

Automatic Differentiation

Alfan F. Wicaksono

Fakultas Ilmu Komputer, Universitas Indonesia

Some slides are taken from "Alice's Adventures in a Differentiable Wonderland" by [Simone Scardapane](#).

Why not numerical differentiation?

- We could directly apply the definition of gradients to obtain a suitable numerical approximation of the gradient.

$$\partial_{x_i} f(\mathbf{x}) = \frac{\partial y}{\partial x_i} = \lim_{h \rightarrow 0} \frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h},$$

- However, each scalar value to be differentiated requires **2 function calls** in a naive implementation, making this approach unfeasible except for numerical checks over the implementation.

Why not symbolic differentiation?

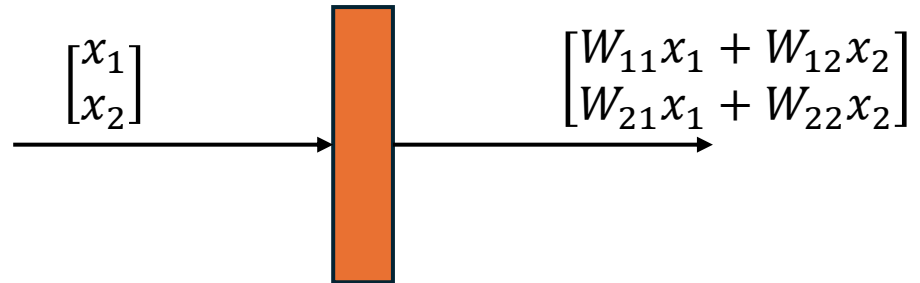
- We can ask a **symbolic engine** to pre-compute the full, symbolic equation of the derivative. This is called symbolic differentiation.
- Finding an optimal implementation for the Jacobian which avoids any unnecessary computation is an **NP-complete task** (**optimal Jacobian accumulation**).

```
import sympy as sp
x, a, b = sp.symbols('x a b')
y = a*sp.sin(x) + b*x*sp.sin(x)
sp.diff(y, x) # [Out]: acos(x)+bxcos(x)+bsin(x)
```

Box C.6.1: *Symbolic differentiation in Python using SymPy.*

$f(x, w)$ adalah sebuah fungsi "fully-connected layer" dengan **input vektor** $\sim (c)$ dan **output vektor** $\sim (c)$; dan mempunyai parameter $W \sim (c, c)$.

Contoh, untuk $c = 2$:



$$f(x, W) = Wx = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

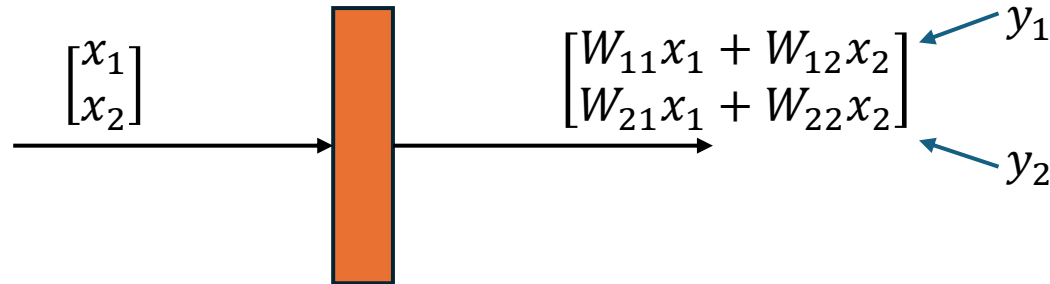
Saat proses optimization untuk mencari W , biasanya kita membutuhkan perhitungan dua buah kuantitas:

Input Jacobian : $\partial_x f(x, W)$

Weight Jacobian : $\partial_W f(x, W)$

$f(x, w)$ adalah sebuah fungsi "fully-connected layer" dengan **input vektor** $\sim (c)$ dan **output vektor** $\sim (c)$; dan mempunyai parameter $W \sim (c, c)$.

Contoh, untuk $c = 2$:



$$f(x, W) = Wx = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Untuk contoh di atas:

Input Jacobian $\sim (2, 2)$:

$$\partial_x f(x, W) = W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix}$$

Weight Jacobian $\sim (2, 2 \times 2)$:

$$\partial_W f(x, W) = \begin{bmatrix} \frac{\partial y_1}{\partial W_{11}} & \frac{\partial y_1}{\partial W_{12}} & \frac{\partial y_1}{\partial W_{21}} & \frac{\partial y_1}{\partial W_{22}} \\ \frac{\partial y_2}{\partial W_{11}} & \frac{\partial y_2}{\partial W_{12}} & \frac{\partial y_2}{\partial W_{21}} & \frac{\partial y_2}{\partial W_{22}} \end{bmatrix}$$

$$\partial_W f(x, W) = \begin{bmatrix} x_1 & x_2 & 0 & 0 \\ 0 & 0 & x_1 & x_2 \end{bmatrix}$$

