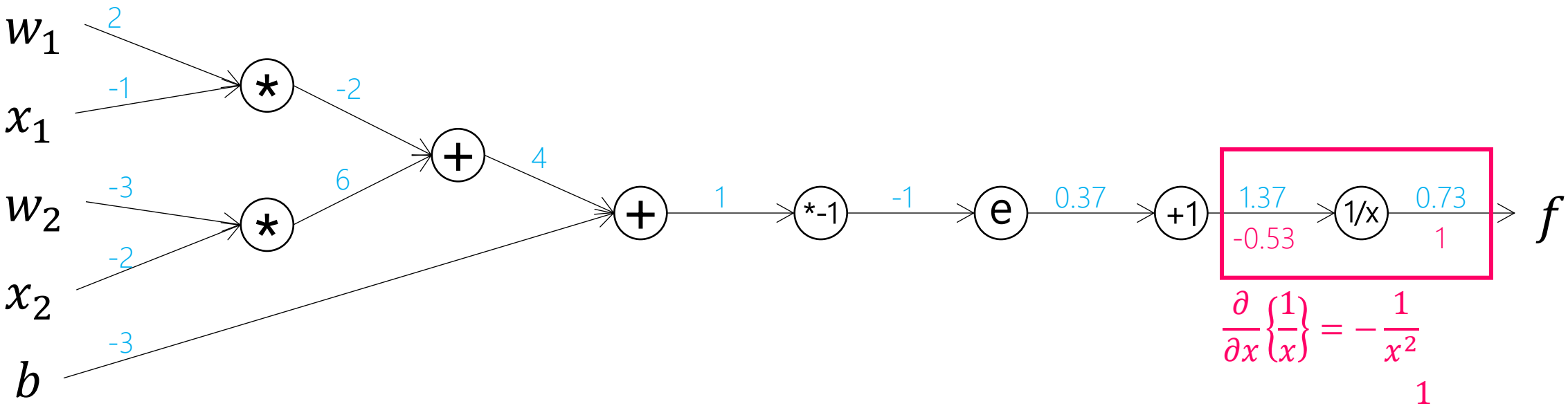
Another Example

$$f(w,x) = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$



Another Example

$$f(w,x) = \frac{1}{1+e^{-(w_1x_1+w_2x_2+b)}}$$



$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

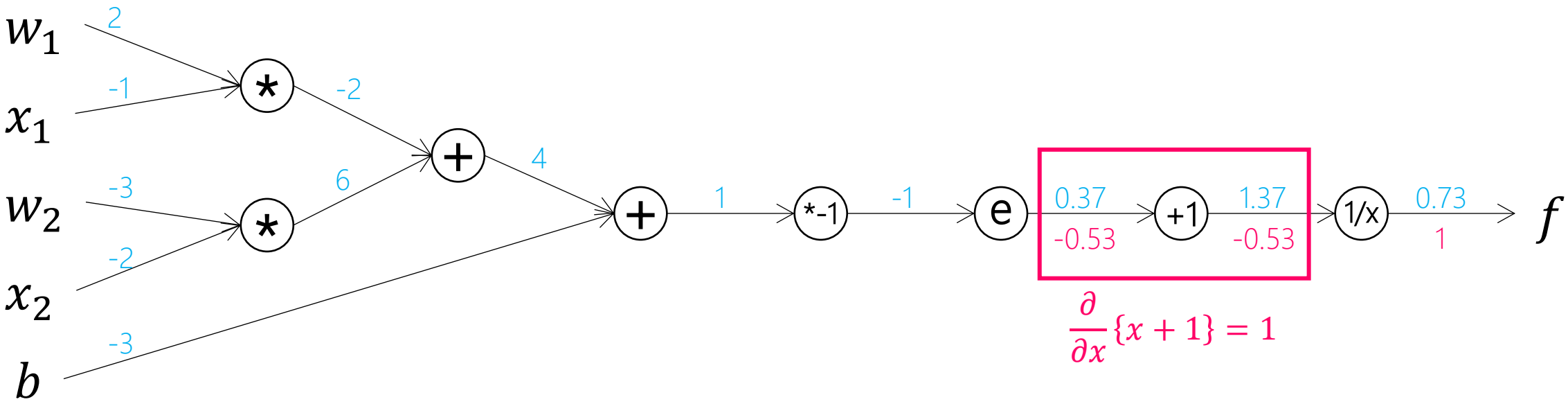
$$= -\frac{1}{1.37^2} = -0.53$$

Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

Another Example

$$f(w,x) = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$

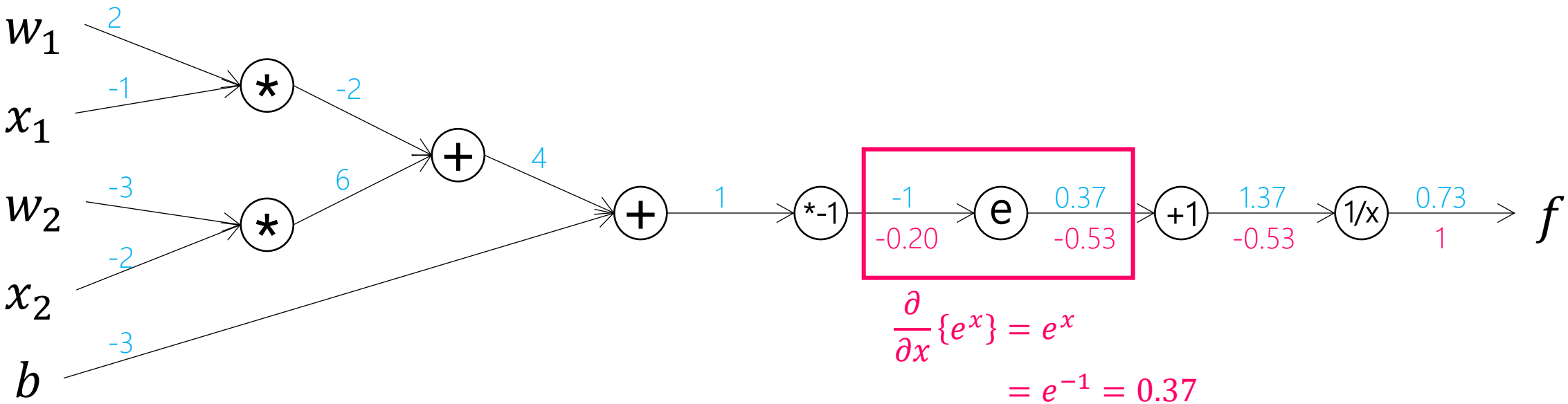


Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

Another Example

$$f(w,x) = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$



