Back-propagation

CPT_S 434/534 Neural network design and application

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \ \rightarrow 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

- Matrix multiplication:

$$\boxed{XX': \mathrm{d}\times\mathrm{n}\times\mathrm{d}} \quad \boxed{XY: \mathrm{d}\times\mathrm{n}} \quad \boxed{(XX')^{-1}XY: d\times d\times n} \quad \rightarrow O(d^2 n)$$

- Inverse of a matrix:

$$(XX')^{-1}:\ O(d^{2.373})$$

- Total complexity

$$O(d^2 n + d^{2.373})$$

Optimization for machine learning

- First-order algorithms (commonly used and researched in machine learning)
  - Gradient descent
  - Momentum methods
  - Stochastic variants
  - Hessian vector products
  - ……

Optimization for machine learning

- First-order algorithms (commonly used and researched in machine learning)
  - Gradient descent
  - Momentum methods
  - Stochastic variants
  - Hessian vector products
  - ……

First-order → need to compute gradients

Optimization for machine learning

- First-order algorithms (commonly used and researched in machine learning)
  - Gradient descent
  - Momentum methods
  - Stochastic variants
  - Hessian vector products
  - ……

$$\boxed{h_S^{\mathrm{ERM}} = \operatorname*{argmin}_{h \in \mathcal{H}} \widehat{R}_S(h).}$$

First-order → need to compute gradients

Optimization for machine learning

- First-order algorithms (commonly used and researched in machine learning)
  - Gradient descent
  - Momentum methods
  - Stochastic variants
  - Hessian vector products
  - ……

$$h_S^{\mathrm{ERM}} = \operatorname*{argmin}_{h \in \mathcal{H}} \widehat{R}_S(h).$$

$$\min_{w_1,\ldots,w_K} \frac{1}{n} \sum_{i=1}^{n} \underbrace{\left(- \sum_{k=1}^{K} y_{ik} \cdot \log(f(w_k; x_i)) \right)}_{\triangleq L_i(W) \text{ (see Section 2)}},$$

First-order → need to compute gradients

Optimization for machine learning

- First-order algorithms (commonly used and researched in machine learning)
  - Gradient descent
  - Momentum methods
  - Stochastic variants
  - Hessian vector products
  - ……

$$\boxed{h_S^{\mathrm{ERM}} = \operatorname*{argmin}_{h \in \mathcal{H}} \widehat{R}_S(h).}$$

$$\downarrow$$

$$\boxed{\min_{w_1,\ldots,w_K} \frac{1}{n} \sum_{i=1}^{n} \underbrace{\left(- \sum_{k=1}^{K} y_{ik} \cdot \log(f(w_k; x_i)) \right)}_{\triangleq L_i(W) \text{ (see Section } \boxed{2}\text{)}},}$$

$$\downarrow$$

$$\boxed{\frac{\partial F}{\partial w_k} = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial L_i}{w_k} + \lambda w_k.}$$

Gradients for updating

First-order → need to compute gradients

Determining model parameters

Analytical solution
$O(d^2 n + d^{2.373})$

$dn \gg b/\epsilon$

$\epsilon = O(1/\sqrt{n})$

- Stochastic gradient descent (SGD)

Randomly sample b data

b>=1

$$\nabla_w f(w) = \boxed{\frac{1}{b}\sum_{i=1}^{b}} x_i' w x_i - y_i x_i \rightarrow O(db)$$

$$w_{t+1} = w_t - \alpha_t \nabla_w f(w_t) \rightarrow O(d)$$

An iterative algorithm

$O(db\frac{1}{\epsilon})$

Theorem 5 *Set the parameters $T_1 = 4$ and $\eta_1 = \frac{1}{\lambda}$ in the* Epoch-GD *algorithm. The final point $\mathbf{x}_1^k$ returned by the algorithm has the property that*

$$\mathbb{E}[F(\mathbf{x}_1^k)] - F(\mathbf{x}^\star) \leq \frac{16G^2}{\lambda T}. \quad \boxed{= \epsilon(T) \rightarrow T = O(\tfrac{1}{\epsilon})}$$

The total number of gradient updates is at most T.

$O(dbT)$

Hazan, Elad, and Satyen Kale. "Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimization." *The Journal of Machine Learning Research* 15, no. 1 (2014): 2489-2512.

Determining model parameters

Analytical solution
$O(d^2 n + d^{2.373})$

- Stochastic gradient descent (SGD)

Randomly sample b data

$b>=1$

$$\underbrace{\boxed{\nabla_w f(w)}}_{\text{gradients}} = \boxed{\frac{1}{b}\sum_{i=1}^{b}} x_i' w x_i - y_i x_i \rightarrow O(db)$$

$dn \gg b/\epsilon$

$\epsilon = O(1/\sqrt{n})$

$$w_{t+1} = \underbrace{\boxed{w_t - \alpha_t \nabla_w f(w_t)}}_{\text{gradient descent}} \rightarrow O(d)$$

An iterative algorithm

$O(db\frac{1}{\epsilon})$

Theorem 5 *Set the parameters $T_1 = 4$ and $\eta_1 = \frac{1}{\lambda}$ in the* EPOCH-GD *algorithm. The final point $\mathbf{x}_1^k$ returned by the algorithm has the property that*

$$\mathbb{E}[F(\mathbf{x}_1^k)] - F(\mathbf{x}^\star) \leq \frac{16G^2}{\lambda T}. \boxed{= \epsilon(\mathrm{T}) \rightarrow T = O(\tfrac{1}{\epsilon})}$$

The total number of gradient updates is at most T.

$O(dbT)$

Hazan, Elad, and Satyen Kale. "Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimization." *The Journal of Machine Learning Research* 15, no. 1 (2014): 2489-2512.

How to compute gradient?

$$f(x) \rightarrow ?$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

How to compute gradient?

$$f(x) \to \nabla f(x) = \frac{df}{dx}$$

What if we have composition structure?
f and g both have their own parameters
x is the parameter of function g

$$f(g(x)) \to ?$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

What if we have composition structure?
f and g both have their own parameters
x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boldsymbol{h}^{(2)} = g^{(2)}\left(\boldsymbol{W}^{(2)\top}\boldsymbol{h}^{(1)} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top} \boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top} \boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$f(g(x)) \rightarrow ?$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

- Chain rule of calculus

$$y = g(x) \text{ and } z = f(g(x)) = f(y)$$

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus

$$y = g(x) \text{ and } z = f(g(x)) = f(y)$$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$f(g(x)) \rightarrow ?$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

- Chain rule of calculus

$$y = g(x) \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus

$$y = g(x) \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$

$= f(g(x))$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus

$$y = g(x) \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$

$= f(g(x))$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

$f(y) = \log(y)$

$g(x) = 1 + e^{-x}$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^\top \boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^\top \boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus

$$y = g(x) \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$
$= f(g(x))$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

$f(y) = \log(y)$

$\nabla f(y) = 1/y$

$g(x) = 1 + e^{-x}$

$\nabla g(x) = -e^{-x}$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus

$$\boxed{y = g(x)} \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$
$= f(g(x))$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

$f(y) = \log(y)$

$\nabla f(y) = 1/y$

$g(x) = 1 + e^{-x}$

$\nabla g(x) = -e^{-x}$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

What if we have composition structure?
f and g both have their own parameters
x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

- Chain rule of calculus

$$\boxed{y = g(x)} \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$

$= f(g(x))$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

$f(y) = \log(y)$

$\nabla f(y) = 1/y$

$g(x) = 1 + e^{-x}$

$\nabla g(x) = -e^{-x}$

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

How to compute gradient?

$$f(x) \to \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \to ?$$

- Chain rule of calculus

$$\boxed{y = g(x)} \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$

$$= f(g(x))$$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

$f(y) = \log(y)$

$\nabla f(y) = 1/y$

$g(x) = 1 + e^{-x}$

$\nabla g(x) = -e^{-x}$

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

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus

$$\boxed{y = g(x)} \text{ and } z = f(g(x)) = f(y)$$

An example: $h(x) = \log(1 + e^{-x})$

$= f(g(x))$

$$\frac{dz}{dx} = \boxed{\frac{dz}{dy}}\boxed{\frac{dy}{dx}}$$

$f(y) = \log(y)$

$\nabla f(y) = 1/y$

$g(x) = 1 + e^{-x}$

$\nabla g(x) = -e^{-x}$

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

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\left(f_2(f_1(x))\right)\right) \rightarrow ?$$

How to compute gradient?

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$
$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$
$$\vdots$$

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

What if we have composition structure?
f and g both have their own parameters
x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\ldots\left(f_2\left(\underbrace{f_1(x)}_{x_1}\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\left(f_2\left(\underbrace{f_1(x)}_{x_1}\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$f_n\left(\dots\left(f_2(x_1)\right)\right) \rightarrow ?$$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\underbrace{\left(f_2\left(f_1(x)\right)\right)}_{x_2}\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

How to compute gradient?

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

$$f(x) \to \nabla f(x) = \frac{df}{dx}$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \to ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\ldots\left(f_2(f_1(x))\right)\right) \to ?$$

$$f_i \to x_i$$

$$\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \cdots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}$$

How to compute gradient?

$$f(x) \to \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \to ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \to ?$$

$$f_i \to x_i$$

$$\boxed{\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \dots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}}$$

Q: is $\frac{dx_n}{dx_1}$ enough to update the model (a lot of layers)?

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\ldots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$\boxed{\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \ldots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}}$$

Q: is $\frac{dx_n}{dx_1}$ enough to update the model (a lot of layers)?

