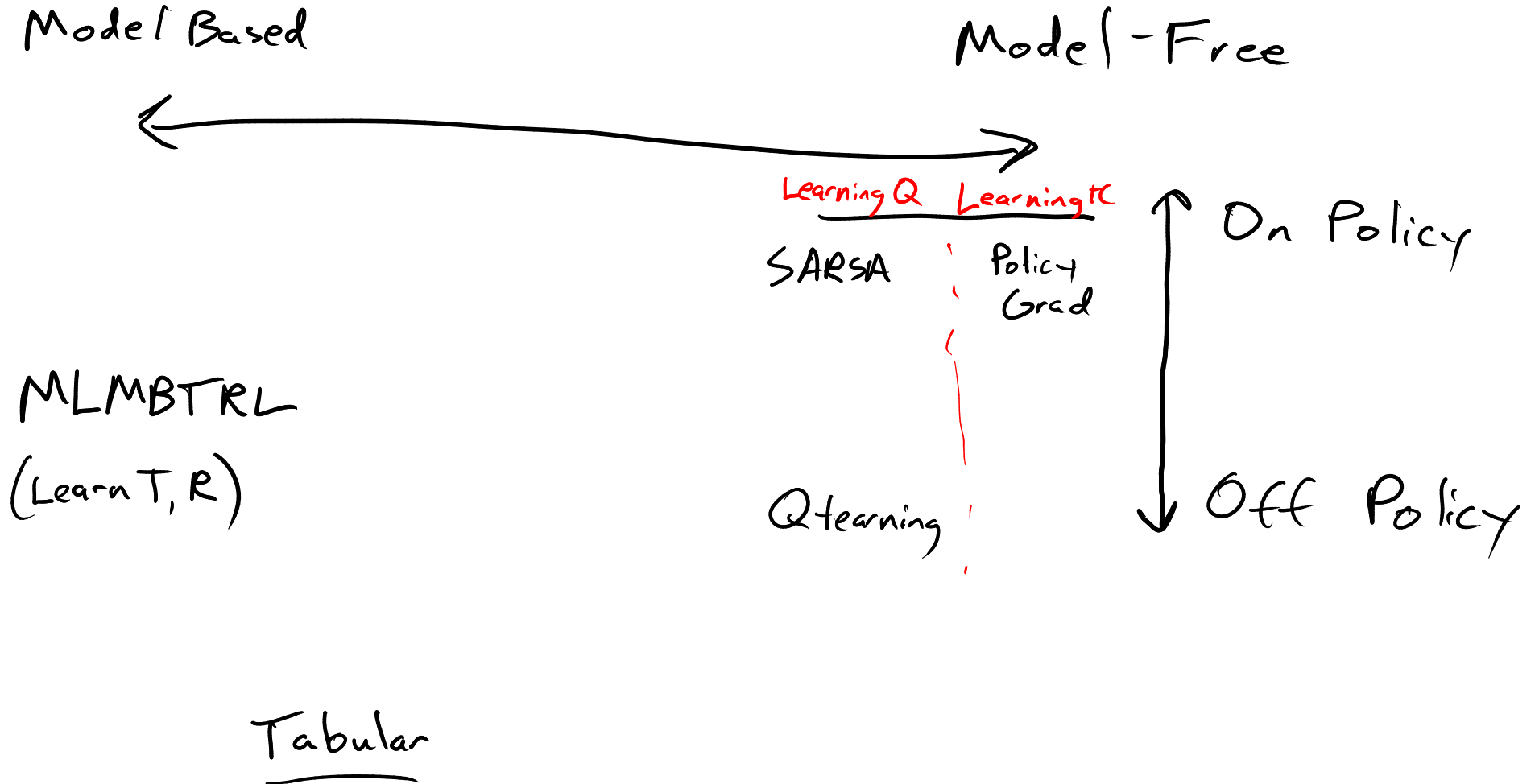


Map of RL Algorithms



This Time

Challenges in Reinforcement Learning:


- Exploration vs Exploitation
- Credit Assignment
- Generalization ←

Function Approximation

Function Approximation

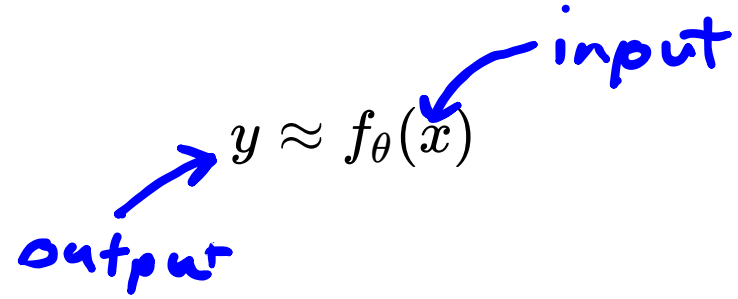
$$y \approx f_{\theta}(x)$$

Function Approximation

$$y \approx f_{\theta}(x)$$


input

Function Approximation



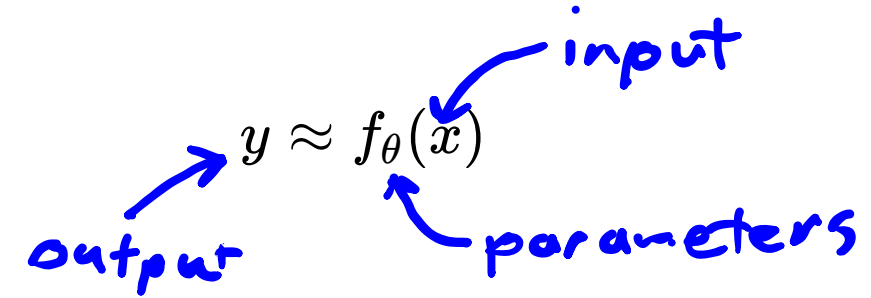
A diagram illustrating the function approximation equation $y \approx f_{\theta}(x)$. The equation is written in black. Two blue arrows are drawn: one points from the handwritten word "output" to the variable y , and another points from the handwritten word "input" to the variable x inside the function $f_{\theta}(x)$.

$$y \approx f_{\theta}(x)$$

output

input

Function Approximation



A diagram illustrating the function approximation equation $y \approx f_{\theta}(x)$. The equation is centered, with three handwritten blue arrows pointing to its components: one from the word "output" to y , one from the word "input" to x , and one from the word "parameters" to θ .

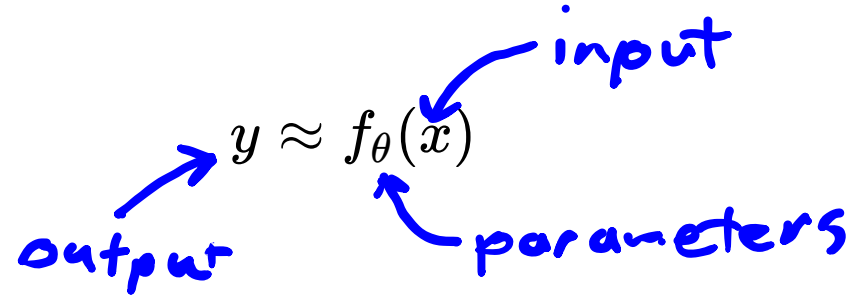
$$y \approx f_{\theta}(x)$$

output

input

parameters

Function Approximation

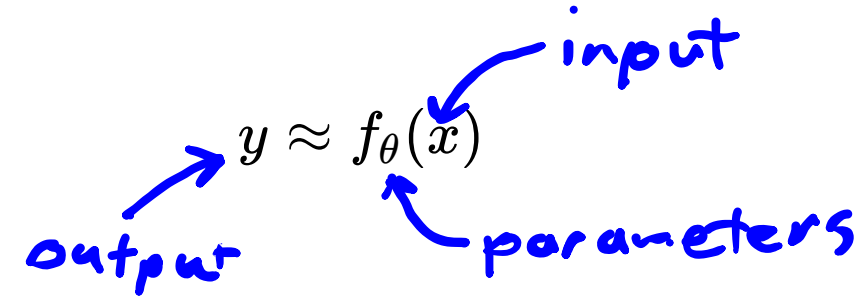


A diagram showing the equation $y \approx f_{\theta}(x)$ with three handwritten blue arrows and labels. An arrow points from the label "input" to the variable x in the function argument. Another arrow points from the label "parameters" to the symbol θ in the subscript of the function. A third arrow points from the label "output" to the variable y on the left side of the equation.