$$\frac{\partial}{\partial x}\{e^x\} = e^x$$

$$= e^{-1} = 0.37$$

$$0.37 * (-0.53) \approx -0.20$$

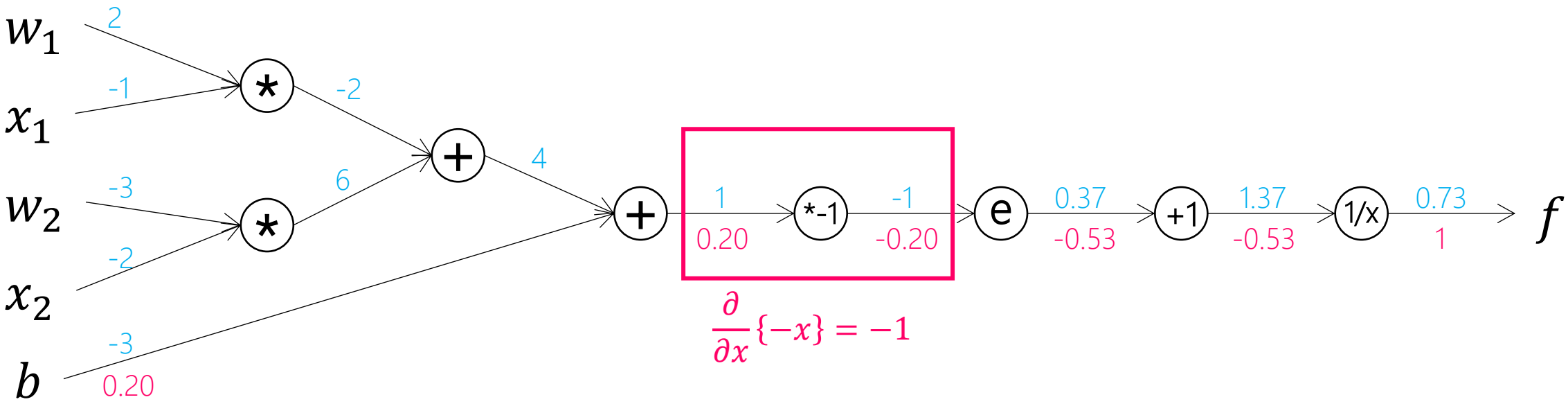
Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

$$\frac{\partial}{\partial x} e^x = e^x$$

Another Example

$$f(w,x) = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$



$$\frac{\partial}{\partial x}\{-x\} = -1$$

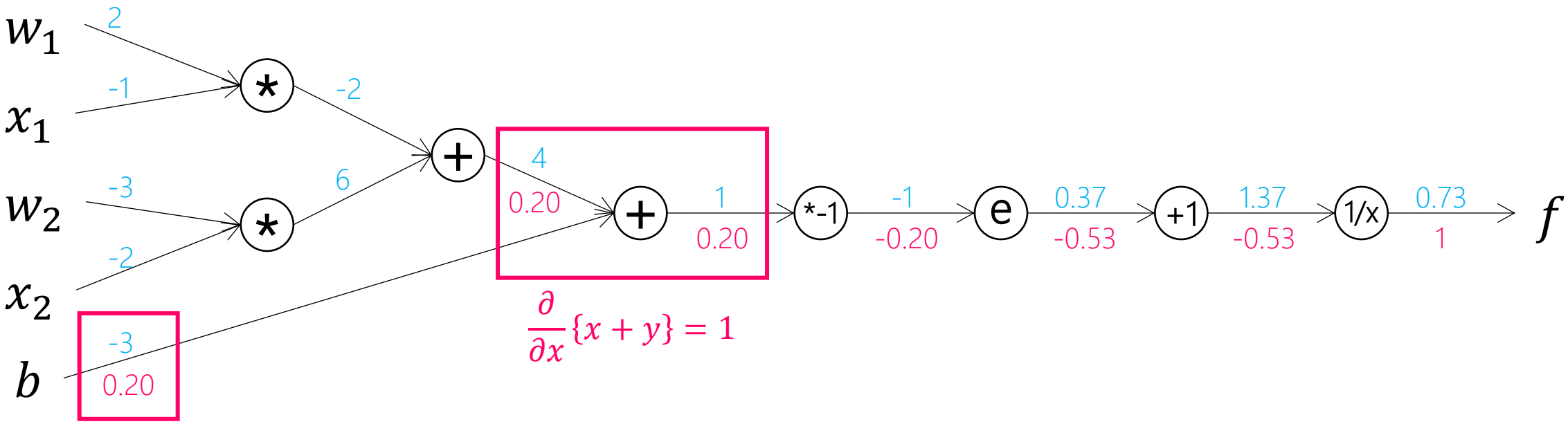
Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

$$\frac{\partial}{\partial x} e^x = e^x$$

Another Example

$$f(w,x) = \frac{1}{1 + e^{-(w_1x_1 + w_2x_2 + b)}}$$



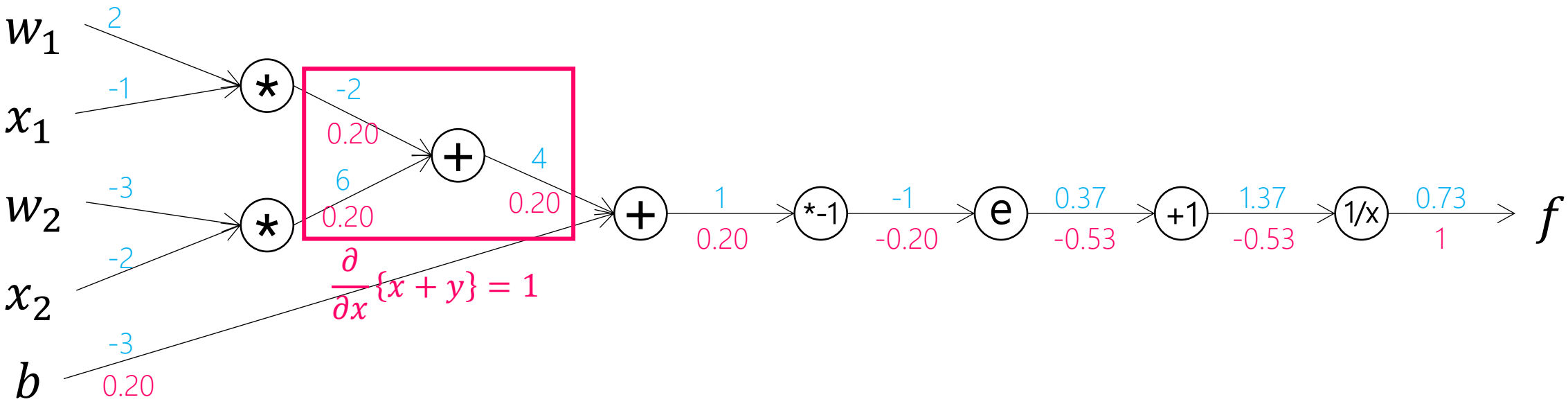
Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