NO. There are parameters to be determined in each layer

How to compute gradient?

$$f(x) \to \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \to ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\left(f_2(f_1(x))\right)\right) \to ?$$

$$f_i \to x_i$$

$$\boxed{\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \dots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}}$$

Q: is $\frac{dx_n}{dx_1}$ enough to update the model (a lot of layers)?

NO. There are parameters to be determined in each layer

We still need $\frac{dx_n}{dx_i}, for\ i = 1, \dots, n-1$

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\ldots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \ldots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}$$

Q: gradient at other hidden layers?

How to compute gradient?

$$f(x) \rightarrow \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \rightarrow ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \dots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}$$

Q: gradient at other hidden layers?

$$\frac{dx_n}{dx_i}$$

How to compute gradient?

$$f(x) \to \nabla f(x) = \frac{df}{dx}$$

$$\boxed{\boldsymbol{h}^{(1)}} = g^{(1)}\left(\boxed{\boldsymbol{W}^{(1)}}^{\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right);$$

$$\boxed{\boldsymbol{h}^{(2)}} = g^{(2)}\left(\boxed{\boldsymbol{W}^{(2)}}^{\top}\boxed{\boldsymbol{h}^{(1)}} + \boldsymbol{b}^{(2)}\right)$$

$$\vdots$$

What if we have composition structure?

f and g both have their own parameters

x is the parameter of function g

$$f(g(x)) \to ?$$

- Chain rule of calculus (generalize to multi-dimensional cases)

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \to ?$$

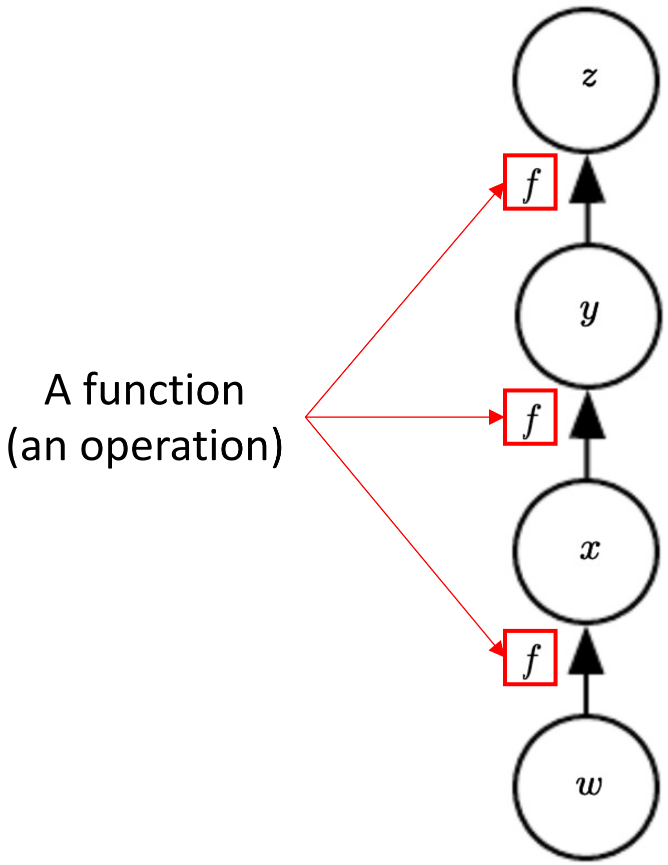
$$f_i \to x_i$$

$$\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \dots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}$$

Q: gradient at other hidden layers?

$$\frac{dx_n}{dx_i} = \frac{dx_n}{dx_{n-1}} \cdot \frac{dx_{n-1}}{dx_{n-2}} \cdot \dots \cdot \frac{dx_{i+1}}{dx_i}$$

Computation graphs



Computation graphs

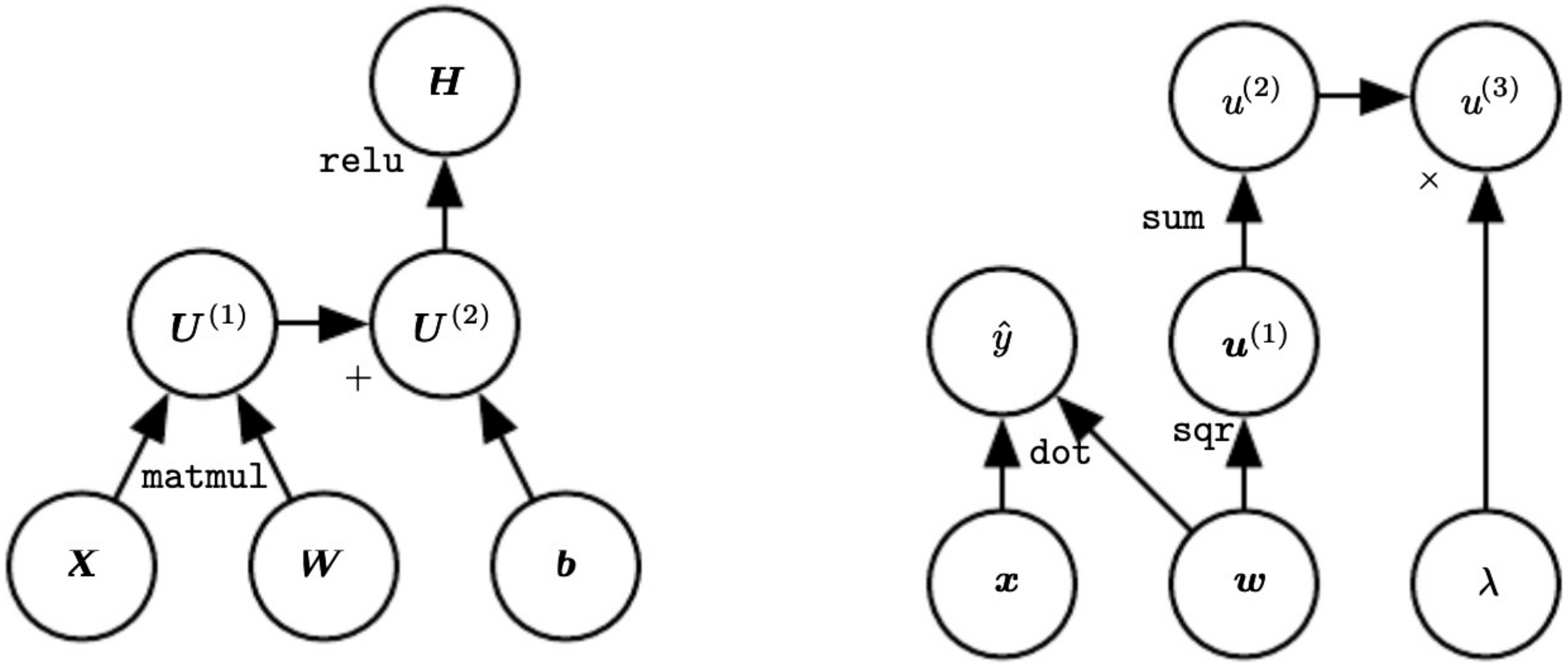
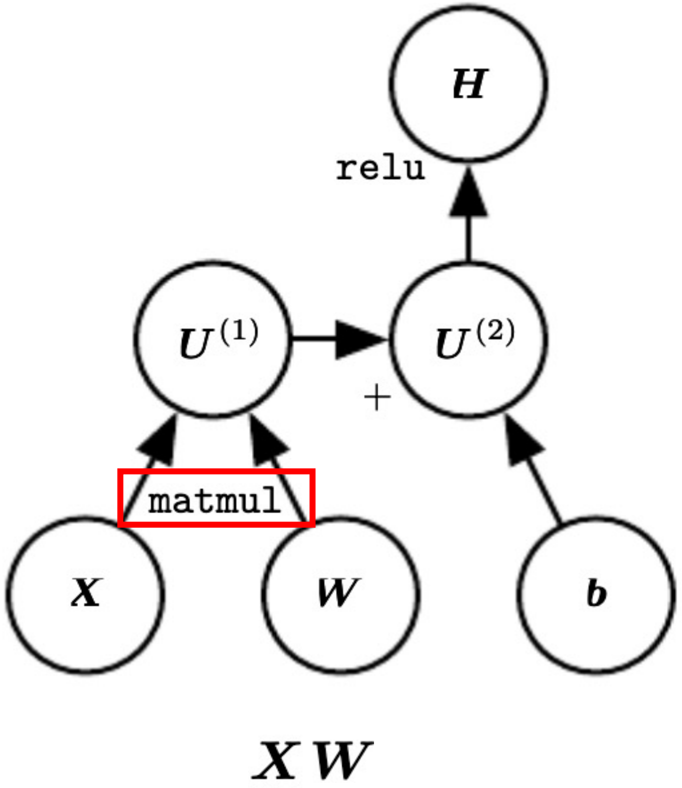


Figure 6.8
“Deep Learning”