Previously, Linear:

$$f_{\theta}(x) = \theta^{\top} \beta(x)$$

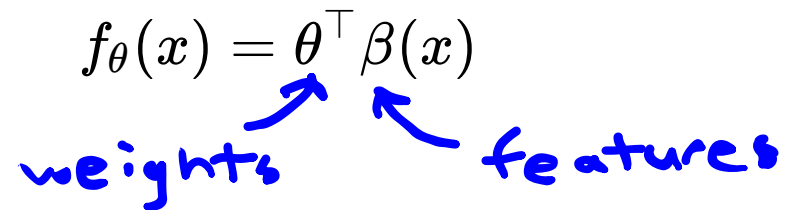
Function Approximation



A diagram showing the general function approximation equation $y \approx f_{\theta}(x)$. Handwritten blue arrows point from the terms to their meanings: 'input' points to x , 'parameters' points to θ , and 'output' points to y .

$$y \approx f_{\theta}(x)$$

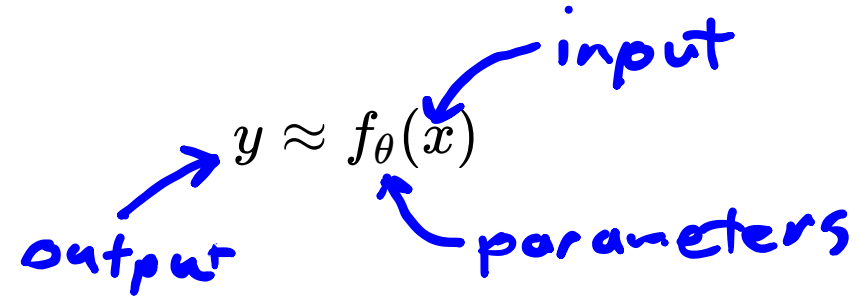
Previously, Linear:



A diagram showing the linear function approximation equation $f_{\theta}(x) = \theta^{\top} \beta(x)$. Handwritten blue arrows point from the terms to their meanings: 'weights' points to θ , 'features' points to β , and the entire expression is labeled as the function output.

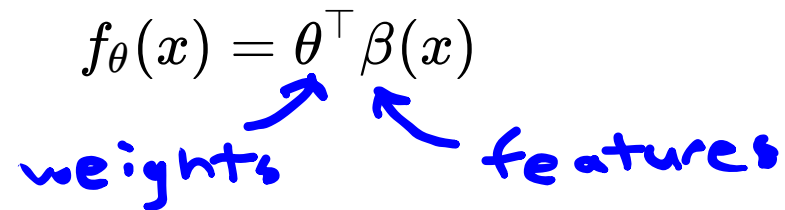
$$f_{\theta}(x) = \theta^{\top} \beta(x)$$

Function Approximation



A diagram showing the equation $y \approx f_{\theta}(x)$. A blue arrow points from the handwritten word "output" to the variable y . Another blue arrow points from the handwritten word "input" to the variable x inside the function. A third blue arrow points from the handwritten word "parameters" to the symbol θ as a subscript of the function f .

Previously, Linear:



A diagram showing the equation $f_{\theta}(x) = \theta^{\top} \beta(x)$. A blue arrow points from the handwritten word "weights" to the symbol θ . Another blue arrow points from the handwritten word "features" to the symbol β .