$$\frac{\partial}{\partial x}e^x = e^x$$

Another Example

$$f(w,x) = \frac{1}{1+e^{-(w_1x_1+w_2x_2+b)}}$$



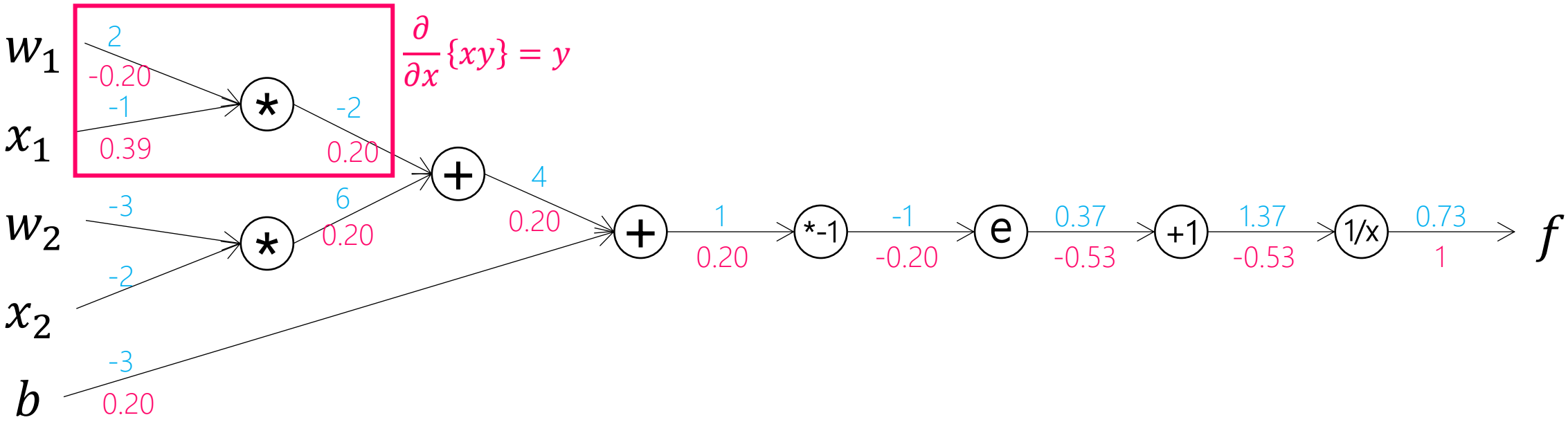
Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

$$\frac{\partial}{\partial x}e^x = e^x$$

Another Example

$$f(w,x) = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$



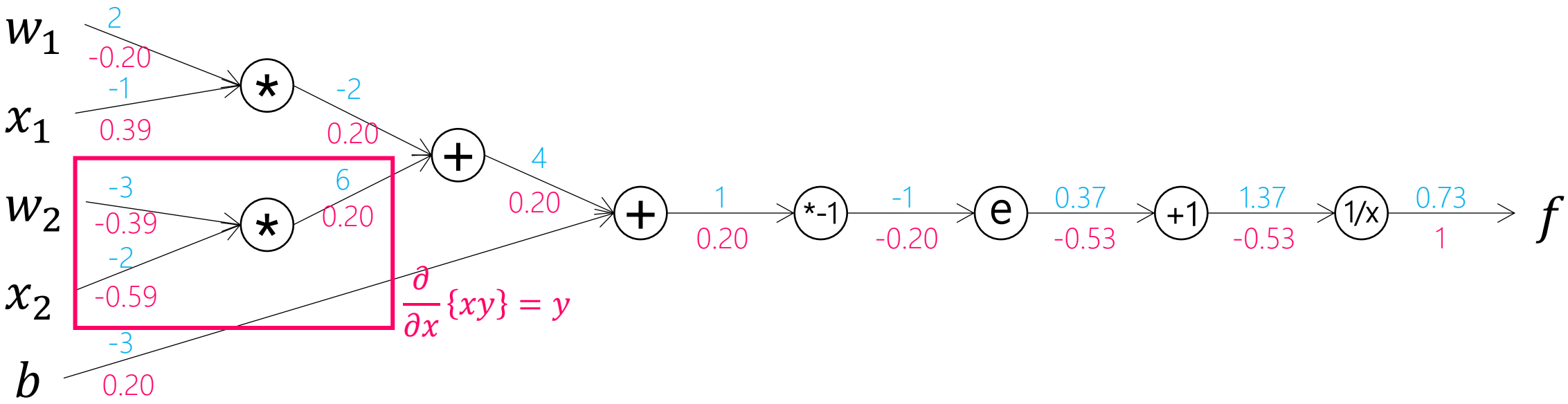
Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