It is a precise language to describe structure of operations in neural networks

Computation graphs



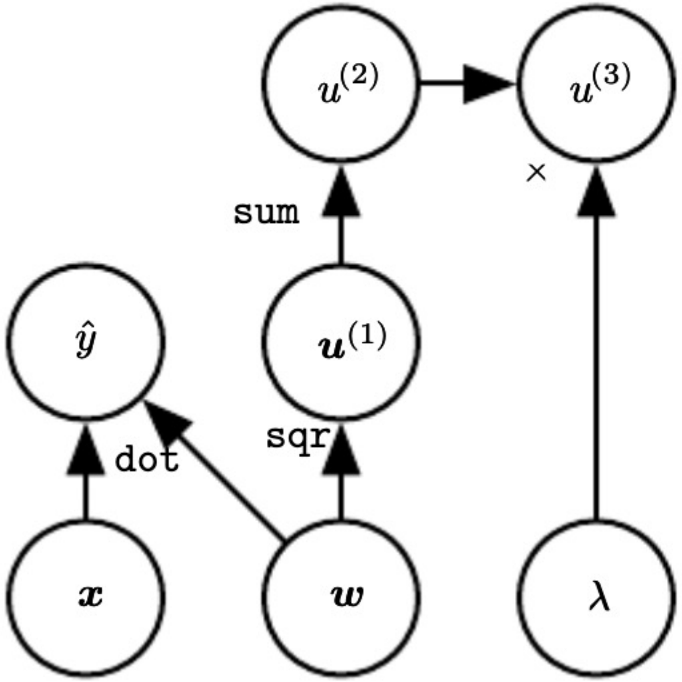


Figure 6.8
“Deep Learning”

It is a precise language to describe structure of operations in neural networks

Computation graphs

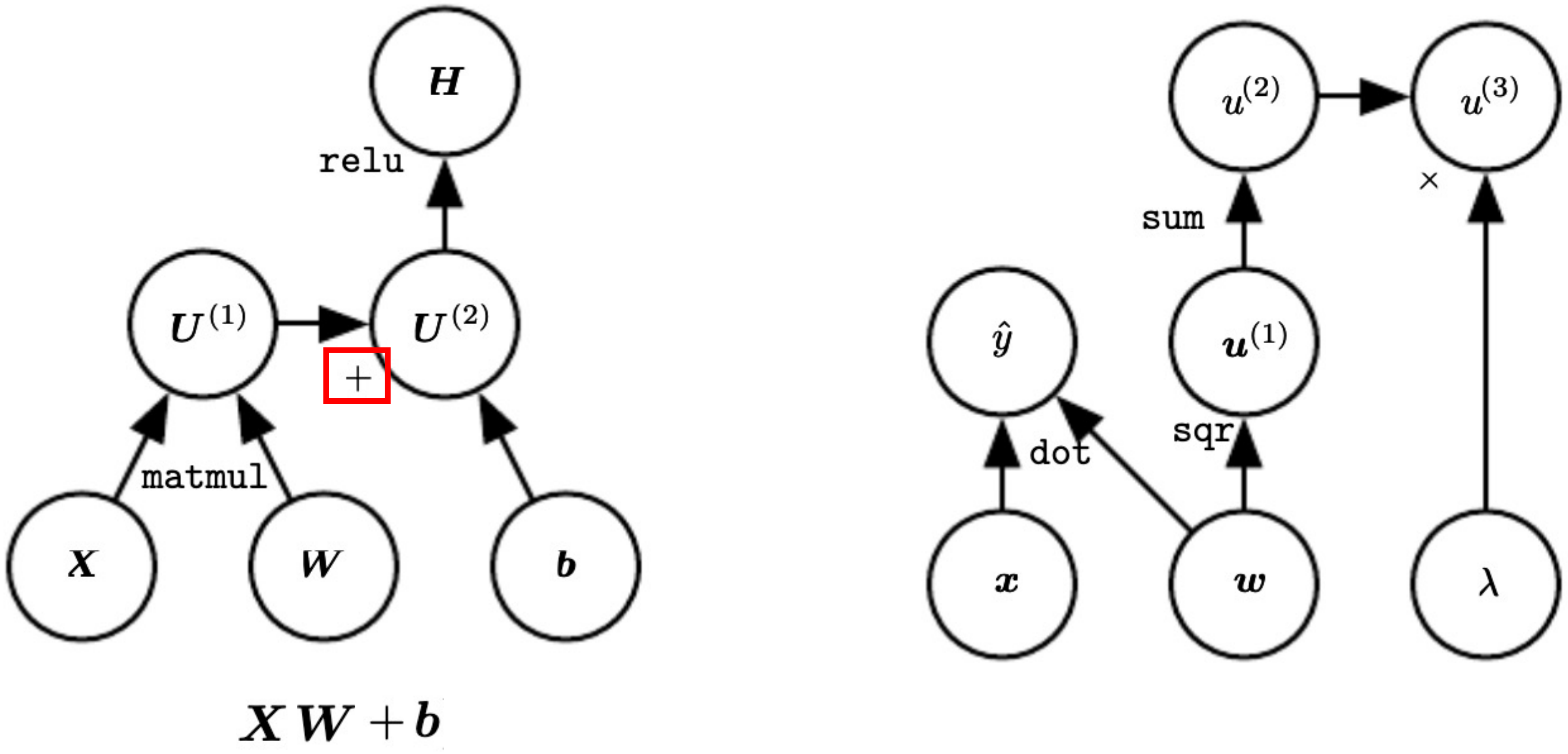
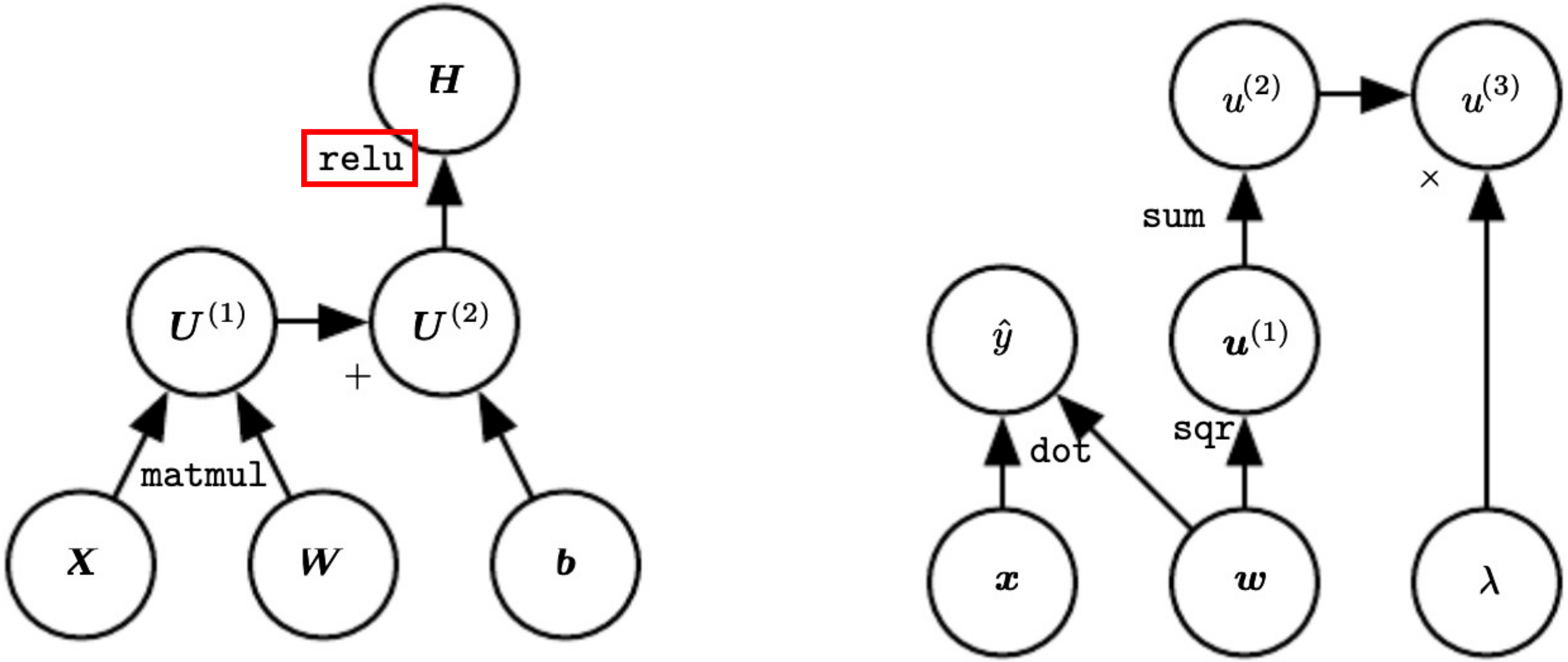


Figure 6.8
“Deep Learning”