e.g. $\beta_i(x) = \sin(i \pi x)$

~~AI = Neural Nets~~

π_θ


Q_θ

gathering data

~~Neural Nets are
just another
function
approximator~~

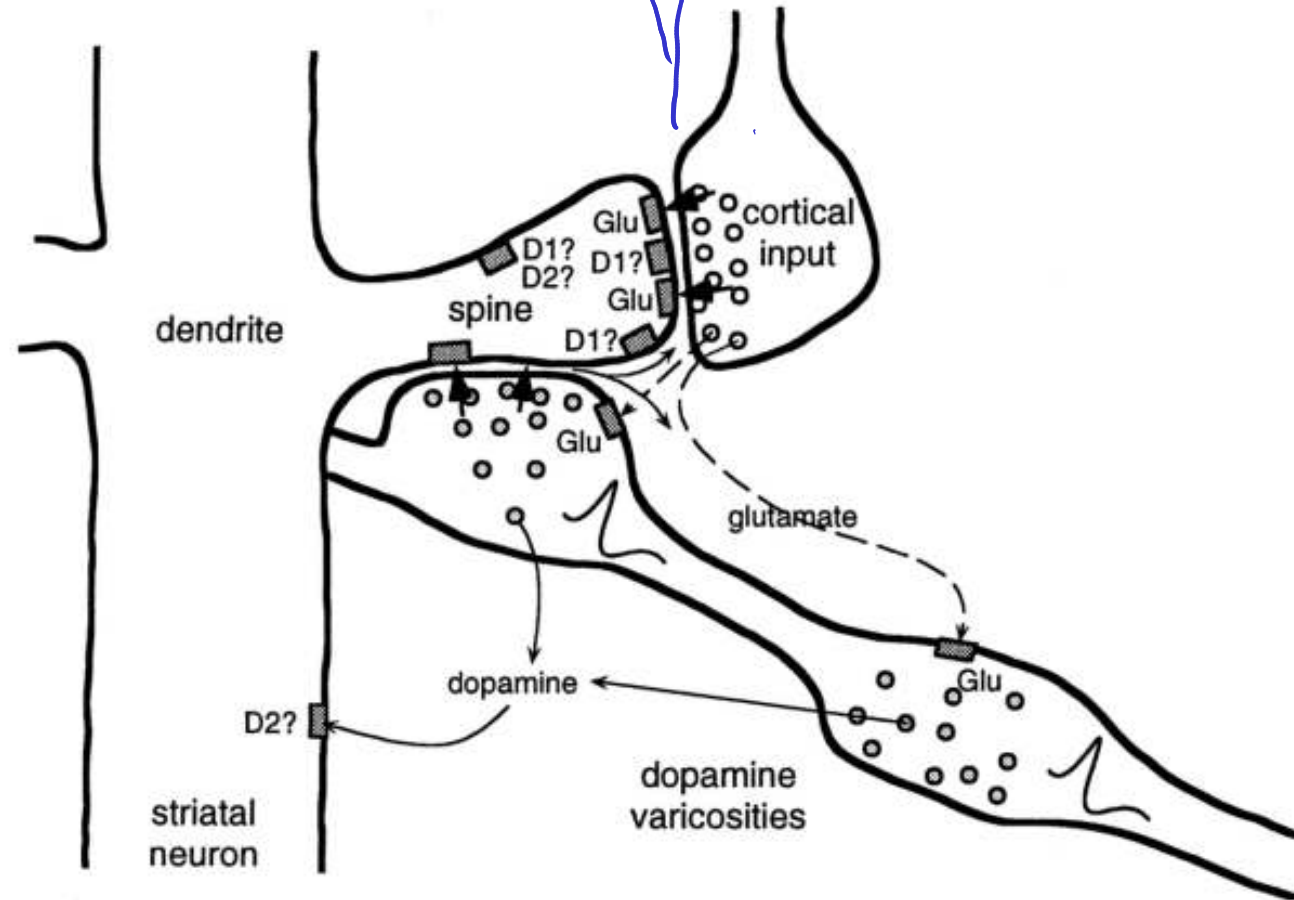
Neural Network

Neural Network

$$h(x) = \sigma(Wx + b)$$


Neural Network

$$h(x) = \sigma(Wx + b)$$



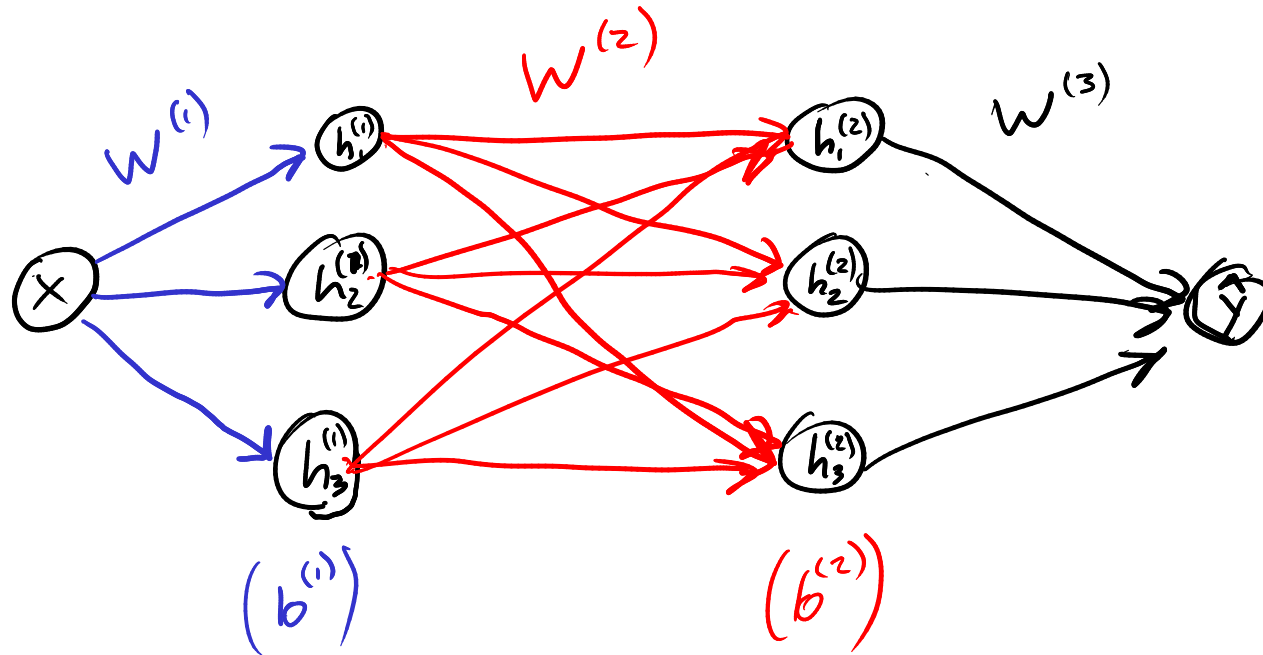
Neural Network

Neural Network

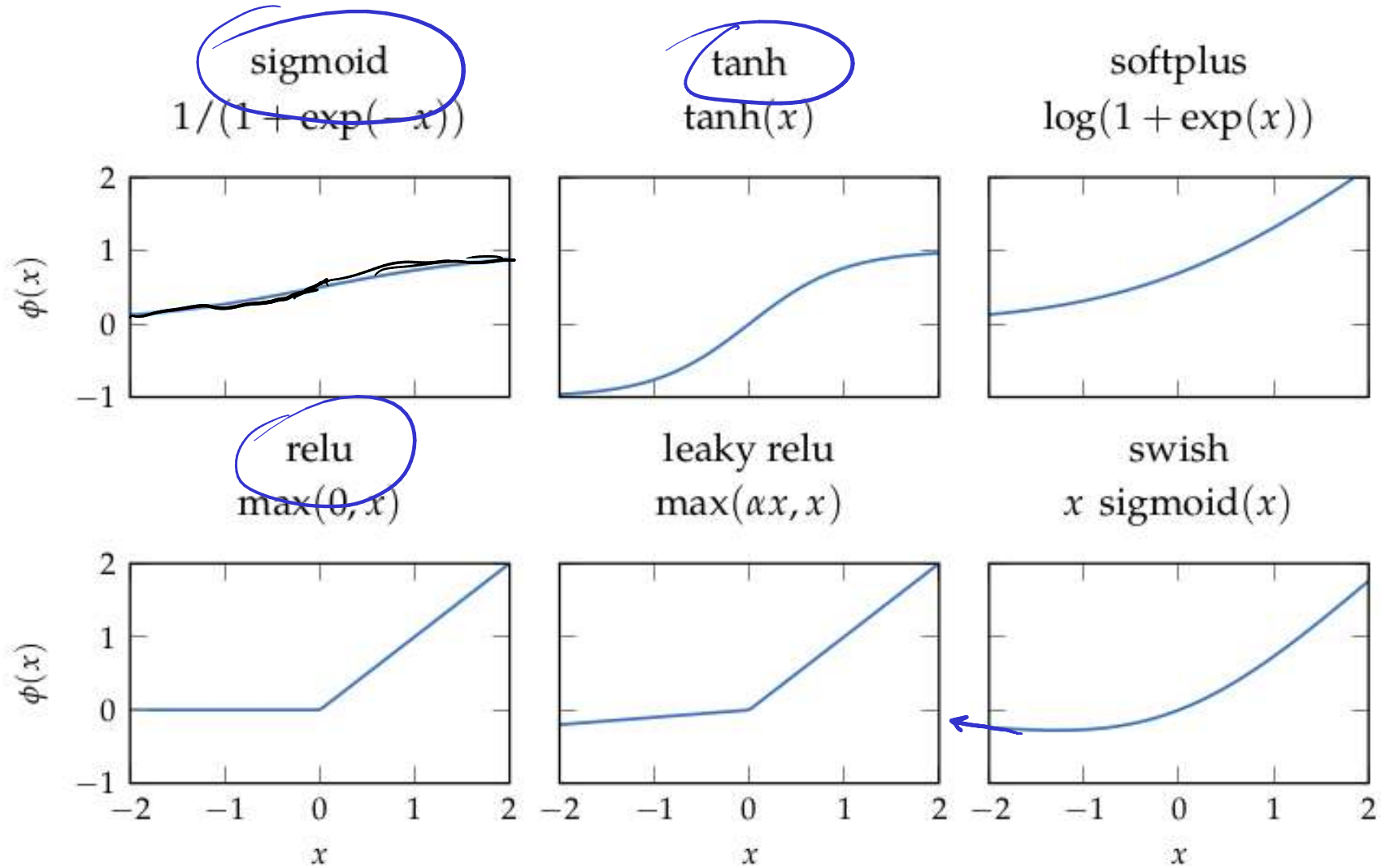
$$h(x) = \sigma(Wx + b)$$

$$f_{\theta}(x) = h^{(2)}(h^{(1)}(x)) = \sigma^{(2)}(W^{(2)} \sigma^{(1)}(W^{(1)}x + b^{(1)}) + b^{(2)})$$