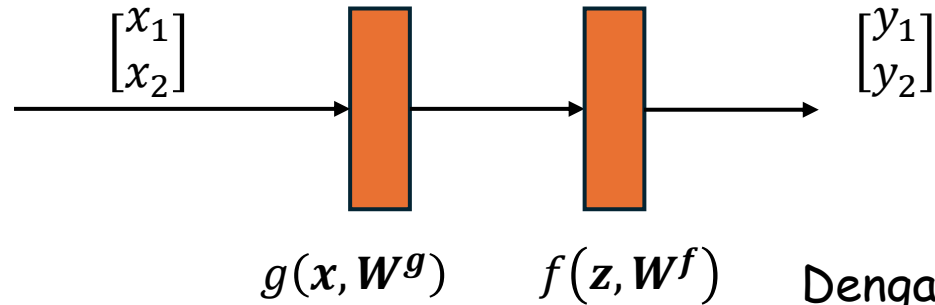
Setelah Anda mengetahui cara menghitung **input jacobian** dan **weight jacobian** dari sebuah fungsi **fully-connected layer** berparameter, sekarang kita mencoba jika terdapat 2 buah layer.

W^f bukan berarti W pangkat f ; tetapi parameter W pada fungsi f . Fungsi f dan g punya parameter yang berbeda.

Sekarang, seandainya ada 2 layer:

$$f(z, W^f) = W^f z = \begin{bmatrix} W_{11}^f & W_{12}^f \\ W_{21}^f & W_{22}^f \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$g(x, W^g) = W^g x = \begin{bmatrix} W_{11}^g & W_{12}^g \\ W_{21}^g & W_{22}^g \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



Dengan kata lain, ini adalah komposisi fungsi:

$$y = f(g(x, W^g), W^f)$$

Biasanya, kita ingin menghitung $\partial y / \partial W^g$ dan $\partial y / \partial W^f$.

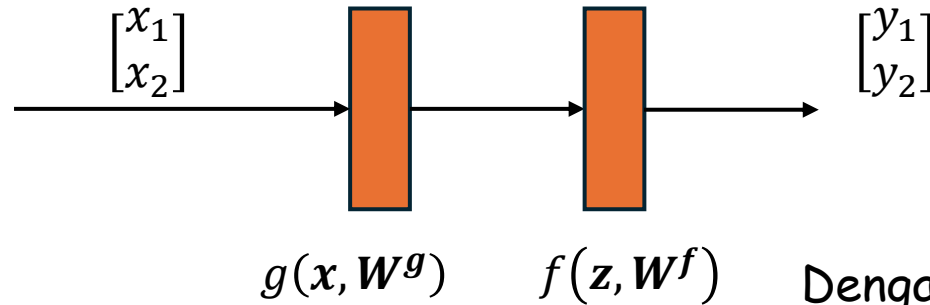
Bagaimana caranya?

W^f bukan berarti W pangkat f ; tetapi parameter W pada fungsi f . Fungsi f dan g punya parameter yang berbeda.

Sekarang, seandainya ada 2 layer:

$$f(z, W^f) = W^f z = \begin{bmatrix} W_{11}^f & W_{12}^f \\ W_{21}^f & W_{22}^f \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$g(x, W^g) = W^g x = \begin{bmatrix} W_{11}^g & W_{12}^g \\ W_{21}^g & W_{22}^g \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



Dengan kata lain, ini adalah komposisi fungsi:

$$y = f(g(x, W^g), W^f)$$

Biasanya, kita ingin menghitung $\partial y / \partial W^g$ dan $\partial y / \partial W^f$.

Chain Rule again!

$$\frac{\partial y}{\partial W^g} = \frac{\partial f}{\partial W^g} = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial W^g}$$

Input Jacobian f :
 $\partial_x f(\cdot, W^f)$

Weight Jacobian g :
 $\partial_{W^g} g(\cdot, W^g)$

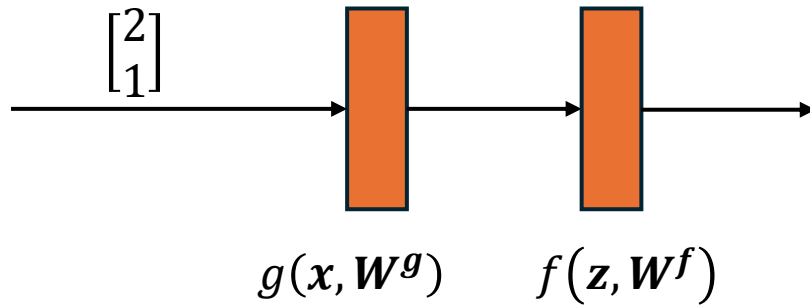
$$\frac{\partial y}{\partial W^f} = \frac{\partial f}{\partial W^f}$$