It is a precise language to describe structure of operations in neural networks

Computation graphs

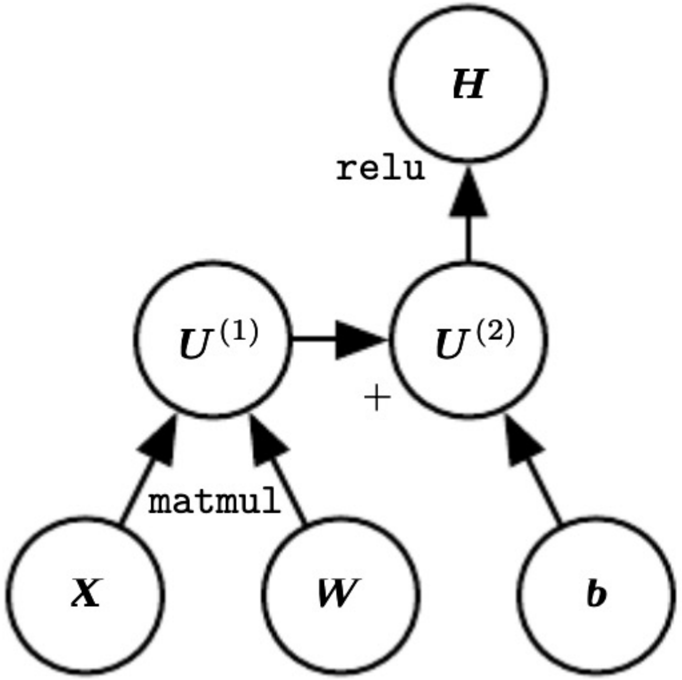


$$\max\{XW + b, 0\}$$

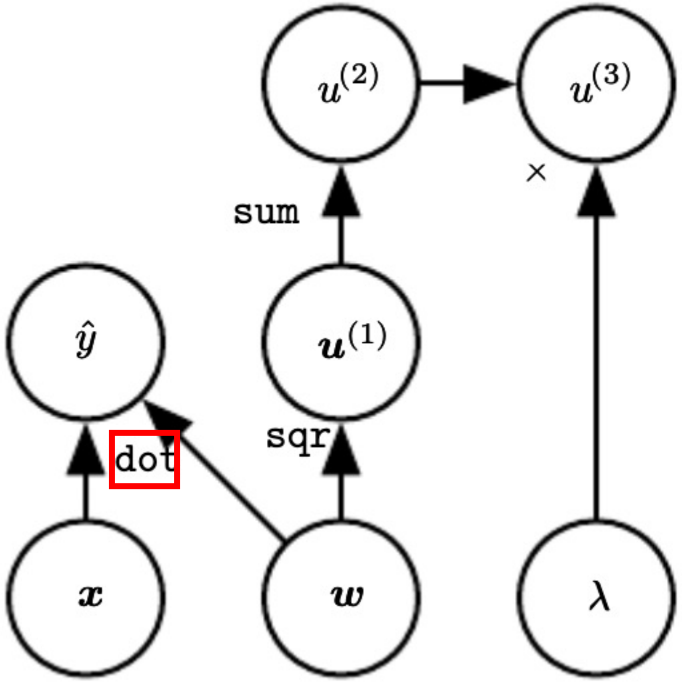
Figure 6.8
“Deep Learning”

It is a precise language to describe structure of operations in neural networks

Computation graphs



$\max\{XW + b, 0\}$

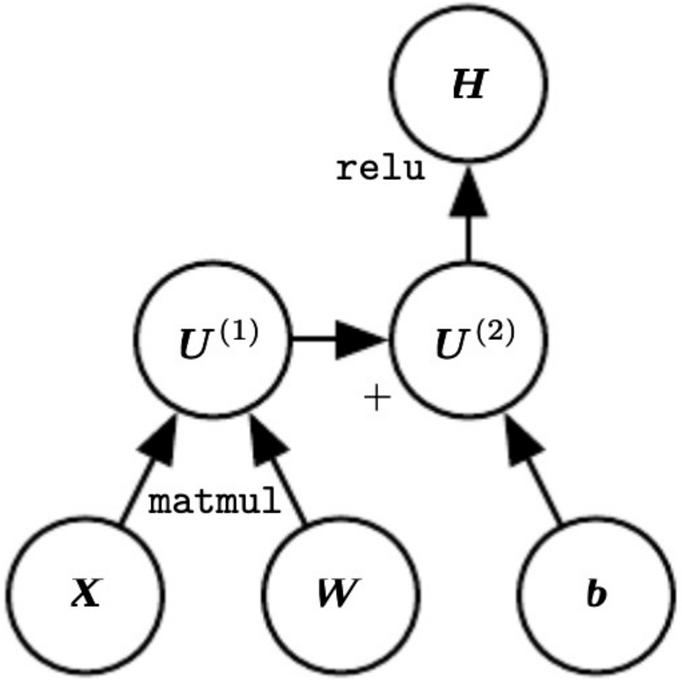


$x^\top w \longrightarrow \hat{y}$

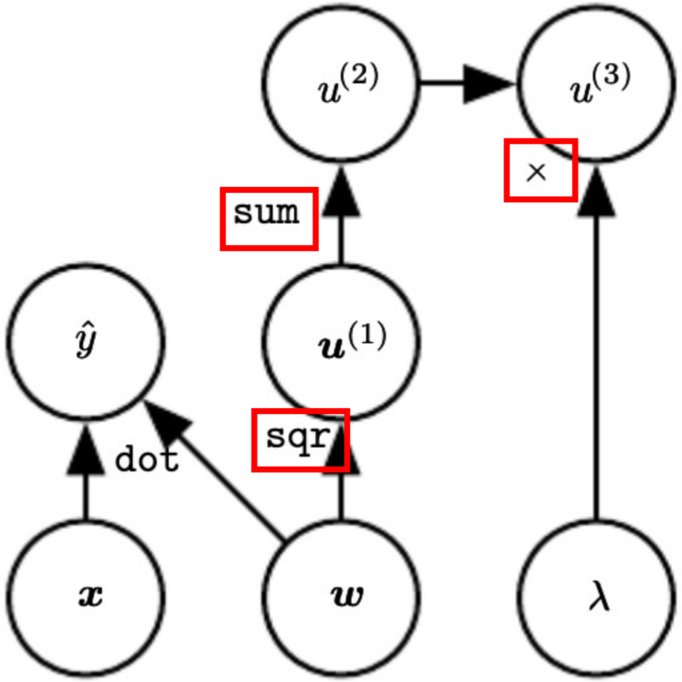
Figure 6.8
"Deep Learning"

It is a precise language to describe structure of operations in neural networks

Computation graphs



$\max\{XW + b, 0\}$

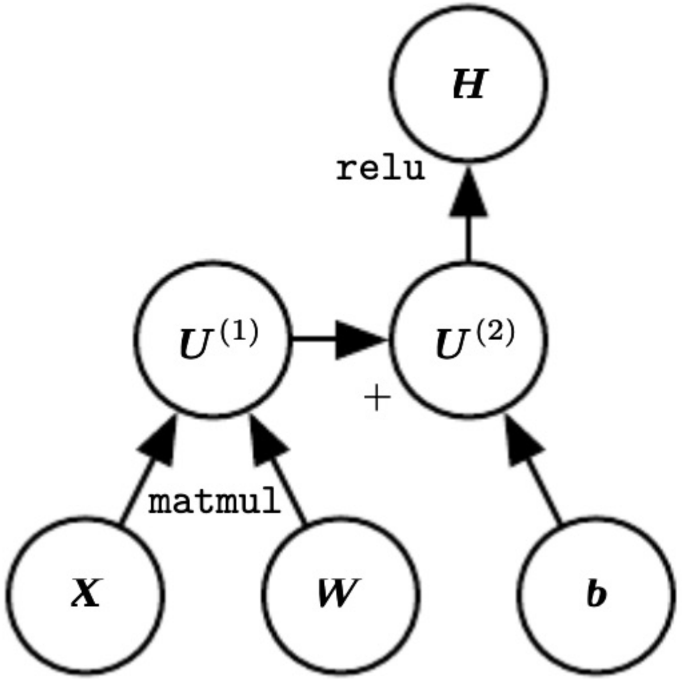


$$\begin{cases} x^\top w \longrightarrow \hat{y} \\ \boxed{\lambda \sum_i w_i^2} \end{cases}$$

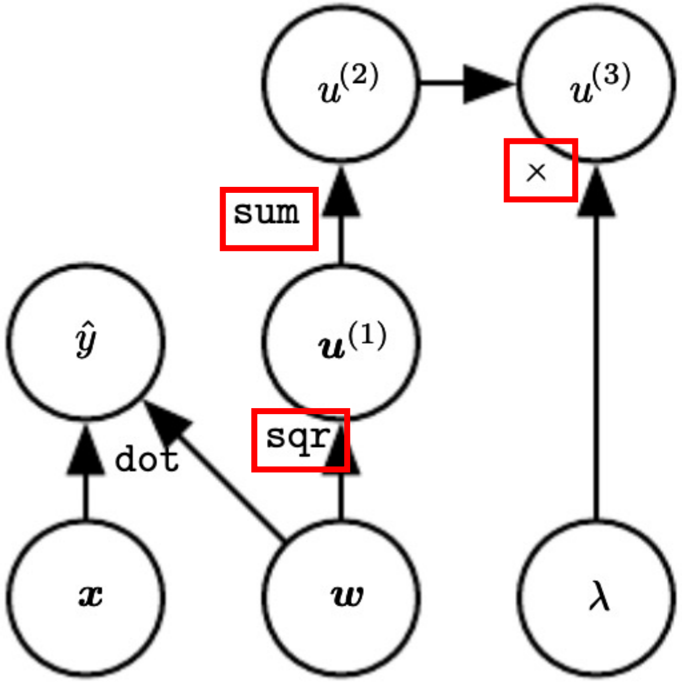
Figure 6.8
"Deep Learning"

It is a precise language to describe structure of operations in neural networks

Computation graphs



$$\max\{\boldsymbol{X}\boldsymbol{W}+\boldsymbol{b},0\}$$



$$\begin{cases} \boldsymbol{x}^\top \boldsymbol{w} \longrightarrow \hat{y} \\ \boxed{\lambda \sum_i w_i^2} \end{cases}$$

Regularization

Figure 6.8
“Deep Learning”

It is a precise language to describe structure of operations in neural networks

Forward propagation

Require: Network depth, l
Require: $\boldsymbol{W}^{(i)}, i \in \{1, \ldots, l\}$, the weight matrices of the model
Require: $\boldsymbol{b}^{(i)}, i \in \{1, \ldots, l\}$, the bias parameters of the model
Require: $\boxed{\boldsymbol{x}\text{, the input to process}}$
Require: $\boldsymbol{y}$, the target output