$$\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)})$$

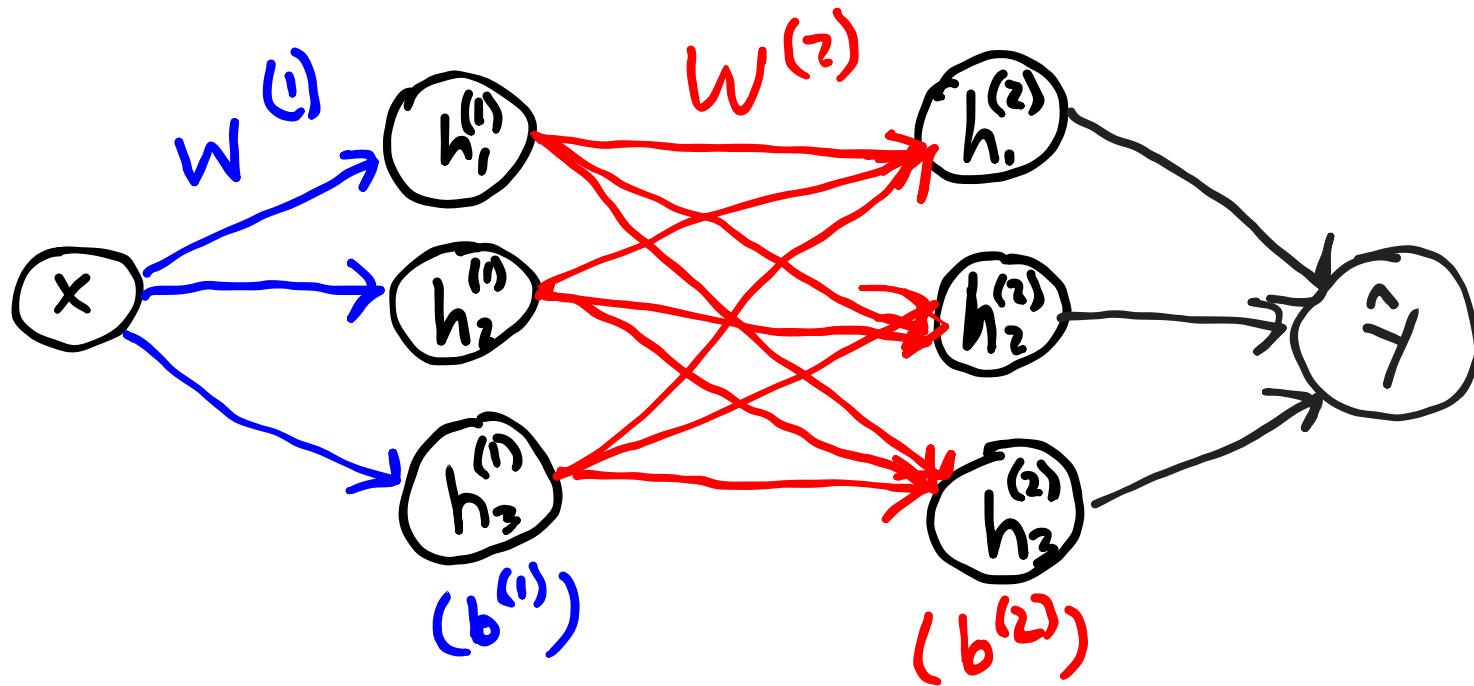


Nonlinearities

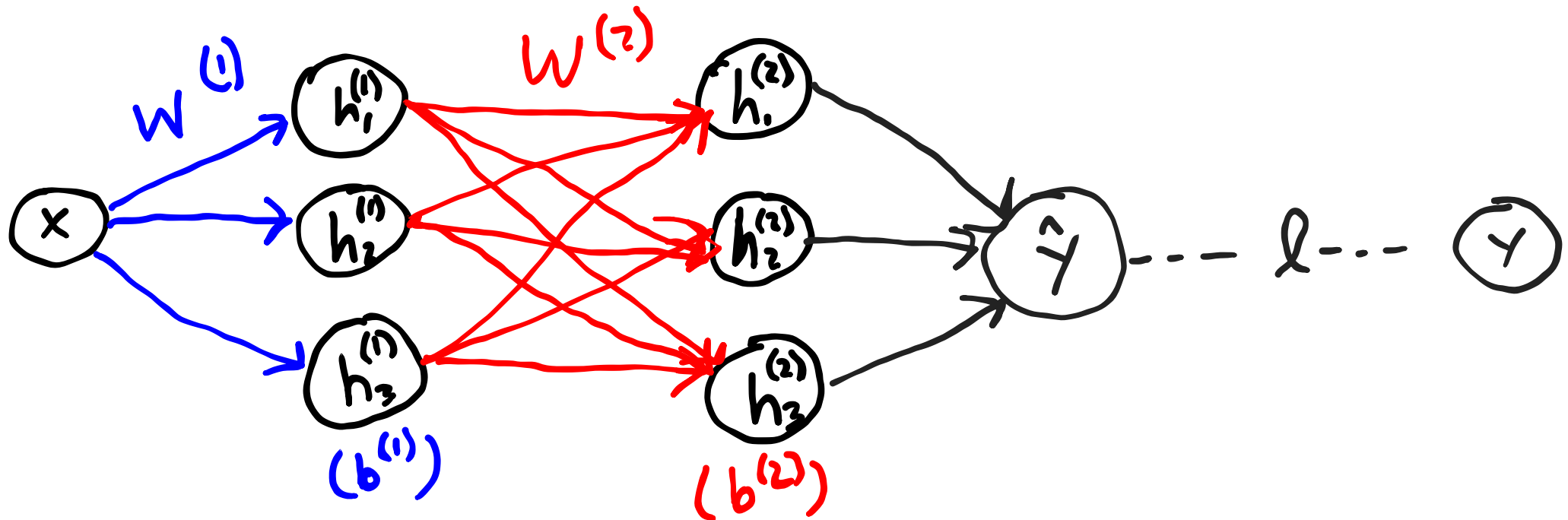


Training

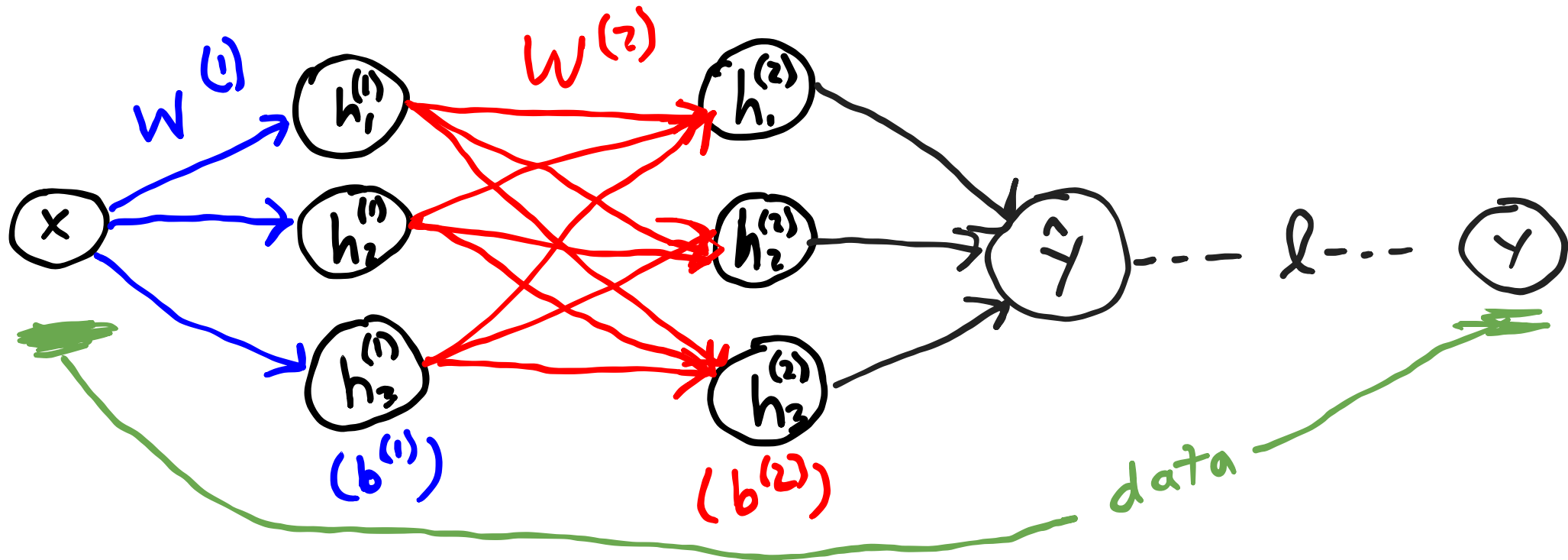
Training



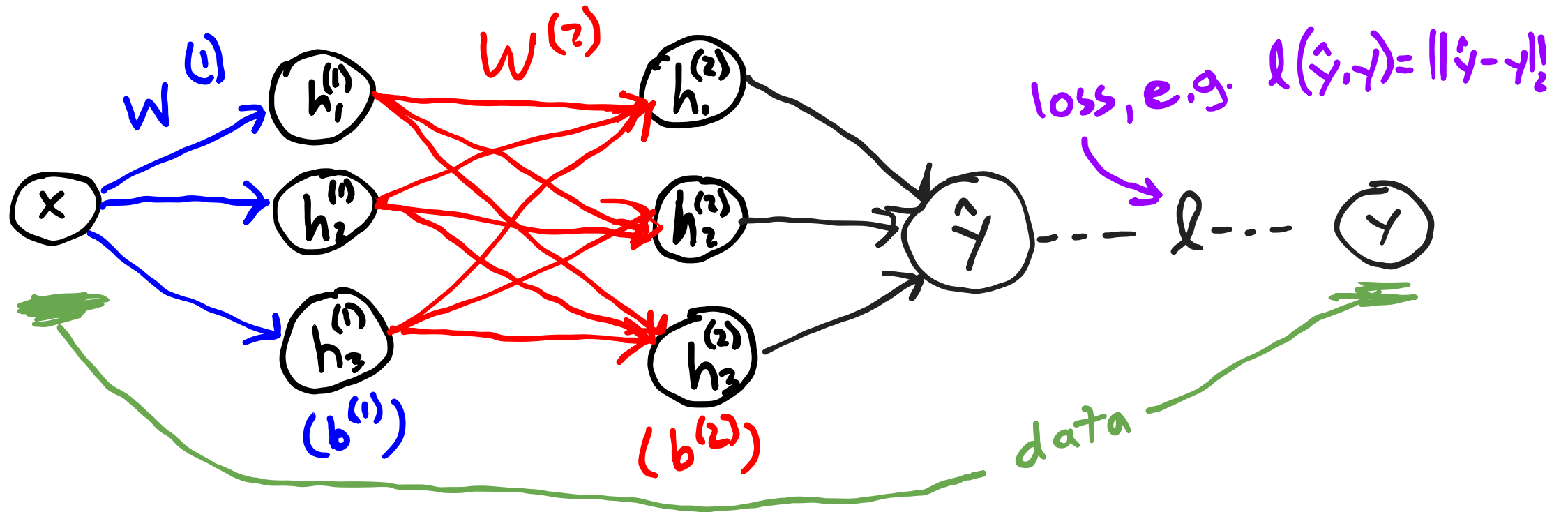
Training