Weight Jacobian f :
 $\partial_{W^f} f(\cdot, W^f)$

Sekarang, dengan input tertentu dan parameter \mathbf{W} diinisialisasi dengan nilai tertentu.

$$f(\mathbf{z}, \mathbf{W}^f) = \mathbf{W}^f \mathbf{z} = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$g(\mathbf{x}, \mathbf{W}^g) = \mathbf{W}^g \mathbf{x} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



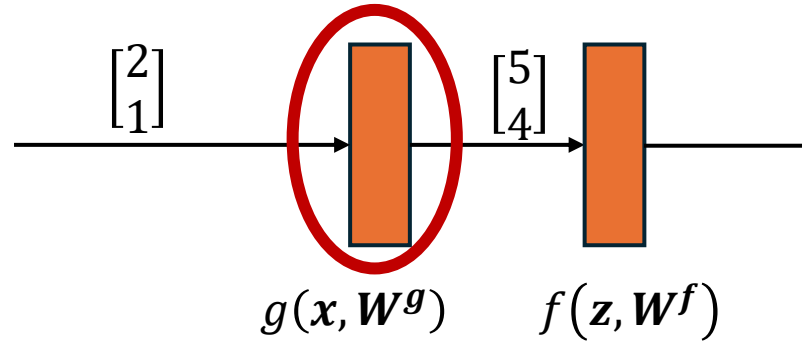
$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^g} = \frac{\partial f}{\partial \mathbf{W}^g} = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial \mathbf{W}^g}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^f} = \frac{\partial f}{\partial \mathbf{W}^f}$$

Sekarang, dengan input tertentu dan parameter \mathbf{W} diinisialisasi dengan nilai tertentu.

$$f(\mathbf{z}, \mathbf{W}^f) = \mathbf{W}^f \mathbf{z} = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$g(\mathbf{x}, \mathbf{W}^g) = \mathbf{W}^g \mathbf{x} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$$



$$\partial_{\mathbf{W}^g} g(\mathbf{z}, \mathbf{W}^g) = \begin{bmatrix} x_1 & x_2 & 0 & 0 \\ 0 & 0 & x_1 & x_2 \end{bmatrix}$$

$$\partial_{\mathbf{W}^g} g(\mathbf{z}, \mathbf{W}^g) = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 \end{bmatrix}$$

Buat sebuah matrix $\widehat{\mathbf{W}}^g$ yang diinisialisasi dengan Weight Jacobian g :

$$\widehat{\mathbf{W}}^g \leftarrow \begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 \end{bmatrix}$$

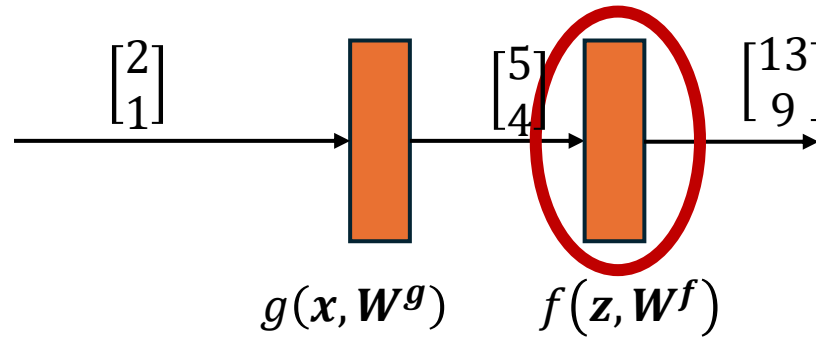
$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^g} = \frac{\partial f}{\partial \mathbf{W}^g} = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial \mathbf{W}^g}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^f} = \frac{\partial f}{\partial \mathbf{W}^f}$$

Sekarang, dengan input tertentu dan parameter \mathbf{W} diinisialisasi dengan nilai tertentu.

$$f(\mathbf{z}, \mathbf{W}^f) = \mathbf{W}^f \mathbf{z} = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 5 \\ 4 \end{bmatrix} = \begin{bmatrix} 13 \\ 9 \end{bmatrix}$$

$$g(\mathbf{x}, \mathbf{W}^g) = \mathbf{W}^g \mathbf{x} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$$



$$\partial_{\mathbf{z}} f(\mathbf{z}, \mathbf{W}^f) = \begin{bmatrix} W_{11}^f & W_{12}^f \\ W_{21}^f & W_{22}^f \end{bmatrix}$$

$$\partial_{\mathbf{z}} f(\mathbf{z}, \mathbf{W}^f) = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix}$$

$$\partial_{\mathbf{W}^f} f(\mathbf{z}, \mathbf{W}^f) = \begin{bmatrix} z_1 & z_2 & 0 & 0 \\ 0 & 0 & z_1 & z_2 \end{bmatrix}$$

$$\partial_{\mathbf{W}^f} f(\mathbf{z}, \mathbf{W}^f) = \begin{bmatrix} 5 & 4 & 0 & 0 \\ 0 & 0 & 5 & 4 \end{bmatrix}$$