$\boldsymbol{h}^{(0)} = \boldsymbol{x}$
for $k = 1, \ldots, l$ **do**
  $\boldsymbol{a}^{(k)} = \boldsymbol{b}^{(k)} + \boldsymbol{W}^{(k)}\boldsymbol{h}^{(k-1)}$
  $\boldsymbol{h}^{(k)} = f(\boldsymbol{a}^{(k)})$
end for
$\hat{\boldsymbol{y}} = \boldsymbol{h}^{(l)}$
$J = L(\hat{\boldsymbol{y}}, \boldsymbol{y}) + \lambda\Omega(\theta)$

$$h^{(l)}\left(\ldots h^{(3)}\left(h^{(2)}\left(h^{(1)}(x)\right)\right)\right)$$

Algorithm 6.3 in "Deep Learning"

Forward propagation

Require: Network depth, l
Require: $\boldsymbol{W}^{(i)}, i \in \{1, \ldots, l\}$, the weight matrices of the model
Require: $\boldsymbol{b}^{(i)}, i \in \{1, \ldots, l\}$, the bias parameters of the model
Require: $\boxed{\boldsymbol{x}}$, the input to process
Require: $\boldsymbol{y}$, the target output

$\boxed{\boldsymbol{h}^{(0)} = \boldsymbol{x}}$
for $k = 1, \ldots, l$ **do**
  $\boldsymbol{a}^{(k)} = \boldsymbol{b}^{(k)} + \boldsymbol{W}^{(k)} \boldsymbol{h}^{(k-1)}$
  $\boldsymbol{h}^{(k)} = f(\boldsymbol{a}^{(k)})$
end for
$\hat{\boldsymbol{y}} = \boldsymbol{h}^{(l)}$
$J = L(\hat{\boldsymbol{y}}, \boldsymbol{y}) + \lambda \Omega(\theta)$

$$h^{(l)}\left(\ldots h^{(3)}\left(h^{(2)}\left(h^{(1)}\left(h^{(0)}\right)\right)\right)\right)$$

Algorithm 6.3 in “Deep Learning”

Forward propagation

Require: Network depth, l
Require: $\boldsymbol{W}^{(i)}, i \in \{1, \ldots, l\}$, the weight matrices of the model
Require: $\boldsymbol{b}^{(i)}, i \in \{1, \ldots, l\}$, the bias parameters of the model
Require: $\boxed{\boldsymbol{x}}$, the input to process
Require: $\boldsymbol{y}$, the target output

$\quad \boxed{\boldsymbol{h}^{(0)} = \boldsymbol{x}}$
$\quad$ **for** $\boxed{k = 1}, \ldots, l$ **do**
$\quad\quad \boldsymbol{a}^{(k)} = \boldsymbol{b}^{(k)} + \boldsymbol{W}^{(k)} \boxed{\boldsymbol{h}^{(k-1)}}$
$\quad\quad \boxed{\boldsymbol{h}^{(k)}} = f(\boldsymbol{a}^{(k)})$
$\quad$ **end for**
$\quad \hat{\boldsymbol{y}} = \boldsymbol{h}^{(l)}$
$\quad J = L(\hat{\boldsymbol{y}}, \boldsymbol{y}) + \lambda \Omega(\theta)$

$$h^{(l)}\left(\ldots h^{(3)}\left(h^{(2)}\left(h^{(1)}\left(h^{(0)}\right)\right)\right)\right)$$

Algorithm 6.3 in “Deep Learning”

Forward propagation

Require: Network depth, l

Require: $\boldsymbol{W}^{(i)}, i \in \{1, \ldots, l\}$, the weight matrices of the model

Require: $\boldsymbol{b}^{(i)}, i \in \{1, \ldots, l\}$, the bias parameters of the model

Require: $\boxed{\boldsymbol{x}}$, the input to process

Require: $\boldsymbol{y}$, the target output

$\boxed{\boldsymbol{h}^{(0)} = \boldsymbol{x}}$

for $\boxed{k = 1}, \ldots, l$ **do**

$\boldsymbol{a}^{(k)} = \boldsymbol{b}^{(k)} + \boldsymbol{W}^{(k)} \boxed{\boldsymbol{h}^{(k-1)}}$

$\boxed{\boldsymbol{h}^{(k)}} = f(\boldsymbol{a}^{(k)})$

end for

$\hat{\boldsymbol{y}} = \boxed{\boldsymbol{h}^{(l)}}$

$J = L(\hat{\boldsymbol{y}}, \boldsymbol{y}) + \lambda \Omega(\theta)$

$$h^{(l)}\left(\ldots h^{(3)}\left(h^{(2)}\left(h^{(1)}\left(h^{(0)}\right)\right)\right)\right)$$

Algorithm 6.3 in “Deep Learning”

Backward propagation

After the forward computation, compute the gradient on the output layer:

$\boxed{\boldsymbol{g} \leftarrow \nabla_{\hat{\boldsymbol{y}}} J = \nabla_{\hat{\boldsymbol{y}}} L(\hat{\boldsymbol{y}}, \boldsymbol{y})}$ Gradient from loss

for $k = l, l-1, \ldots, 1$ **do**

Convert the gradient on the layer's output into a gradient on the pre-nonlinearity activation (element-wise multiplication if f is element-wise):

$$\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{a}^{(k)}} J = \boldsymbol{g} \odot f'(\boldsymbol{a}^{(k)})$$

Compute gradients on weights and biases (including the regularization term, where needed):

$$\nabla_{\boldsymbol{b}^{(k)}} J = \boldsymbol{g} + \lambda \nabla_{\boldsymbol{b}^{(k)}} \Omega(\theta)$$

$$\nabla_{\boldsymbol{W}^{(k)}} J = \boldsymbol{g}\, \boldsymbol{h}^{(k-1)\top} + \lambda \nabla_{\boldsymbol{W}^{(k)}} \Omega(\theta)$$

Propagate the gradients w.r.t. the next lower-level hidden layer's activations:

$$\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{h}^{(k-1)}} J = \boldsymbol{W}^{(k)\top}\, \boldsymbol{g}$$

end for

Algorithm 6.4 in "Deep Learning"

Backward propagation

After the forward computation, compute the gradient on the output layer:

$\boxed{\boldsymbol{g} \leftarrow \nabla_{\hat{\boldsymbol{y}}} J = \nabla_{\hat{\boldsymbol{y}}} L(\hat{\boldsymbol{y}}, \boldsymbol{y})}$ Gradient from loss

for $k = l, l-1, \ldots, 1$ **do**

Convert the gradient on the layer's output into a gradient on the pre-nonlinearity activation (element-wise multiplication if f is element-wise):

$\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{a}^{(k)}} J = \boldsymbol{g} \odot \boxed{f'(\boldsymbol{a}^{(k)})}$ Gradient from activation layer

Compute gradients on weights and biases (including the regularization term, where needed):

$\nabla_{\boldsymbol{b}^{(k)}} J = \boldsymbol{g} + \lambda \nabla_{\boldsymbol{b}^{(k)}} \Omega(\theta)$

$\nabla_{\boldsymbol{W}^{(k)}} J = \boldsymbol{g}\, \boldsymbol{h}^{(k-1)\top} + \lambda \nabla_{\boldsymbol{W}^{(k)}} \Omega(\theta)$

Propagate the gradients w.r.t. the next lower-level hidden layer's activations:

$\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{h}^{(k-1)}} J = \boldsymbol{W}^{(k)\top}\, \boldsymbol{g}$

end for

Algorithm 6.4 in "Deep Learning"

Backward propagation

After the forward computation, compute the gradient on the output layer:

$\boxed{\boldsymbol{g} \leftarrow \nabla_{\hat{\boldsymbol{y}}} J = \nabla_{\hat{\boldsymbol{y}}} L(\hat{\boldsymbol{y}}, \boldsymbol{y})}$ Gradient from loss

for $k = l, l-1, \ldots, 1$ **do**

Convert the gradient on the layer's output into a gradient on the pre-nonlinearity activation (element-wise multiplication if f is element-wise):

$\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{a}^{(k)}} J = \boldsymbol{g} \odot \boxed{f'(\boldsymbol{a}^{(k)})}$ Gradient from activation layer

Compute gradients on weights and biases (including the regularization term, where needed):

$\nabla_{\boldsymbol{b}^{(k)}} J = \boldsymbol{g} + \boxed{\lambda \nabla_{\boldsymbol{h}^{(k)}} \Omega(\theta)}$ Gradient from regularization

$\nabla_{\boldsymbol{W}^{(k)}} J = \boldsymbol{g}\, \boldsymbol{h}^{(k-1)\top} + \boxed{\lambda \nabla_{\boldsymbol{W}^{(k)}} \Omega(\theta)}$ Gradient from regularization