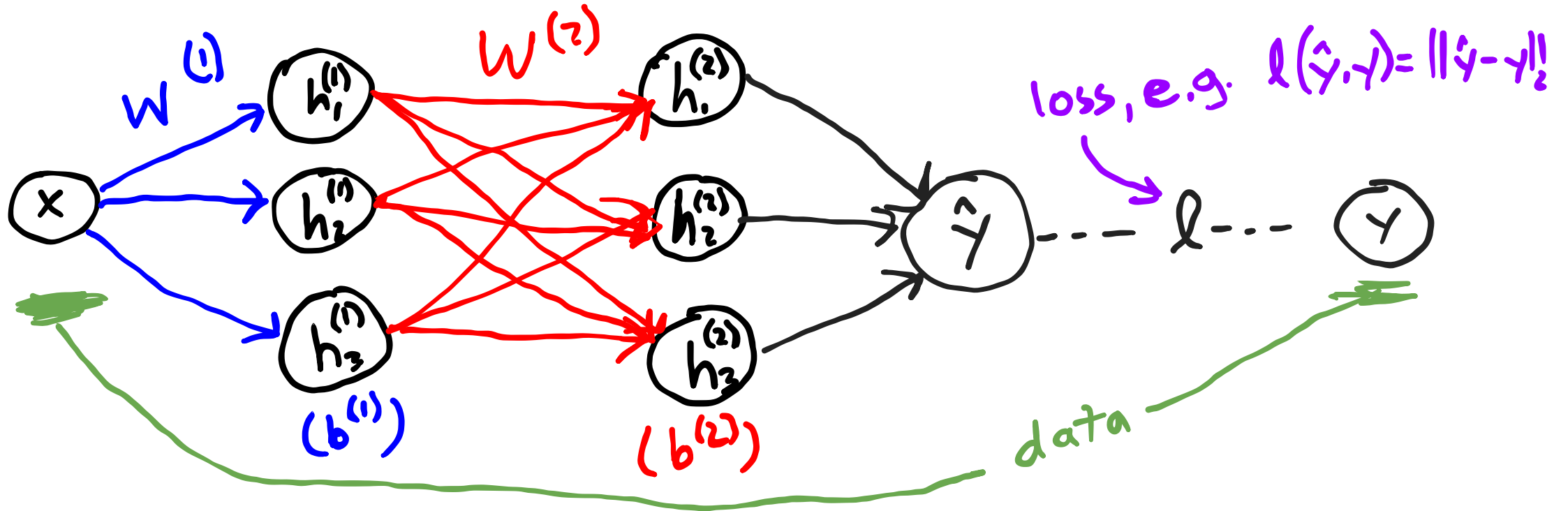
Training



Training

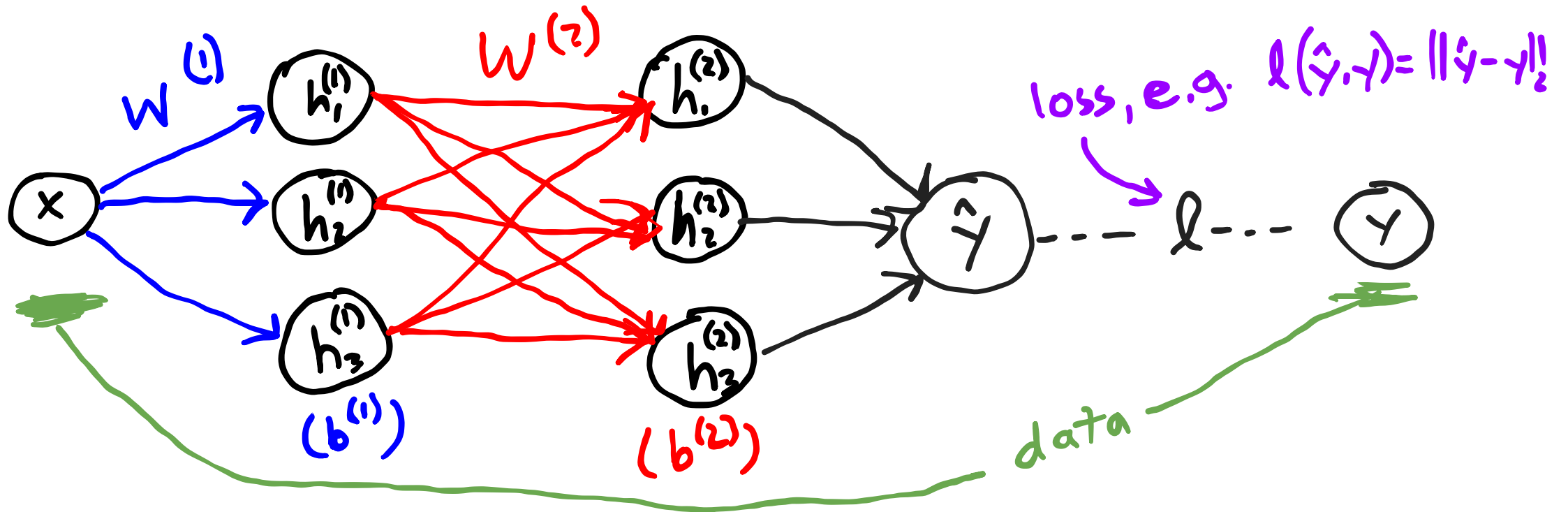


Training



$$\theta^* = \arg \min_{\theta} \sum_{(x,y) \in \mathcal{D}} l(f_{\theta}(x), y)$$

Training



$$\theta^* = \arg \min_{\theta} \sum_{(x,y) \in \mathcal{D}} l(f_{\theta}(x), y)$$