Update matrix $\widehat{\mathbf{W}}^g$ dikali dengan Input Jacobian \mathbf{f} .

$$\widehat{\mathbf{W}}^g \leftarrow \partial_{\mathbf{z}} f(\mathbf{z}, \mathbf{W}^f) \times \widehat{\mathbf{W}}^g$$

$$\widehat{\mathbf{W}}^g \leftarrow \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 4 & 2 \\ 2 & 1 & 2 & 1 \end{bmatrix}$$

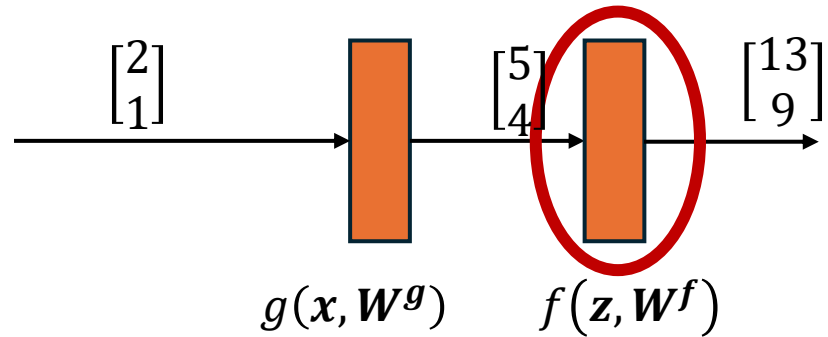
$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^g} = \frac{\partial f}{\partial \mathbf{W}^g} = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial \mathbf{W}^g}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^f} = \frac{\partial f}{\partial \mathbf{W}^f}$$

Buat sebuah matrix $\widehat{\mathbf{W}}^f$ yang diinisialisasi dengan Weight Jacobian \mathbf{f} :

$$\widehat{\mathbf{W}}^f \leftarrow \begin{bmatrix} 5 & 4 & 0 & 0 \\ 0 & 0 & 5 & 4 \end{bmatrix}$$

Sekarang, dengan input tertentu dan parameter \mathbf{W} diinisialisasi dengan nilai tertentu.



$$f(\mathbf{z}, \mathbf{W}^f) = \mathbf{W}^f \mathbf{z} = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 5 \\ 4 \end{bmatrix} = \begin{bmatrix} 13 \\ 9 \end{bmatrix}$$

$$g(\mathbf{x}, \mathbf{W}^g) = \mathbf{W}^g \mathbf{x} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^g} = \frac{\partial f}{\partial \mathbf{W}^g} = \boxed{\frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial \mathbf{W}^g}} = \widehat{\mathbf{W}^g}$$

$$\widehat{\mathbf{W}^g} = \begin{bmatrix} 2 & 1 & 4 & 2 \\ 2 & 1 & 2 & 1 \end{bmatrix}$$

Arrows indicate the flow of gradients from the output derivatives $\frac{\partial y_1}{\partial W_{12}^g}$ and $\frac{\partial y_2}{\partial W_{21}^g}$ to the corresponding elements in the $\widehat{\mathbf{W}^g}$ matrix.

$$\widehat{\mathbf{W}^f} = \begin{bmatrix} 5 & 4 & 0 & 0 \\ 0 & 0 & 5 & 4 \end{bmatrix}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{W}^f} = \boxed{\frac{\partial f}{\partial \mathbf{W}^f}} = \widehat{\mathbf{W}^f}$$

Sekarang, kita perumum semuanya, dan bungkus dalam sebuah konsep yang Bernama "Automatic Differentiation"

Automatic Differentiation: Problem Statement

Consider a sequence of l primitive calls:

$$\mathbf{h}_1 = f_1(\mathbf{x}, \mathbf{W}_1)$$

$$\mathbf{h}_2 = f_2(\mathbf{h}_1, \mathbf{W}_2)$$

$$\vdots$$

$$\mathbf{h}_l = f_l(\mathbf{h}_{l-1}, \mathbf{W}_l)$$

This is called an evaluation trace of the program. Roughly, the first $l - 1$ operations can represent several layers of a differentiable model, operation l can be a per-input loss (e.g., cross-entropy).

Hence, the output of our program is always a scalar, since we require it for numerical optimization.