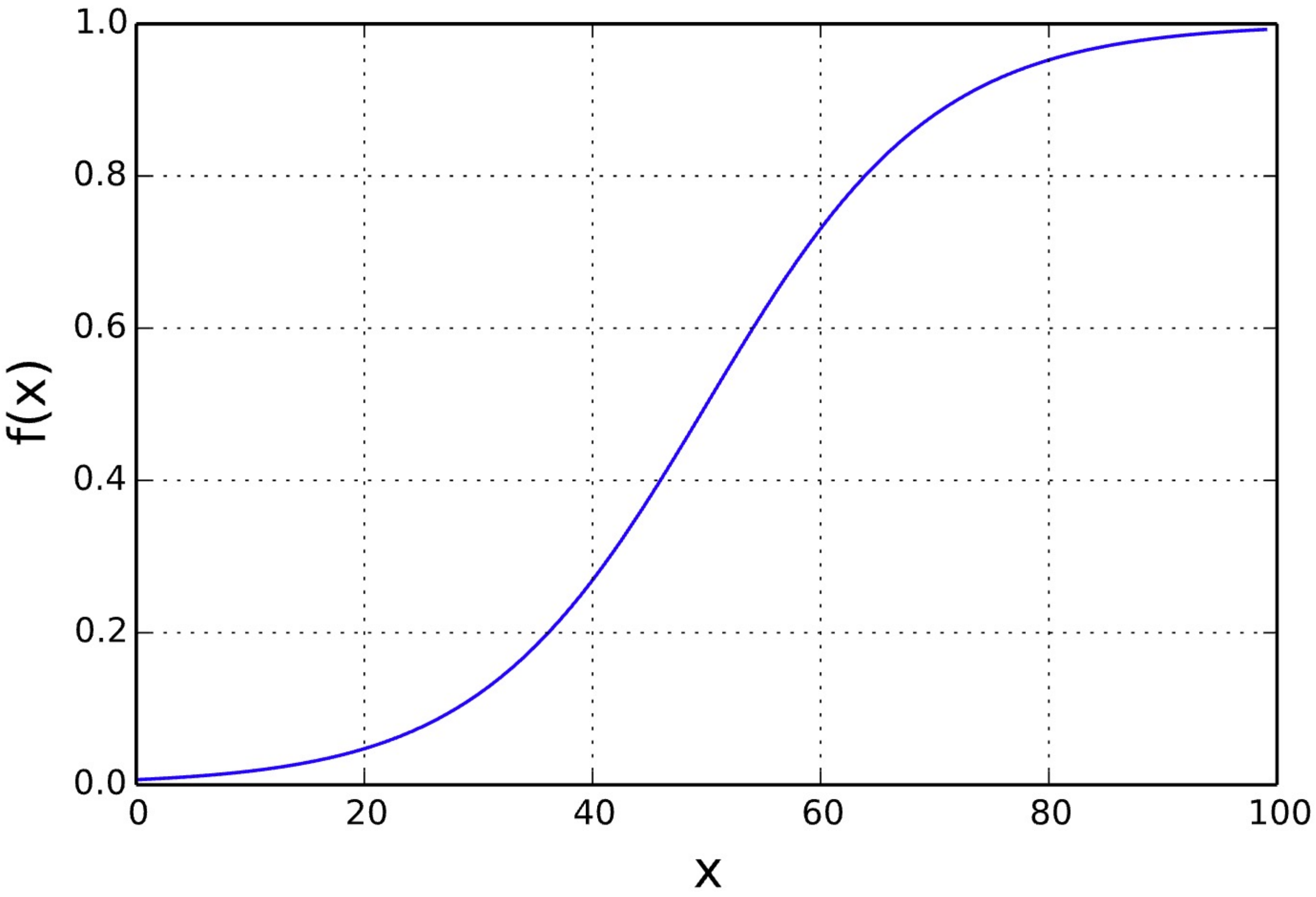
Propagate the gradients w.r.t. the next lower-level hidden layer's activations:

$\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{h}^{(k-1)}} J = \boldsymbol{W}^{(k)\top}\, \boldsymbol{g}$

end for

Algorithm 6.4 in "Deep Learning"

Gradient vanish



Sigmoid function

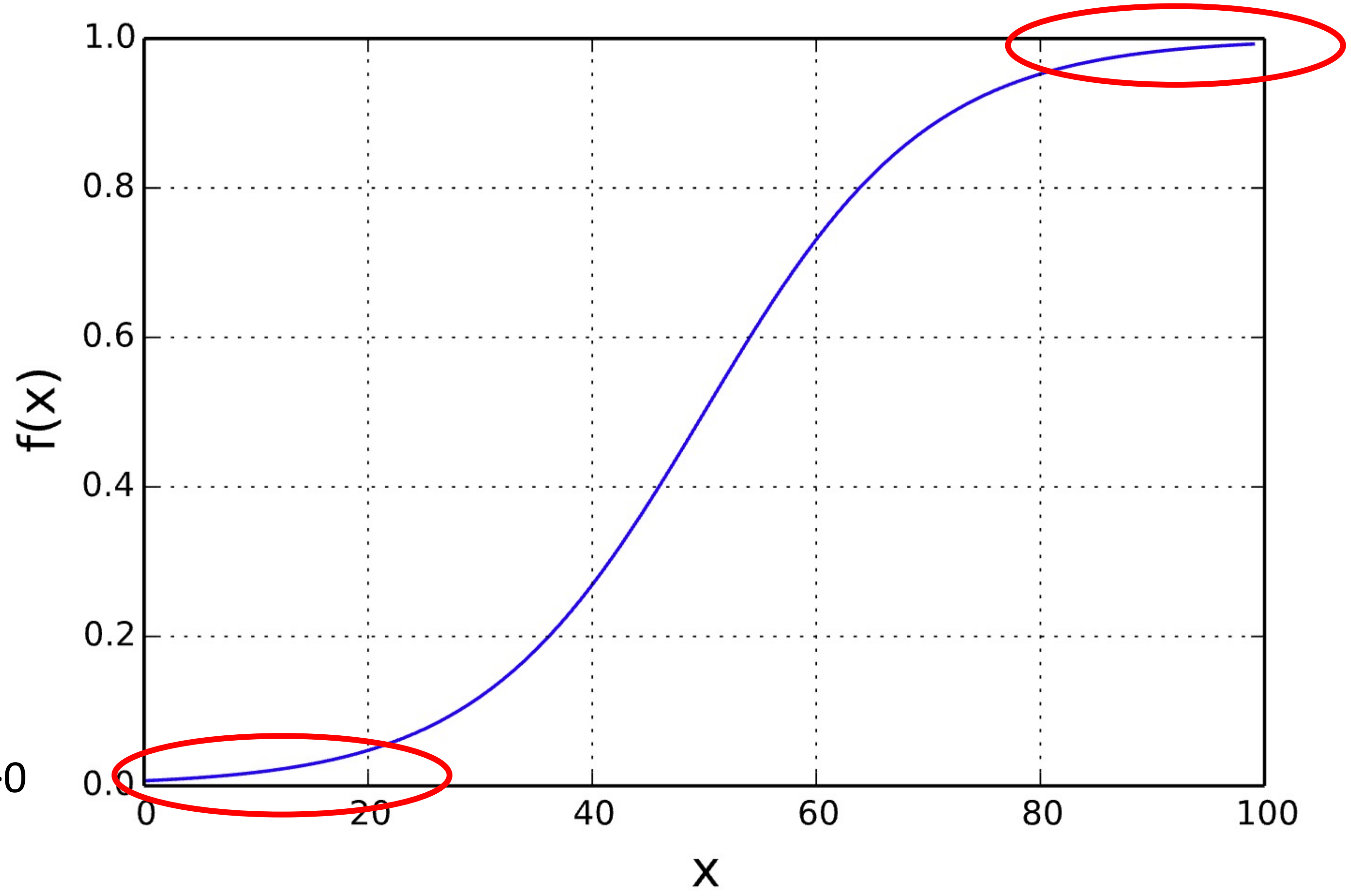
Gradient vanish

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \to ?$$

$$f_i \to x_i$$

$$\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \dots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}$$