$$\frac{\partial}{\partial x} e^x = e^x$$

Another Example

$$f(w,x) = \frac{1}{1 + e^{-(w_1x_1 + w_2x_2 + b)}}$$



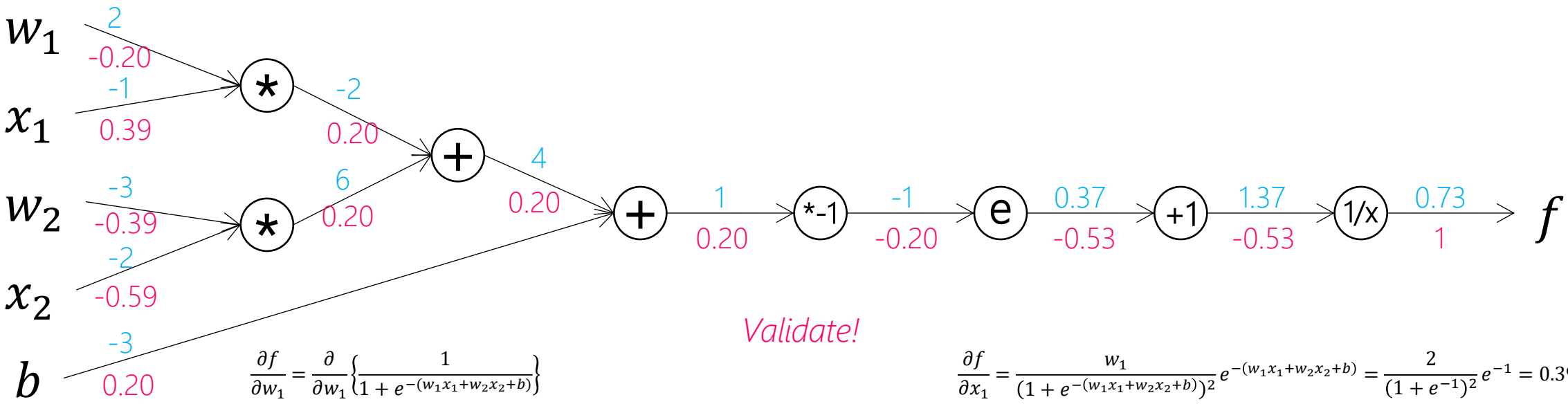
Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

$$\frac{\partial}{\partial x}e^x = e^x$$

Another Example

$$f(w,x) = \frac{1}{1+e^{-(w_1x_1+w_2x_2+b)}}$$



Validate!

$$\frac{\partial f}{\partial w_1} = \frac{\partial}{\partial w_1}\left\{\frac{1}{1+e^{-(w_1x_1+w_2x_2+b)}}\right\}$$

$$= -\frac{1}{(1+e^{-(w_1x_1+w_2x_2+b)})^2}\frac{\partial}{\partial w_1}\left\{1+e^{-(w_1x_1+w_2x_2+b)}\right\}$$

$$= -\frac{1}{(1+e^{-(w_1x_1+w_2x_2+b)})^2}e^{-(w_1x_1+w_2x_2+b)}\frac{\partial}{\partial w_1}\{-(w_1x_1+w_2x_2+b)\}$$

$$= \frac{x_1}{(1+e^{-(w_1x_1+w_2x_2+b)})^2}e^{-(w_1x_1+w_2x_2+b)}$$

$$= \frac{-1}{(1+e^{-(-2+6-3)})^2}e^{-(-2+6-3)} = -0.1966$$

$$\frac{\partial f}{\partial x_1} = \frac{w_1}{(1+e^{-(w_1x_1+w_2x_2+b)})^2}e^{-(w_1x_1+w_2x_2+b)} = \frac{2}{(1+e^{-1})^2}e^{-1} = 0.3932$$

$$\frac{\partial f}{\partial w_2} = \frac{x_2}{(1+e^{-(w_1x_1+w_2x_2+b)})^2}e^{-(w_1x_1+w_2x_2+b)} = \frac{-2}{(1+e^{-1})^2}e^{-1} = -0.3932$$

$$\frac{\partial f}{\partial x_2} = \frac{w_2}{(1+e^{-(w_1x_1+w_2x_2+b)})^2}e^{-(w_1x_1+w_2x_2+b)} = \frac{-3}{(1+e^{-1})^2}e^{-1} = -0.5898$$

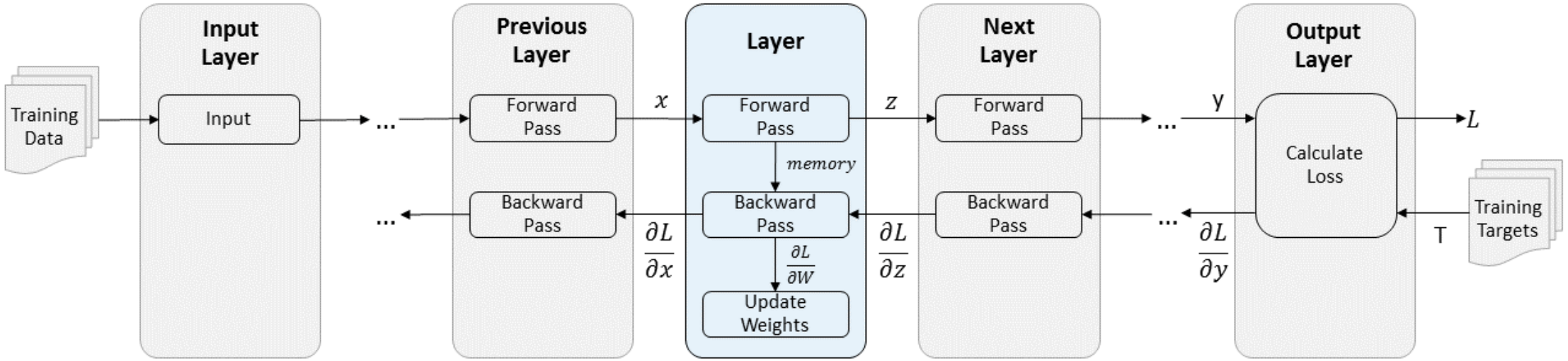
$$\frac{\partial f}{\partial b} = \frac{1}{(1+e^{-(w_1x_1+w_2x_2+b)})^2}e^{-(w_1x_1+w_2x_2+b)} = \frac{1}{(1+e^{-1})^2}e^{-1} = 0.1966$$

Cheat Sheet:

$$\frac{\partial}{\partial x}\left\{\frac{1}{x}\right\} = -\frac{1}{x^2}$$

$$\frac{\partial}{\partial x}e^x = e^x$$

Putting them all together



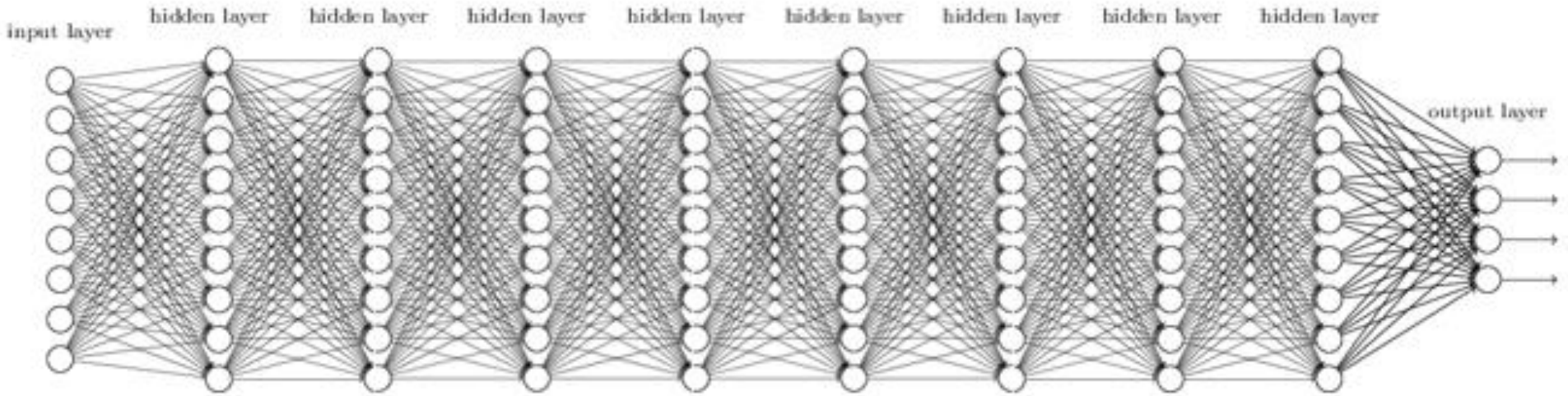
https://www.mathworks.com/help/nnet/ug/define-custom-deep-learning-layers.html

Putting them all together

- Linear model: $f = Wx + b$
- Neural network with a single hidden layer: $f = W_2 \max(0, W_1 x + b_1) + b_2$
- Neural network with two hidden layers:

$$f = W_3 \max(0, W_2 \max(0, W_1 x + b_1) + b_2) + b_3$$

- … stacking up hidden layers → "deep" neural networks



Deep Neural Network

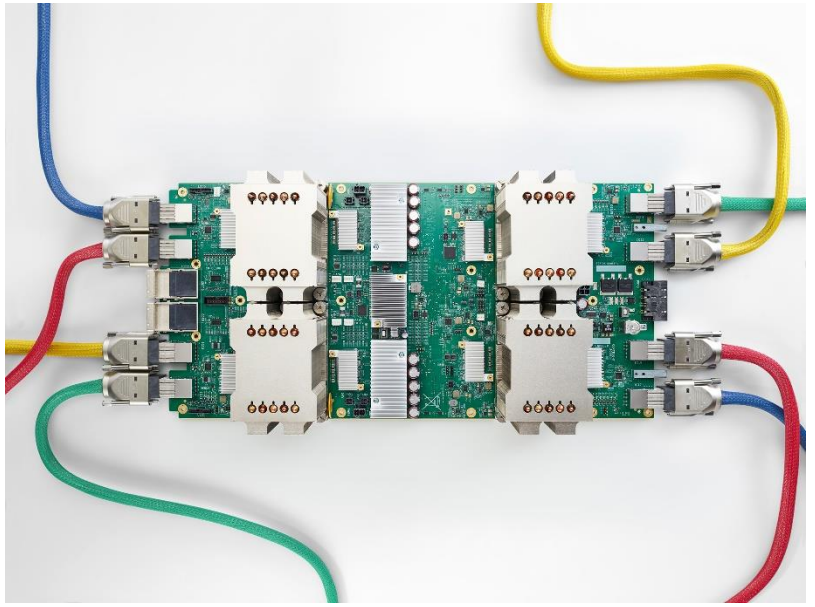
5MB Hard Drive Being Shipped by IBM, 1956


A FEW
MOMENTS LATER

40 years later…

NVIDIA GeForce Quadro GV100

NVIDIA Jetson TX2



Google Tensor Processing Unit (TPU)


IMAGENET

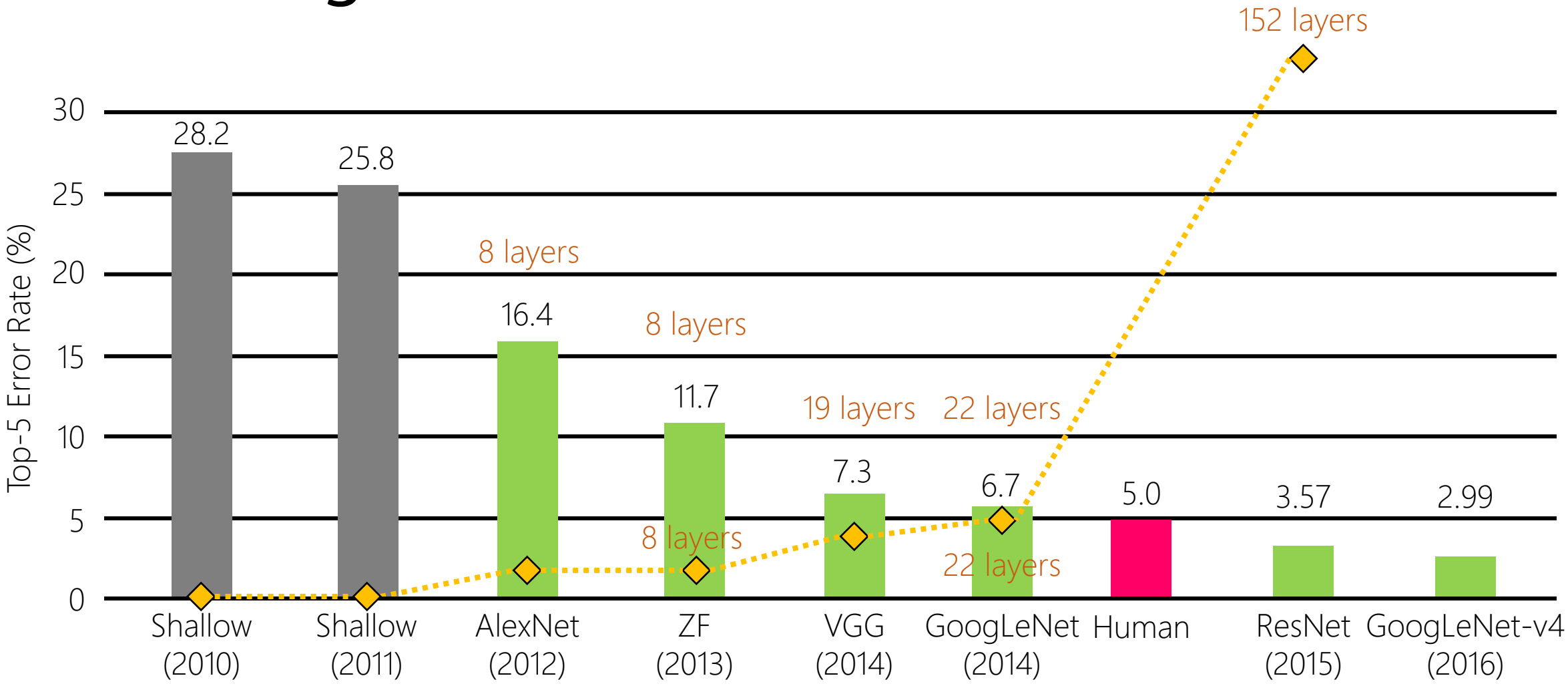
ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- ImageNet (First appearance as a poster at CVPR 2009)
  - Over 14 million images with hand-annotated labels (crowdsourced via M-turk)
  - Over 20 thousand categories
  - 1M+ images with bounding boxes.