Automatic Differentiation: Problem Statement

Definition D.6.1 (Automatic differentiation) *Given a program $F(\mathbf{x})$ composed of a sequence of differentiable primitives, **automatic differentiation** (AD) refers to the task of simultaneously and efficiently computing all weight Jacobians of the program given knowledge of the computational graph and all individuals input and weight Jacobians:*

$$AD(F(\mathbf{x})) = \{\partial_{\mathbf{w}_i} y\}_{i=1}^l$$

There are two major classes of AD algorithms, called **forward-mode** and **backward-mode**, corresponding to a different ordering in the composition of the individual operations.

We will also see that the **backward-mode** (called **back-propagation** in the neural networks' literature) is significantly more efficient in our context.

Automatic Differentiation: Forward Mode AD

$$\mathbf{h}_1 = f_1(\mathbf{x}, \mathbf{W}_1)$$

$$\mathbf{h}_2 = f_2(\mathbf{h}_1, \mathbf{W}_2)$$

\vdots

$$\mathbf{h}_l = f_l(\mathbf{h}_{l-1}, \mathbf{W}_l)$$


The idea is that every time we apply a primitive function, we initialize its corresponding weight Jacobian (called **tangent** in this context) $\widehat{\mathbf{W}}_i$, while simultaneously updating all previous tangent matrices.

What we want to compute:

$$\frac{\partial \mathbf{h}_l}{\partial \mathbf{W}_1} = \frac{\partial \mathbf{h}_l}{\partial \mathbf{h}_{l-1}} \cdot \frac{\partial \mathbf{h}_{l-1}}{\partial \mathbf{h}_{l-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial \mathbf{W}_1}$$

$$\frac{\partial \mathbf{h}_l}{\partial \mathbf{W}_2} = \frac{\partial \mathbf{h}_l}{\partial \mathbf{h}_{l-1}} \cdot \frac{\partial \mathbf{h}_{l-1}}{\partial \mathbf{h}_{l-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}_2}$$

\vdots

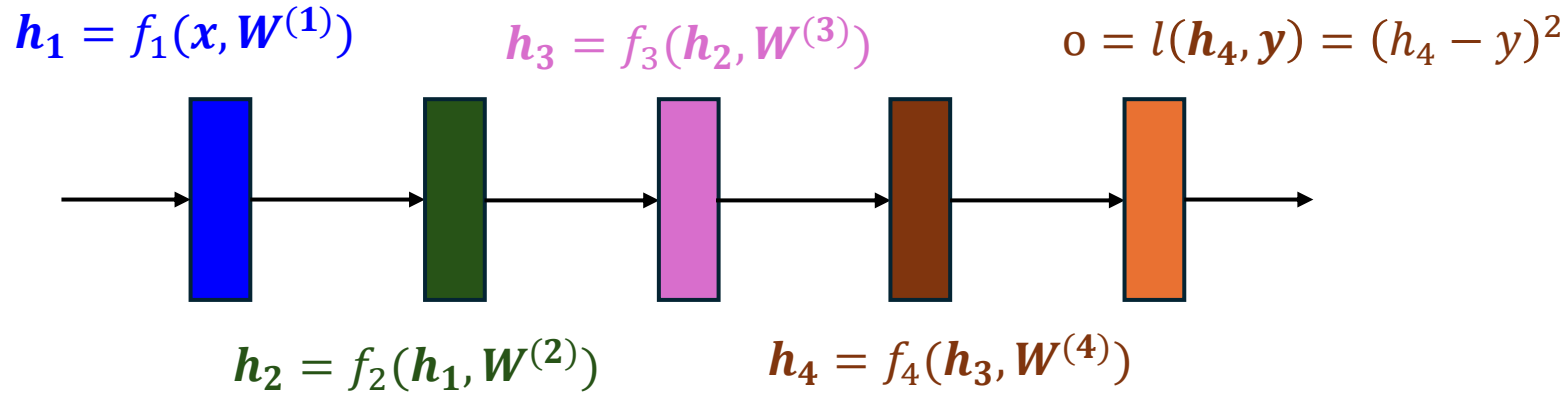

$$\widehat{\mathbf{W}}_j \leftarrow \left[\frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}} \right] \widehat{\mathbf{W}}_j \quad \forall j < i$$

$$\widehat{\mathbf{W}}_i \leftarrow \frac{\partial \mathbf{h}_i}{\partial \mathbf{W}_i}$$