Stochastic Gradient Descent: $\theta \leftarrow \theta - \alpha \nabla_{\theta} l(f_{\theta}(x), y)$

$f \cdot g \cdot h$

$f(g(h(x)))$

Chain Rule

$$\left. \frac{\partial f(g(h(x)))}{\partial x} \right|_x = \left. \frac{\partial f(g(h))}{\partial h} \right|_h \left. \frac{\partial h(x)}{\partial x} \right|_x = \underbrace{\left. \frac{\partial f(g)}{\partial g} \right|_g}_{\text{green}} \underbrace{\left. \frac{\partial g(h)}{\partial h} \right|_h}_{\text{red}} \underbrace{\left. \frac{\partial h(x)}{\partial x} \right|_x}_{\text{red}}$$

$$l(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$\frac{\partial l}{\partial \hat{y}_i} = -\frac{1}{n} 2(y_i - \hat{y}_i)$$

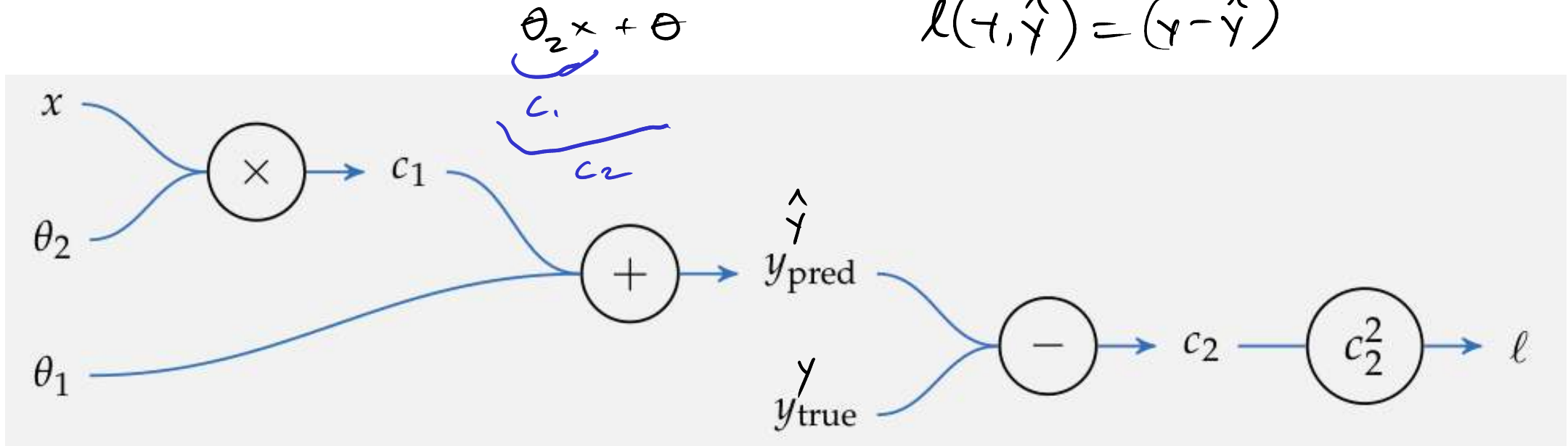
$$\hat{y} = \underline{W^{(2)}} \sigma(\underline{W^{(1)}} x + \underline{b^{(1)}}) + \underline{b^{(2)}}$$
$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial \hat{y}} \left(\frac{\partial \hat{y}}{\partial W^{(2)}} \right)^T = \frac{\partial l}{\partial \hat{y}} \underbrace{\sigma(W^{(1)} x + b^{(1)})^T}_{\text{blue bracket}}$$

$$\underline{W^{(2)}} \leftarrow \underline{W^{(2)}} - \alpha \frac{\partial l}{\partial W^{(2)}}$$