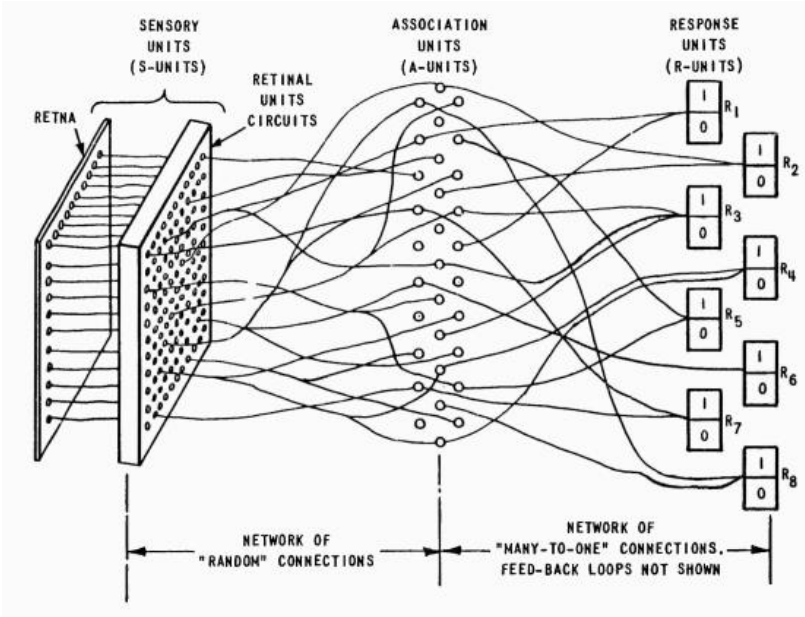
- ImageNet Challenge
  - Since 2010
  - Uses a trimmed set of thousand non-overlapping classes.
  - Humans error is about 5%*
  - Playground of computer vision researchers.
  - Marked the start of the current AI boom.

Russakovsky et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, *115*(3), 211-252.

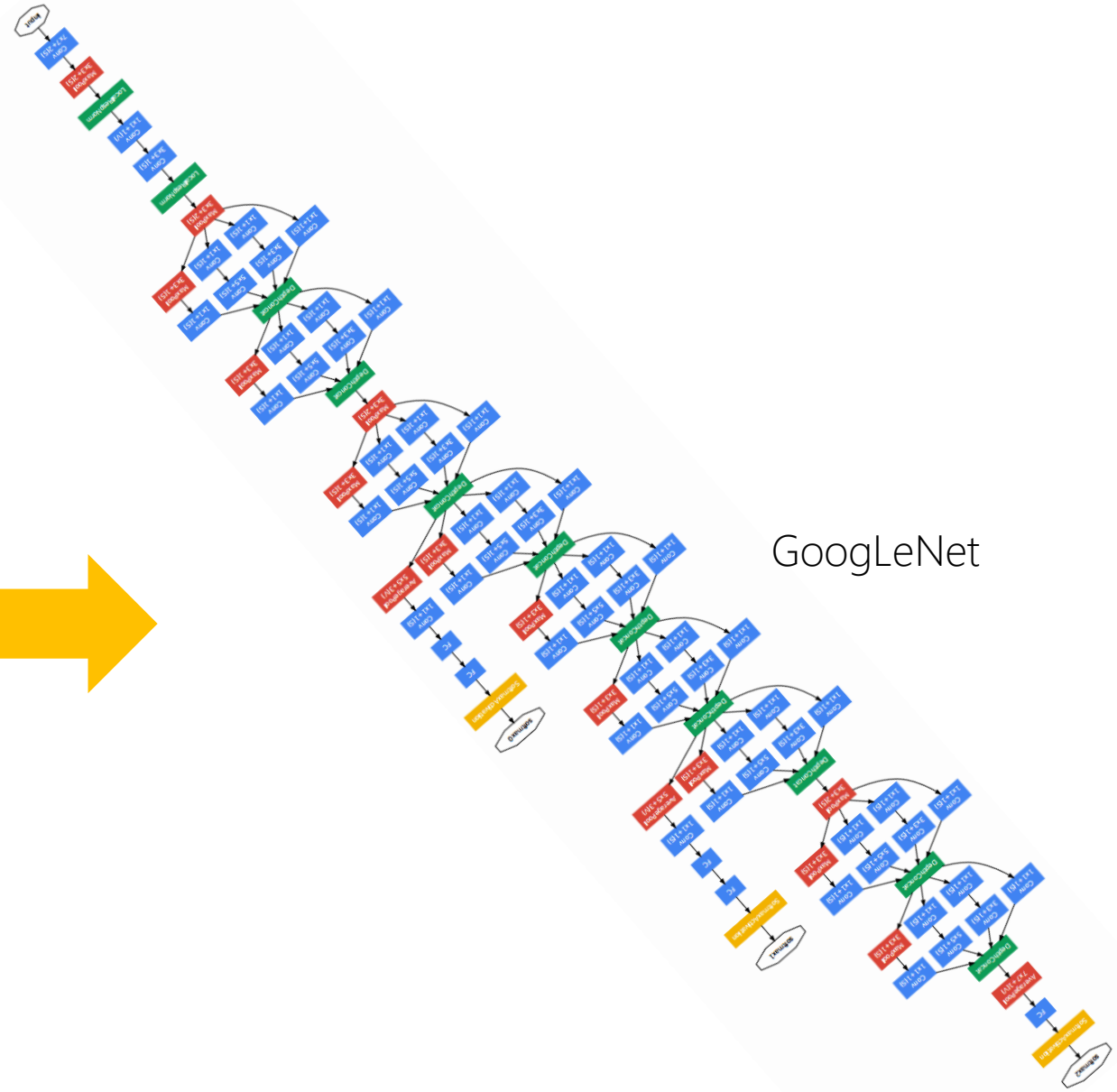
ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