Weight jacobian of f_i

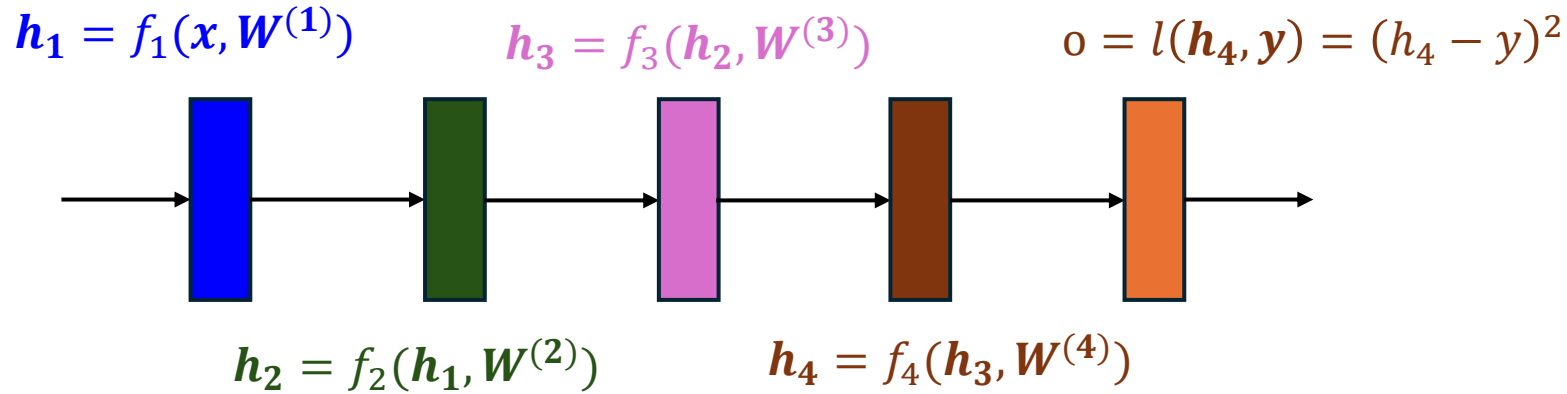
Automatic Differentiation: Forward Mode AD



Misal, Input adalah **vektor** ~ (2).

Misal, 3 layer pertama adalah fully-connected models dengan **output** ~ (2); layer ke-4 adalah fully connected model dengan **output scalar**; dan layer terakhir adalah fungsi loss, yaitu squared loss, yang mengembalikan **scalar**.

Automatic Differentiation: Forward Mode AD



$$f_1(x, W^{(1)}) = W^{(1)}x = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$f_4(h_3, W^{(4)}) = W^{(4)}h_3 = \begin{bmatrix} W_{11}^{(4)} & W_{12}^{(4)} \end{bmatrix} \begin{bmatrix} h_{31} \\ h_{32} \end{bmatrix}$$

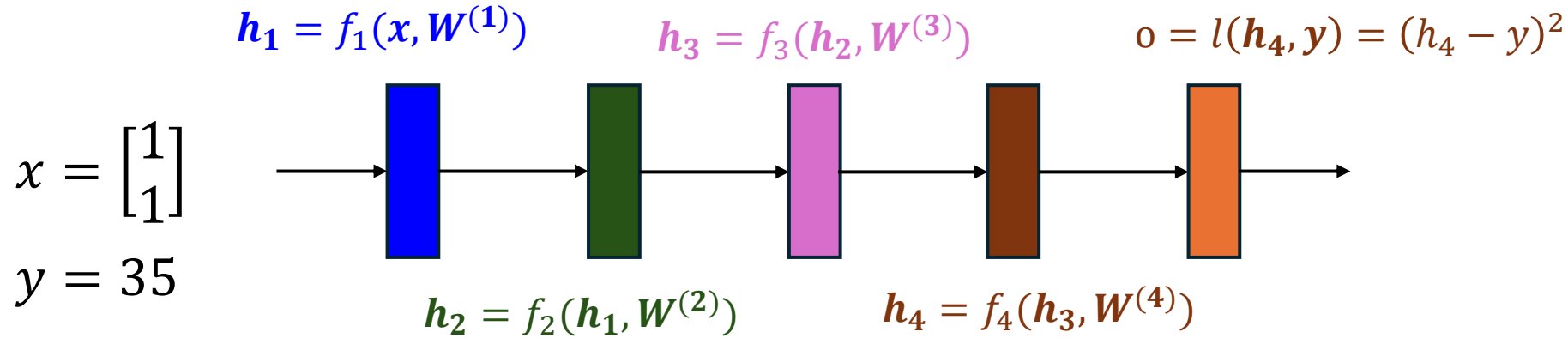
$$f_2(h_1, W^{(2)}) = W^{(2)}h_1 = \begin{bmatrix} W_{11}^{(2)} & W_{12}^{(2)} \\ W_{21}^{(2)} & W_{22}^{(2)} \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \end{bmatrix}$$

$$l(h_4, y) = (h_4 - y)^2$$

$$f_3(h_2, W^{(3)}) = W^{(3)}h_2 = \begin{bmatrix} W_{11}^{(3)} & W_{12}^{(3)} \\ W_{21}^{(3)} & W_{22}^{(3)} \end{bmatrix} \begin{bmatrix} h_{21} \\ h_{22} \end{bmatrix}$$

Total ada 14 parameter!