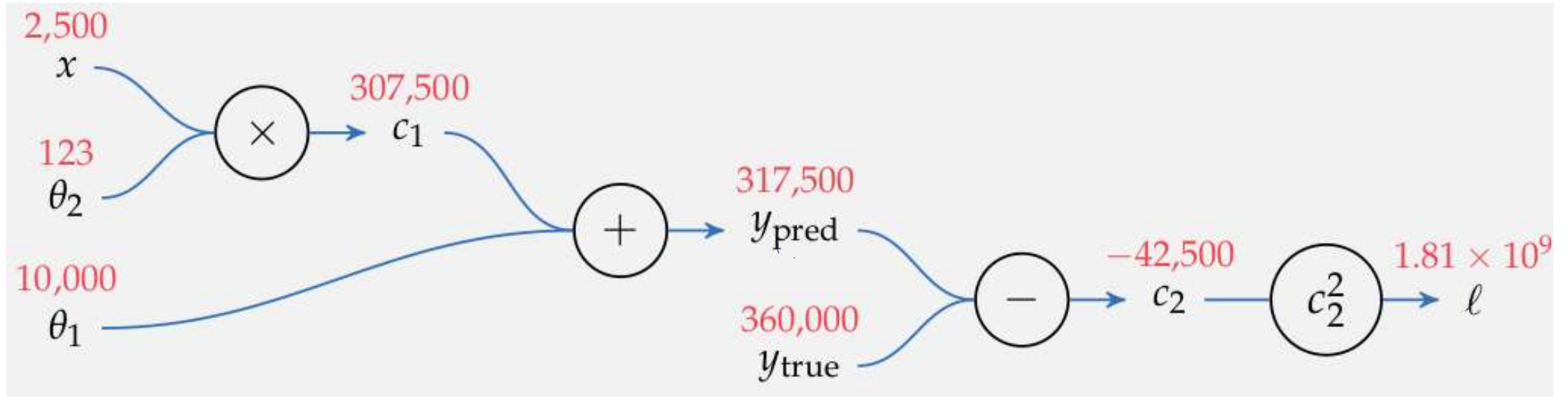
Backprop

Backprop

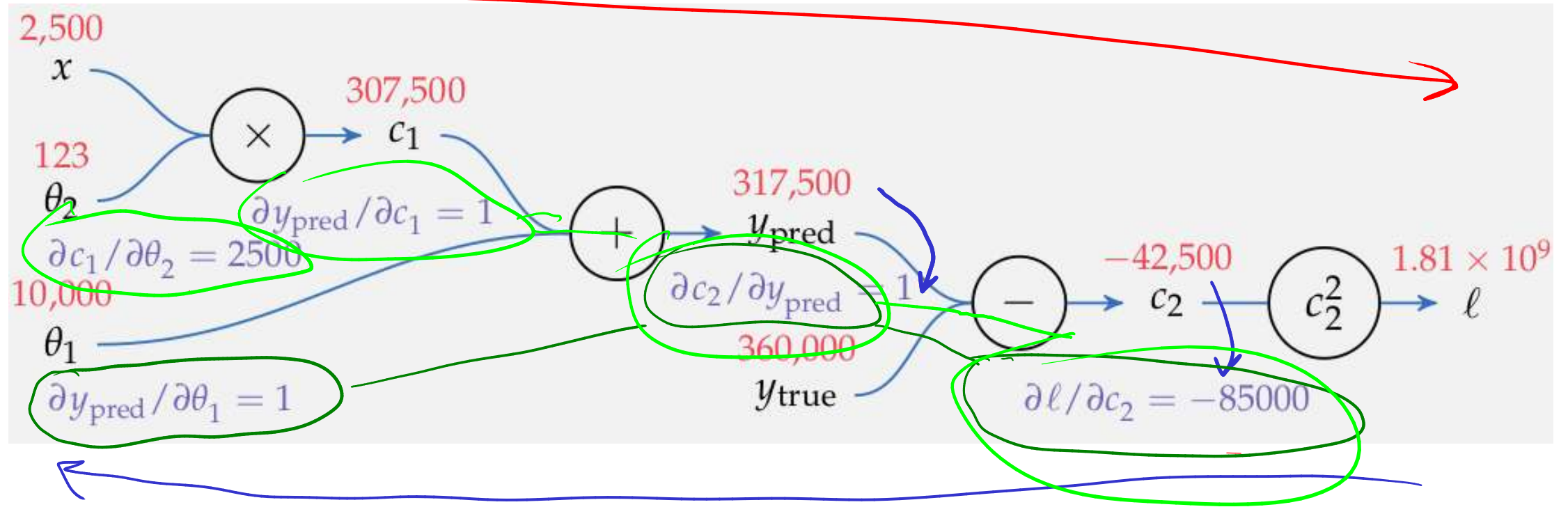
$$\ell(y, \hat{y}) = (y - \hat{y})^2$$



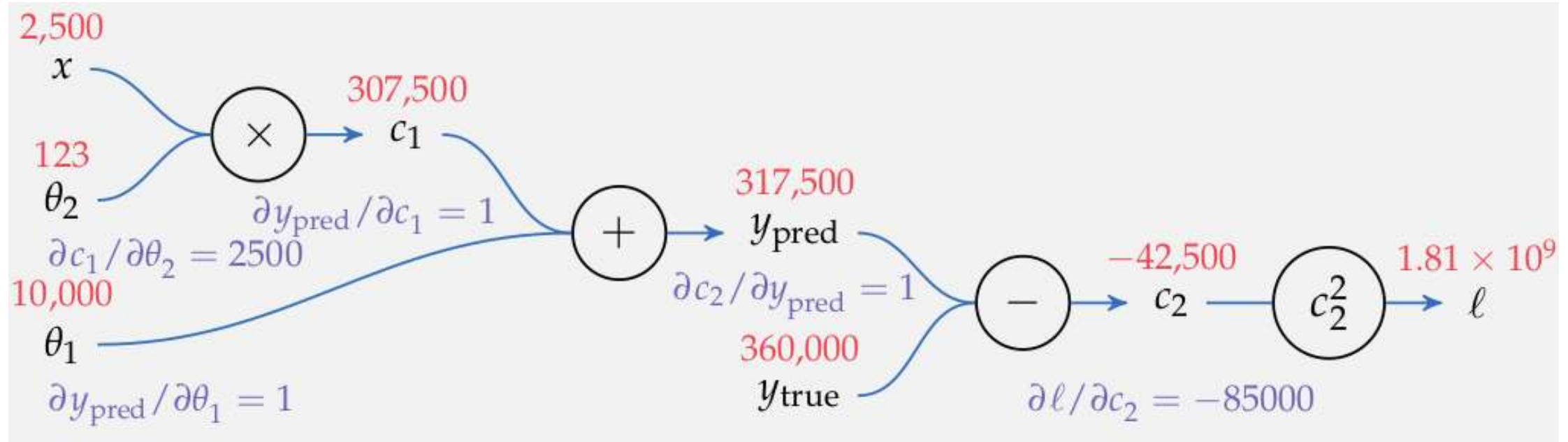
Backprop



Backprop



Backprop



$$\frac{\partial \ell}{\partial \theta_1} = \frac{\partial \ell}{\partial c_2} \frac{\partial c_2}{\partial y_{\text{pred}}} \frac{\partial y_{\text{pred}}}{\partial \theta_1} = -85,000 \cdot 1 \cdot 1 = -85,000$$

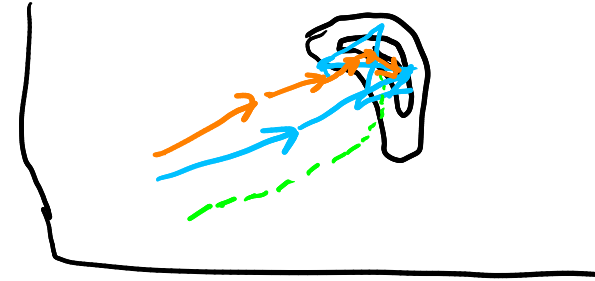
$$\frac{\partial \ell}{\partial \theta_2} = \frac{\partial \ell}{\partial c_2} \frac{\partial c_2}{\partial y_{\text{pred}}} \frac{\partial y_{\text{pred}}}{\partial c_1} \frac{\partial c_1}{\partial \theta_2} = -85,000 \cdot 1 \cdot 1 \cdot 2500 = -2.125 \times 10^8$$

a “fast and furious” approach to training neural networks does not work and only leads to suffering. Now, suffering is a perfectly natural part of getting a neural network to work well, but it can be mitigated by being thorough, defensive, paranoid, and obsessed with visualizations of basically every possible thing. The qualities that in my experience correlate most strongly to success in deep learning are patience and attention to detail.

- Andrej Karpathy

Adaptive Step Size: RMSProp

$$\theta = \theta + \alpha \widehat{\nabla_{\theta} f_{\theta}(x)}$$



$$\hat{s}^{(k+1)} = \gamma \hat{s}^{(k)} + (1-\gamma)(g^{(k)} \odot g^{(k)})$$

↖ elementwise product

$$x_i^{(k+1)} = x_i^{(k)} - \frac{\alpha}{\epsilon + \sqrt{s_i^{(k+1)}}} g_i^{(k)}$$

Adaptive Step Size: ADAM

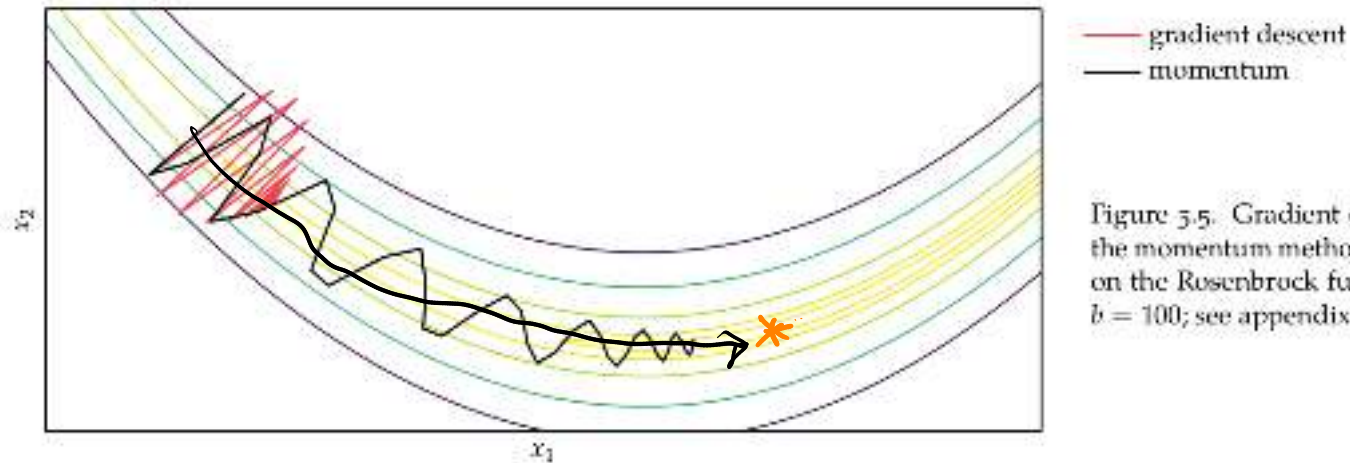


Figure 3.5. Gradient descent and the momentum method compared on the Rosenbrock function with $b = 100$; see appendix B.6.