gradients->0



gradients->0

Sigmoid function

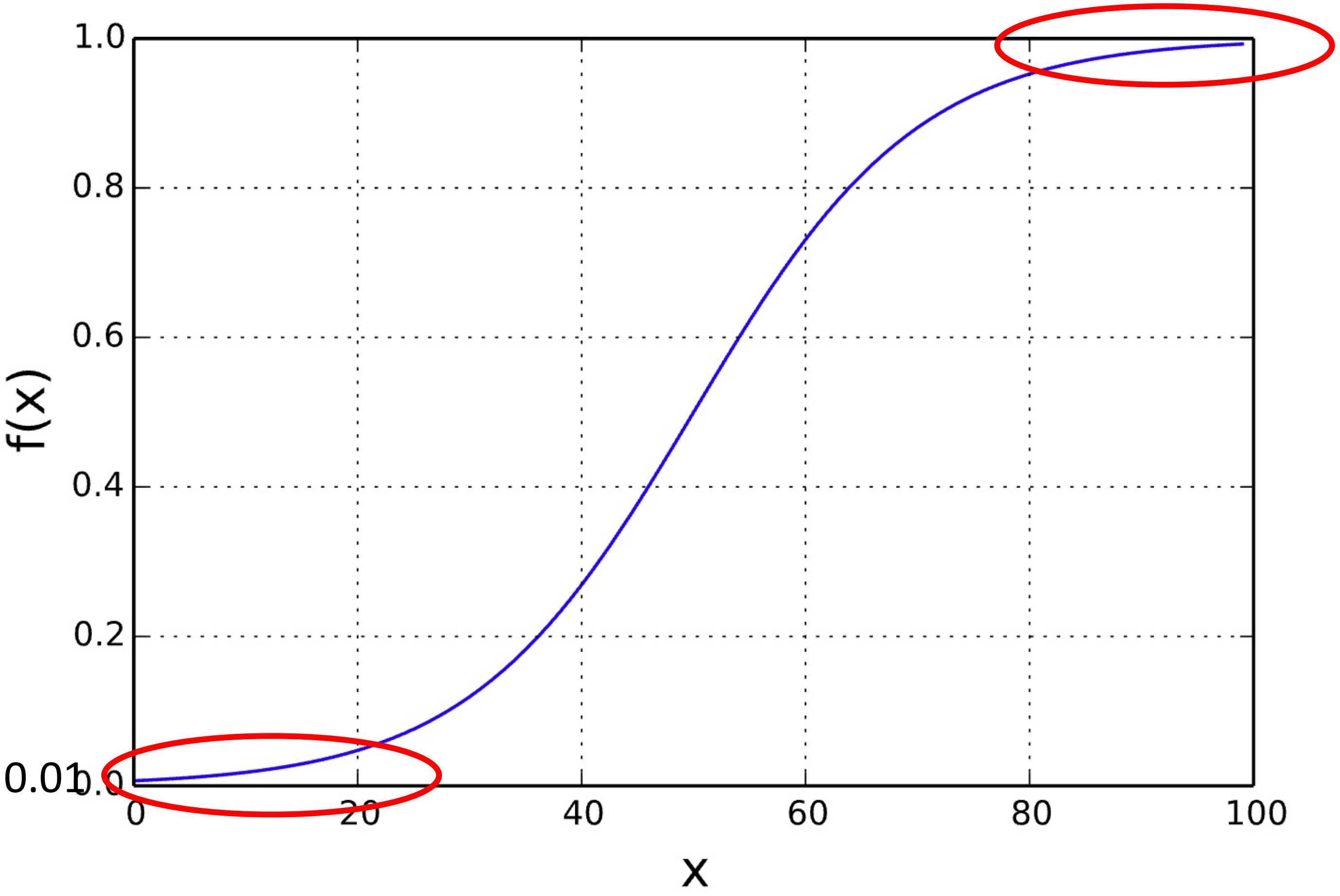
Gradient vanish

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$\frac{dx_n}{dx} = \frac{dx_n}{dx_{n-1}} \cdot \dots \cdot \frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}$$

gradients-> 0.01



gradients-> 0.01

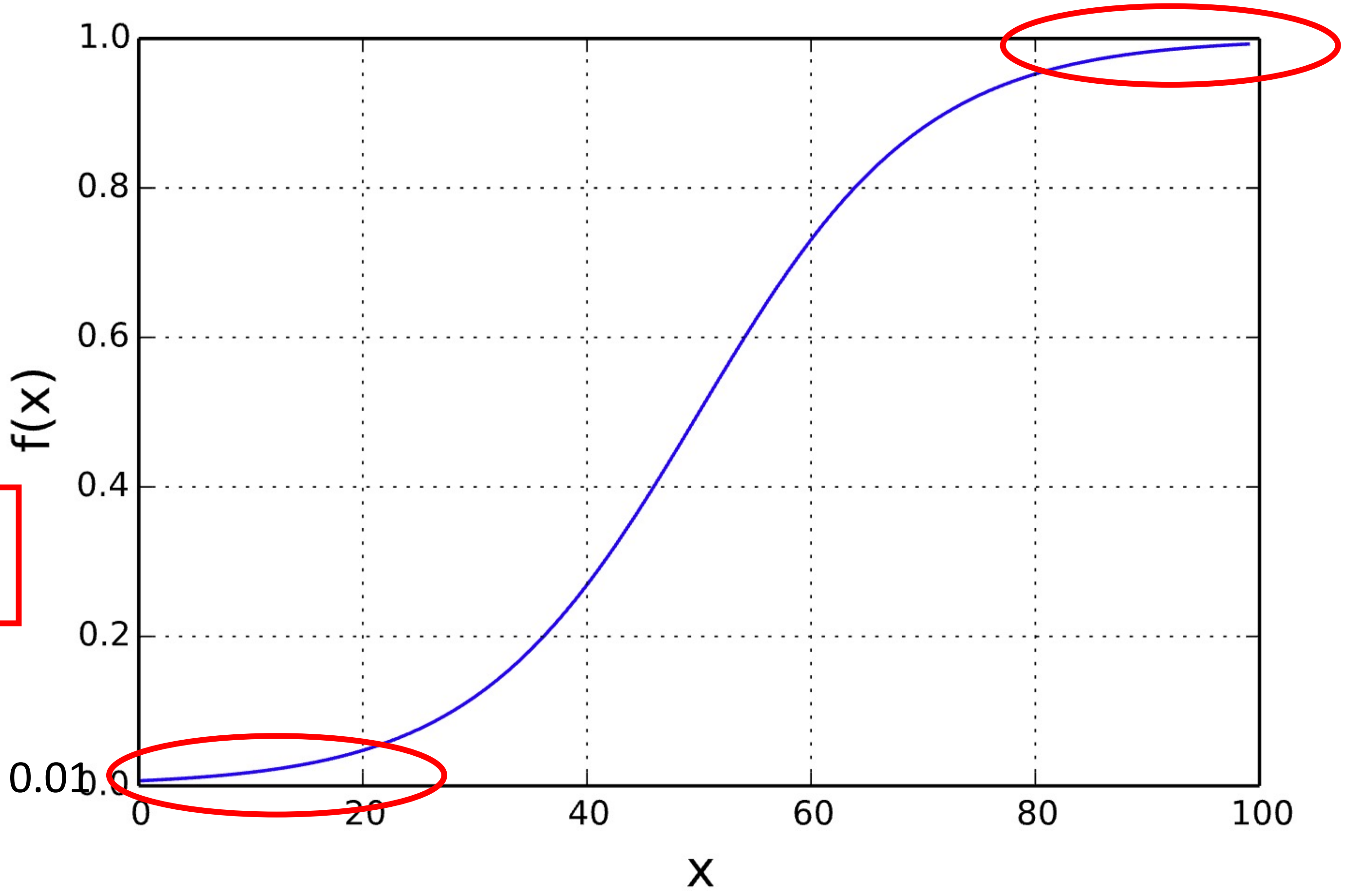
Sigmoid function

Gradient vanish

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \to ?$$

$$f_i \to x_i$$

$$\frac{dx_n}{dx} = \boxed{\frac{dx_n}{dx_{n-1}}} \cdot \dots \cdot \boxed{\frac{dx_2}{dx_1}} \cdot \boxed{\frac{dx_1}{dx}}$$



gradients-> 0.01

gradients-> 0.01

Sigmoid function

Gradient vanish

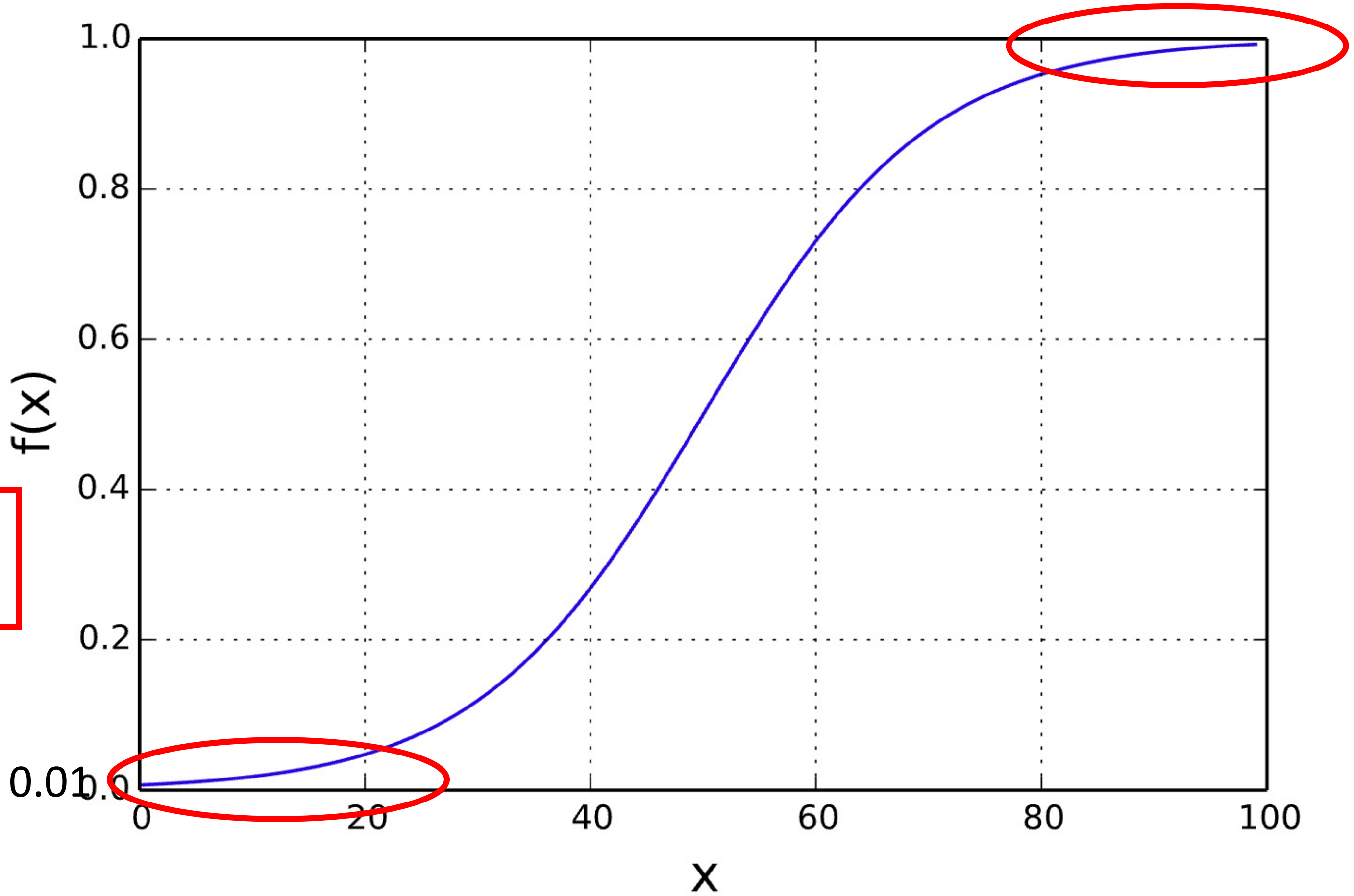
$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$\frac{dx_n}{dx} = \boxed{\frac{dx_n}{dx_{n-1}}} \cdot \dots \cdot \boxed{\frac{dx_2}{dx_1} \cdot \frac{dx_1}{dx}}$$