Automatic Differentiation: Forward Mode AD



$$f_1(x, W^{(1)}) = W^{(1)}x = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$f_4(h_3, W^{(4)}) = W^{(4)}h_3 = \begin{bmatrix} 2 & 1 \end{bmatrix} \begin{bmatrix} h_{31} \\ h_{32} \end{bmatrix}$$

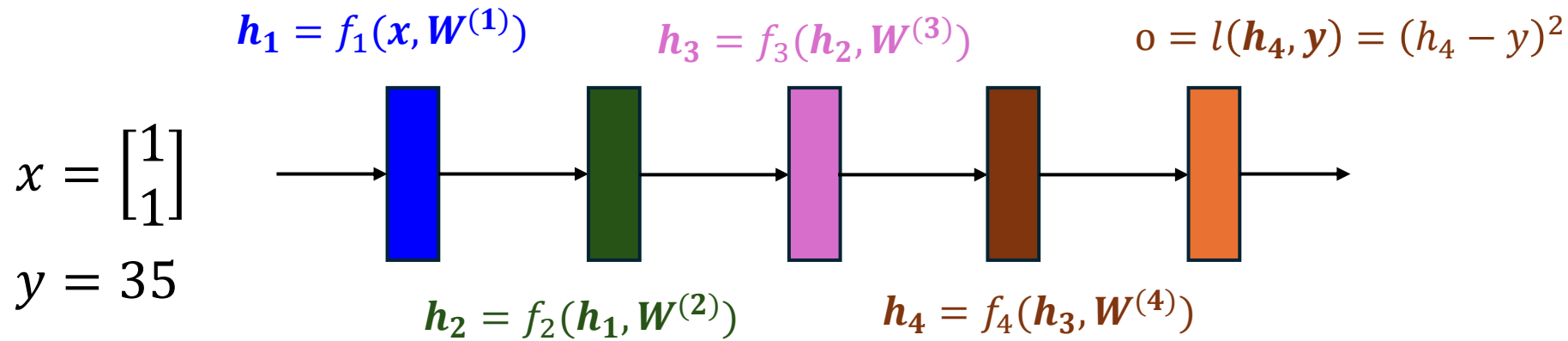
$$f_2(h_1, W^{(2)}) = W^{(2)}h_1 = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \end{bmatrix}$$

$$l(h_4, y) = (h_4 - y)^2$$

$$f_3(h_2, W^{(3)}) = W^{(3)}h_2 = \begin{bmatrix} 1 & 1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} h_{21} \\ h_{22} \end{bmatrix}$$

Misal, kita mulai dari inisialisasi parameter tertentu (random); dan untuk input x & y tertentu.

Automatic Differentiation: Forward Mode AD



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

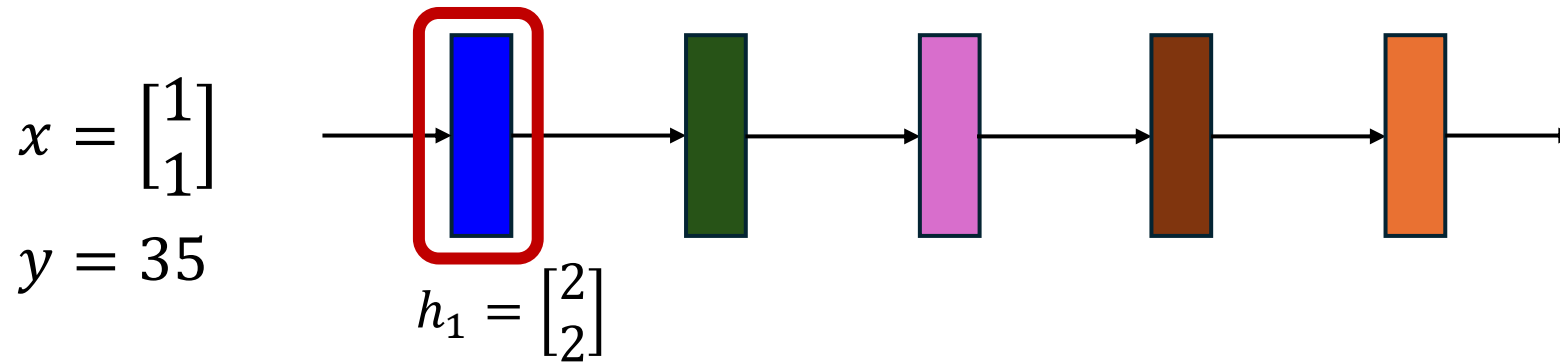
$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