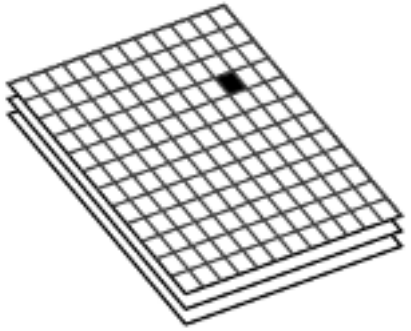
biased decaying moment
biased decaying sq. gradient
corrected d. m.
corrected sq. gradient

$$\begin{aligned} \hat{v}^{(k+1)} &= \gamma_v \hat{v}^{(k)} + (1 - \gamma_v) g^{(k)} \\ \hat{s}^{(k+1)} &= \gamma_s \hat{s}^{(k)} + (1 - \gamma_s) (g^{(k)} \odot g^{(k)}) \\ \hat{v}^{(k+1)} &= \hat{v}^{(k+1)} / (1 - \gamma_v^k) \\ \hat{s} &= \hat{s} / (1 - \gamma_s^k) \end{aligned}$$

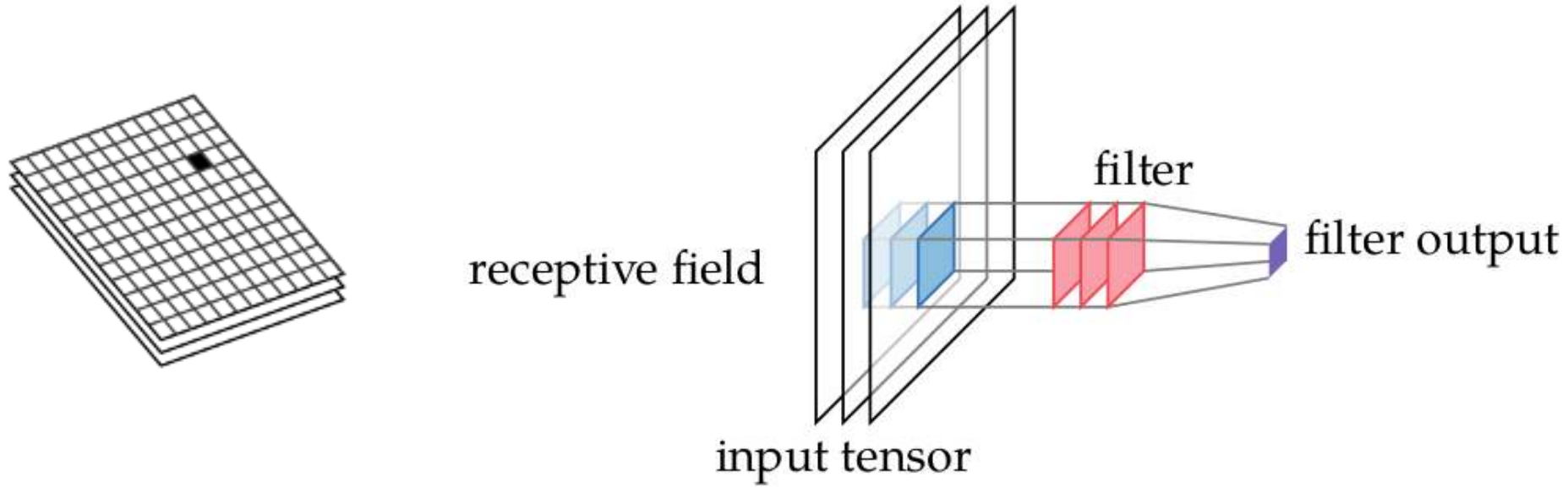
$$x^{(k+1)} = x^{(k)} + \alpha \hat{v}^{(k+1)} / \left(\epsilon + \sqrt{\hat{s}^{(k+1)}} \right)$$

On Your Radar: ConvNets

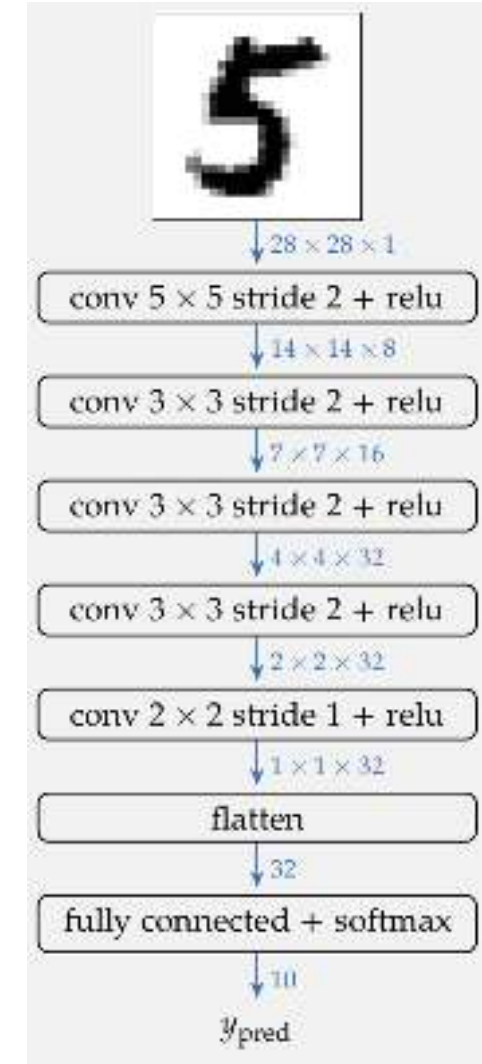
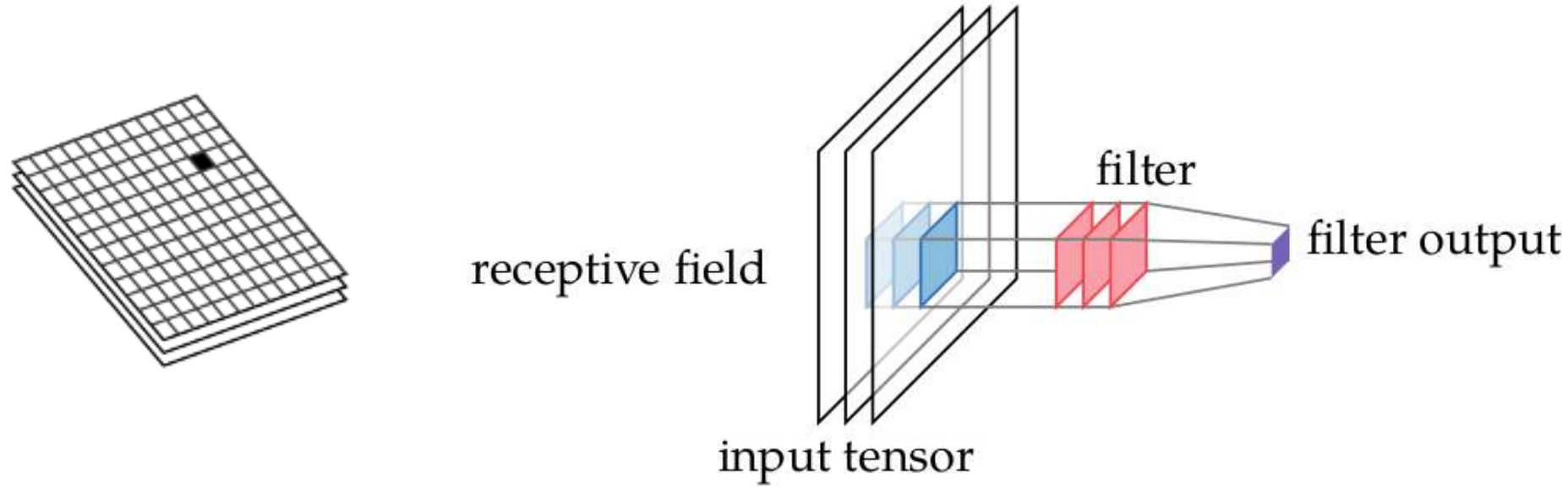
On Your Radar: ConvNets



On Your Radar: ConvNets



On Your Radar: ConvNets



On Your Radar: Regularization

On Your Radar: Regularization

$$\arg \min_{\boldsymbol{\theta}} \sum_{(x,y) \in \mathbf{D}} \ell(f_{\boldsymbol{\theta}}(x), y) - \beta \|\boldsymbol{\theta}\|^2$$

On Your Radar: Regularization

$$\arg \min_{\boldsymbol{\theta}} \sum_{(x,y) \in \mathbf{D}} \ell(f_{\boldsymbol{\theta}}(x), y) - \beta \|\boldsymbol{\theta}\|^2$$

e.g. Batch norm, dropout