$\rightarrow 0.01^n$

gradients-> 0.01



gradients-> 0.01

Sigmoid function

Gradient explosion

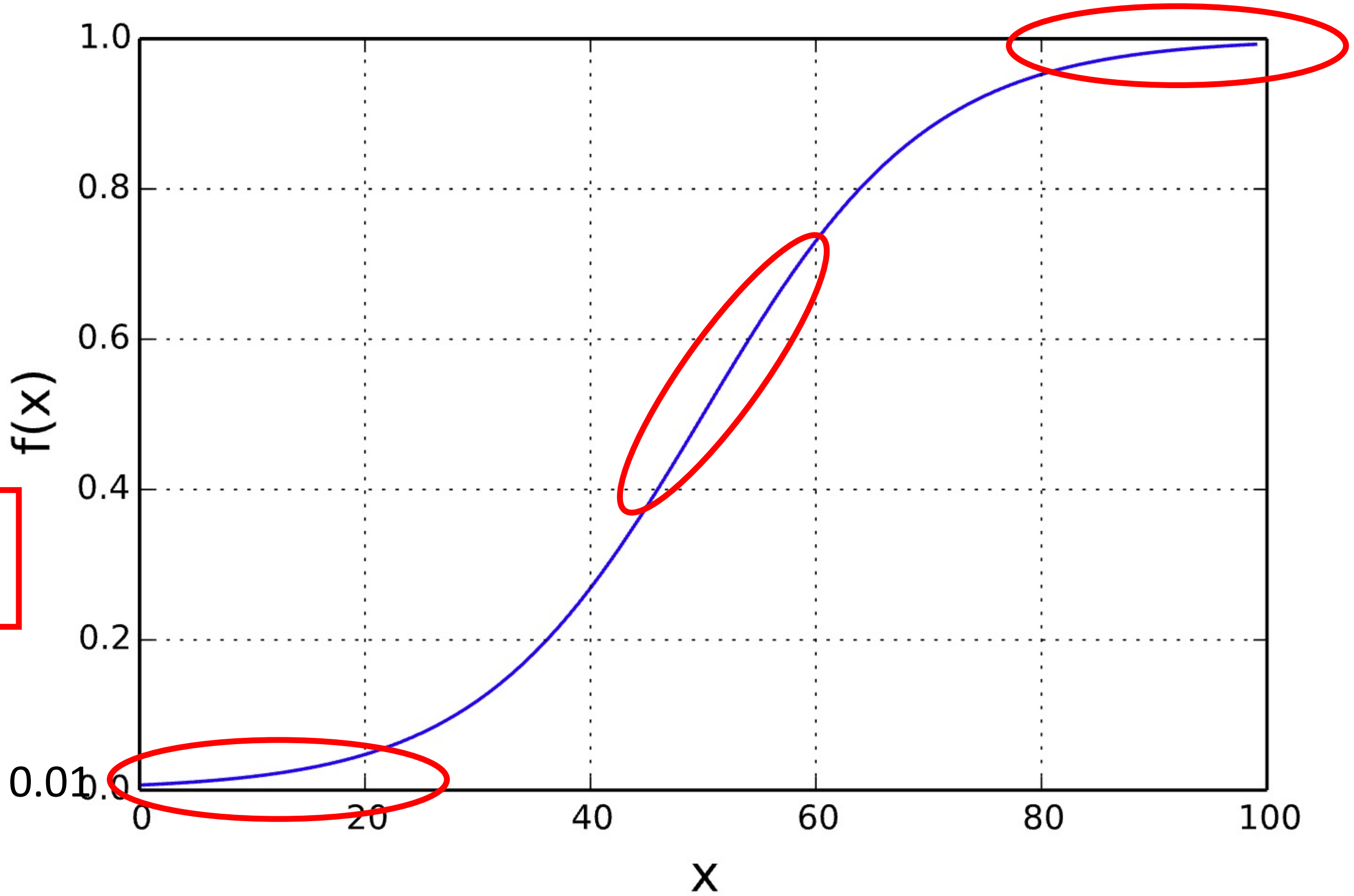
$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$\frac{dx_n}{dx} = \boxed{\frac{dx_n}{dx_{n-1}}} \cdot \dots \cdot \boxed{\frac{dx_2}{dx_1}} \boxed{\frac{dx_1}{dx}}$$

$\rightarrow 1.1^n$

gradients-> 0.01



gradients-> 0.01

Sigmoid function

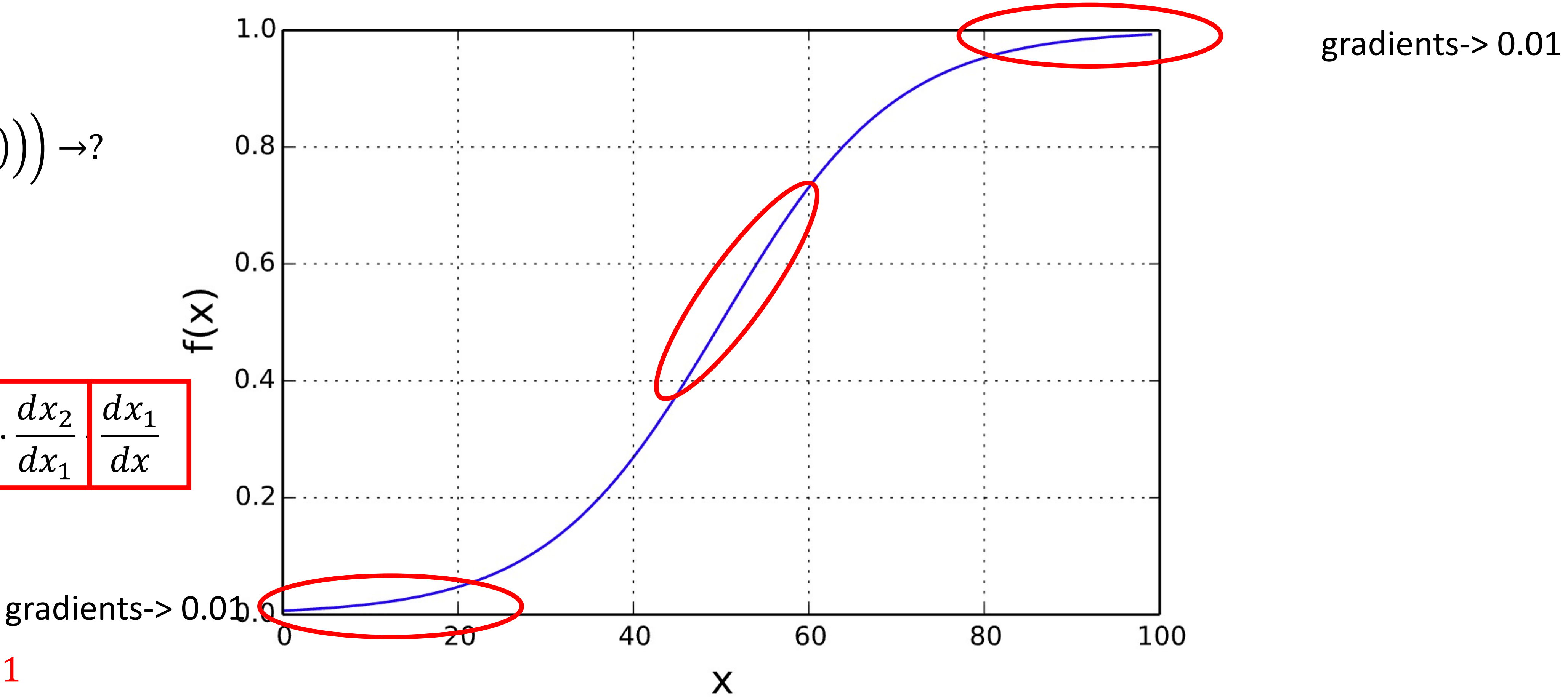
Gradient explosion

$$f_n\left(\dots\left(f_2\left(f_1(x)\right)\right)\right) \rightarrow ?$$

$$f_i \rightarrow x_i$$

$$\frac{dx_n}{dx} = \boxed{\frac{dx_n}{dx_{n-1}}} \cdot \dots \cdot \boxed{\frac{dx_2}{dx_1}} \cdot \boxed{\frac{dx_1}{dx}}$$

$\rightarrow 1.1^n$



$1.1^{100} = 13781$

Sigmoid function