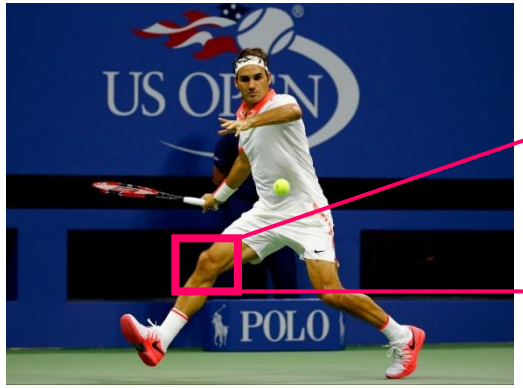
Going Deeper



Perceptron



GoogLeNet

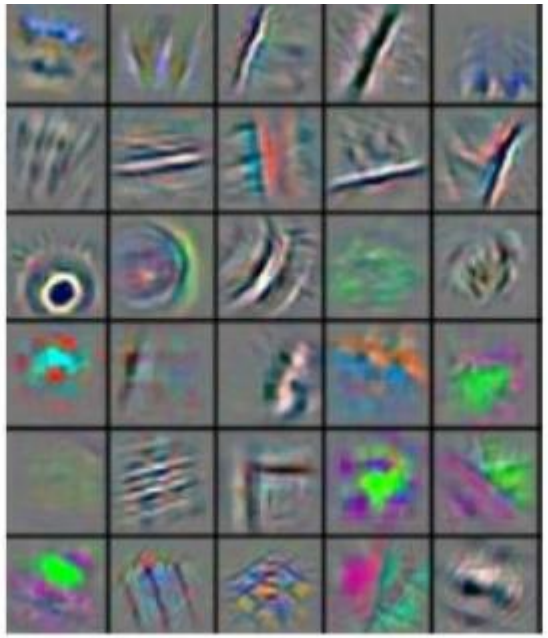


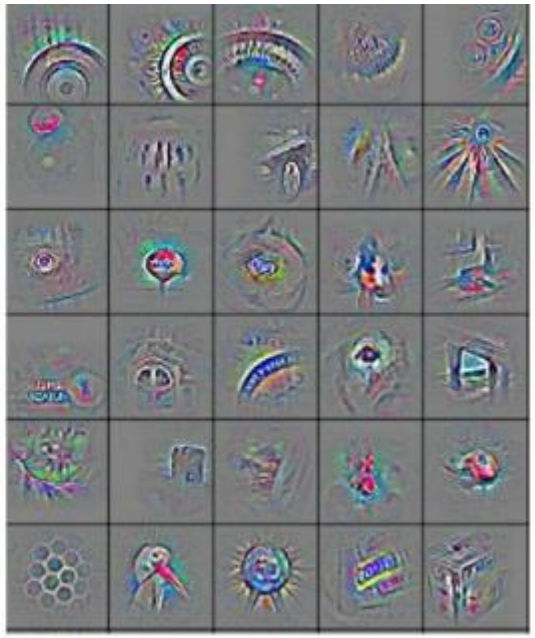
Low-level Features

Mid-level Features

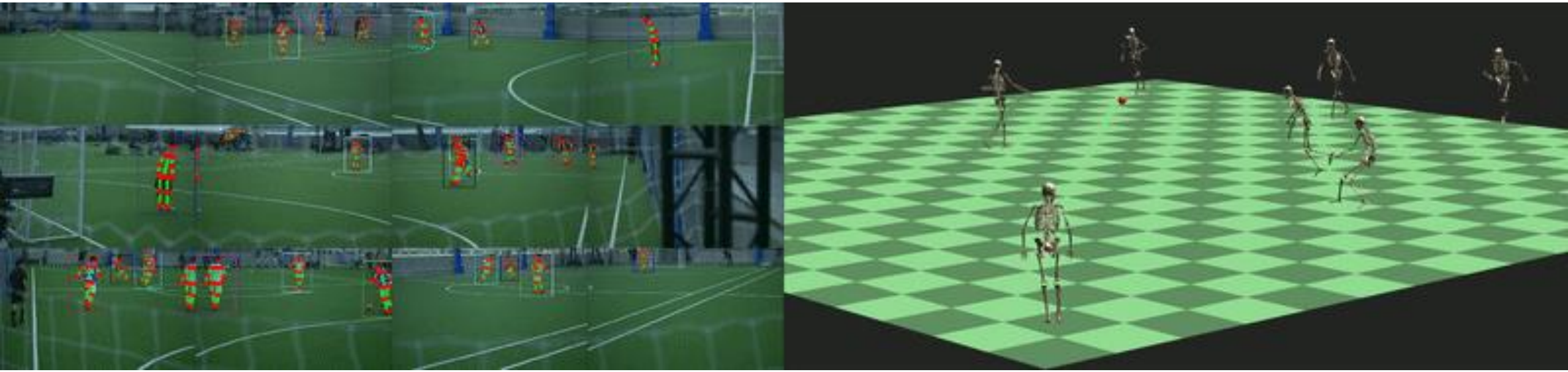
High-level Features

Shoulder: 0.01
Elbow: 0.12
Wrist: 0.01
Knee: 0.74
Ankle: 0.03
...



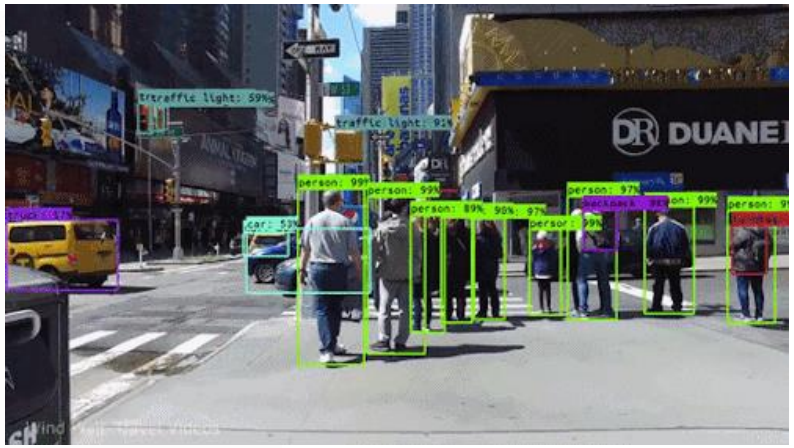


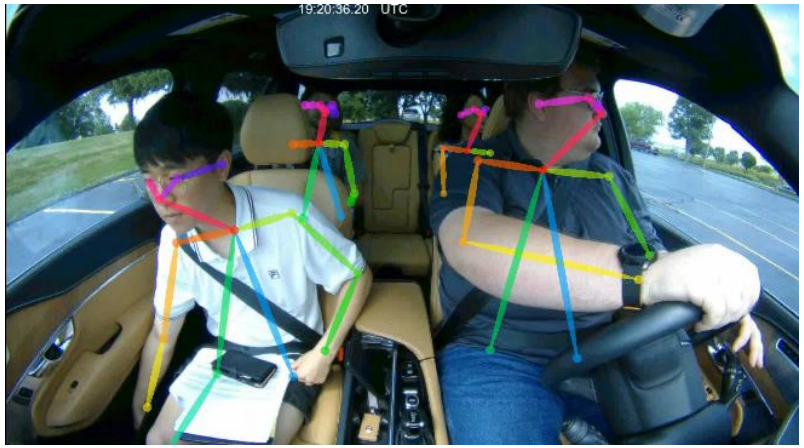
e.g. Pose Estimation

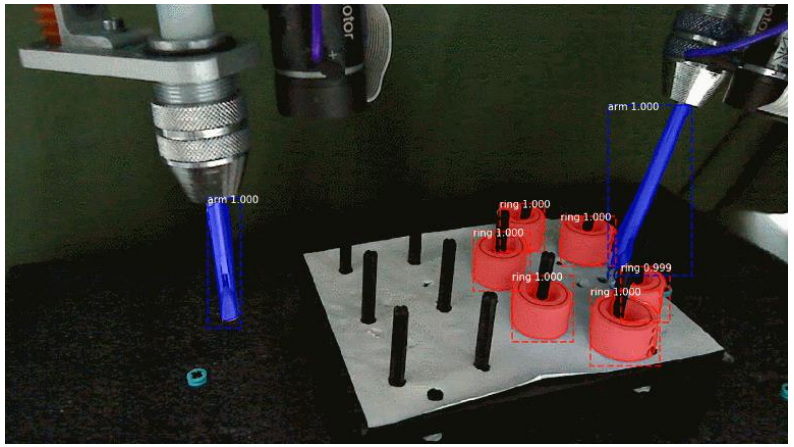


Ohashi et al. (2020). arXiv:2001.05613

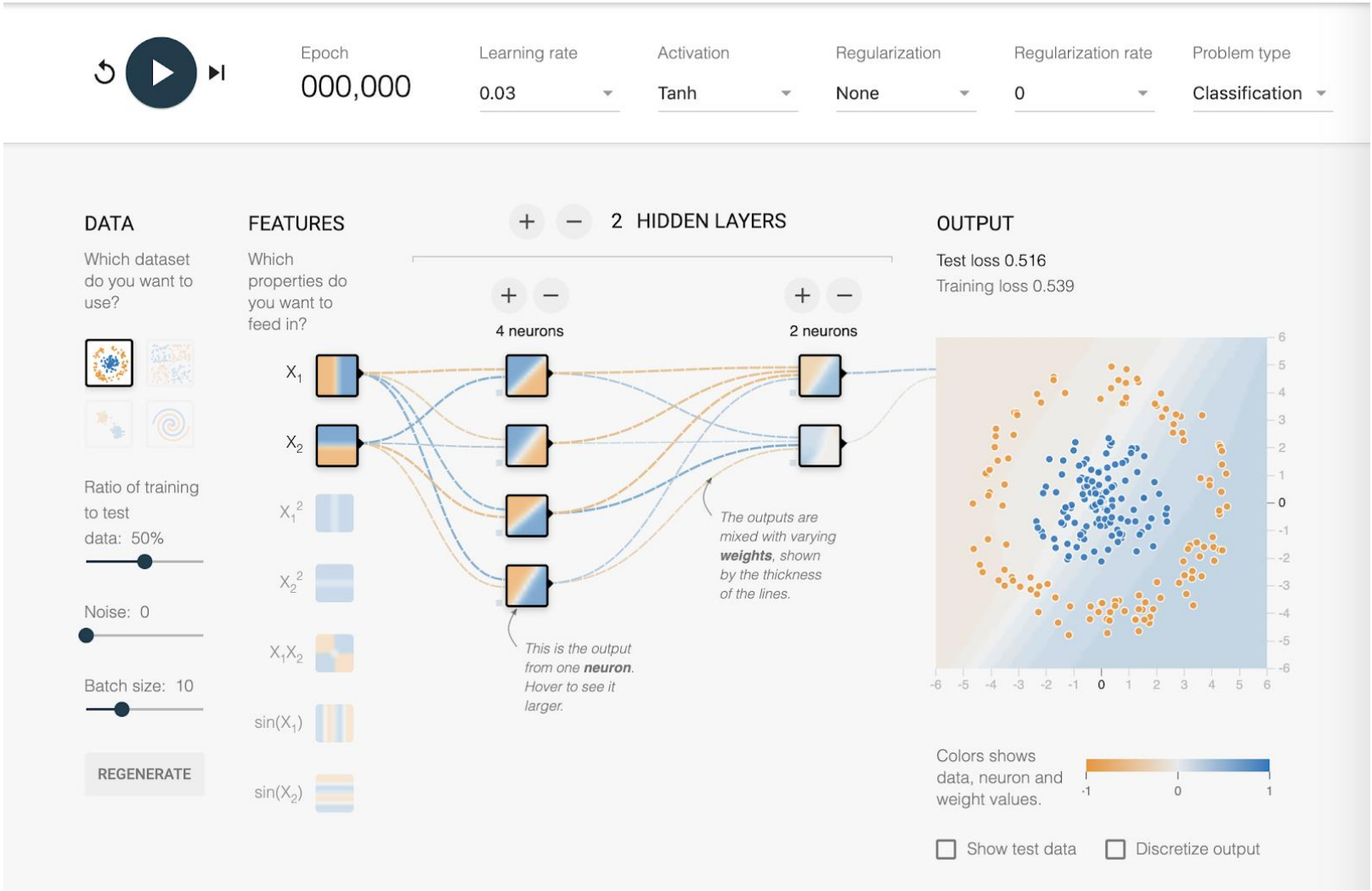
…and many others







TensorFlow Playground



https://playground.tensorflow.org

Coding Time!

- https://github.com/stephenbaek/padl
- ‘labs’ > ‘01_introduction_to_pytorch.ipynb’
- Either Google Colab or your local machine