Input dan output loss function sudah scalar. Turunannya juga scalar:

$$\frac{\partial l}{\partial h_4} = 2(h_4 - y) = 2(h_4 - 2)$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2)$$



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\widehat{w}_1 \leftarrow \frac{\partial h_1}{\partial W^{(1)}}$$

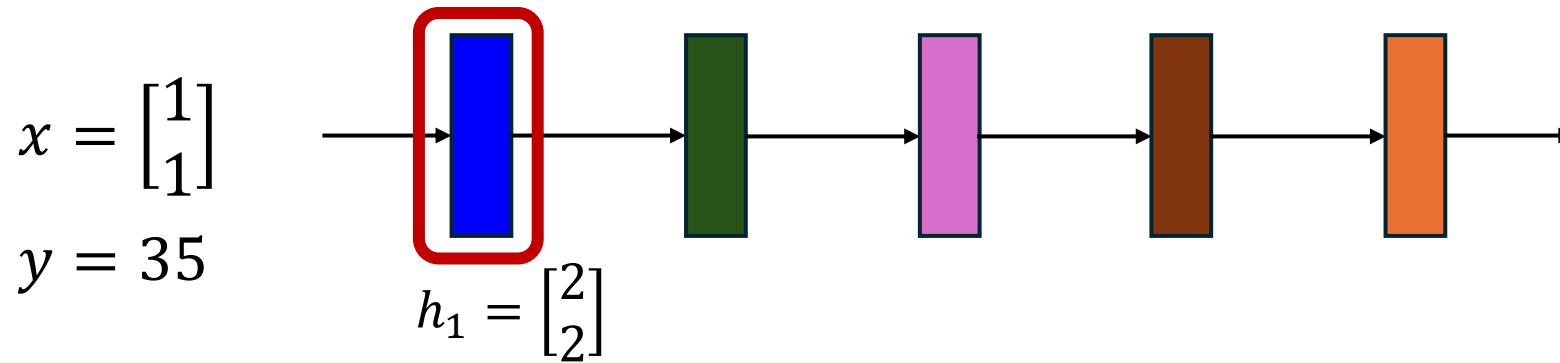
$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial \mathbf{h}_4} = 2(\mathbf{h}_4 - 2)$$



$$\frac{\partial l}{\partial \mathbf{W}^{(1)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial \mathbf{W}^{(1)}}$$

$$\widehat{\mathbf{W}}_1 \leftarrow \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

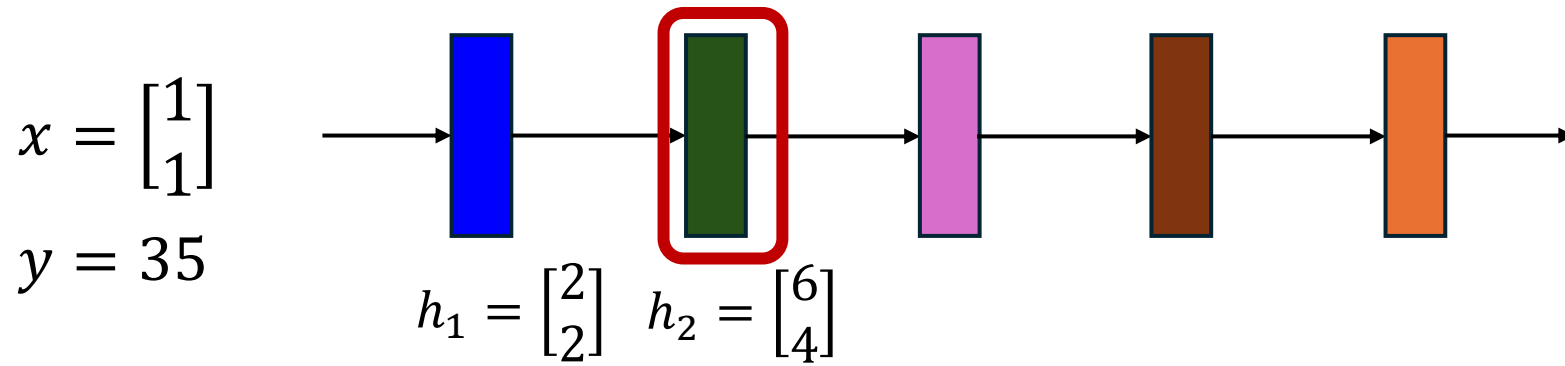
$$\frac{\partial l}{\partial \mathbf{W}^{(2)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}^{(2)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(3)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{W}^{(3)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(4)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{W}^{(4)}}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial \mathbf{h}_4} = 2(\mathbf{h}_4 - 2)$$



$$\frac{\partial l}{\partial \mathbf{W}^{(1)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial \mathbf{W}^{(1)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(2)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}^{(2)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(3)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{W}^{(3)}}$$

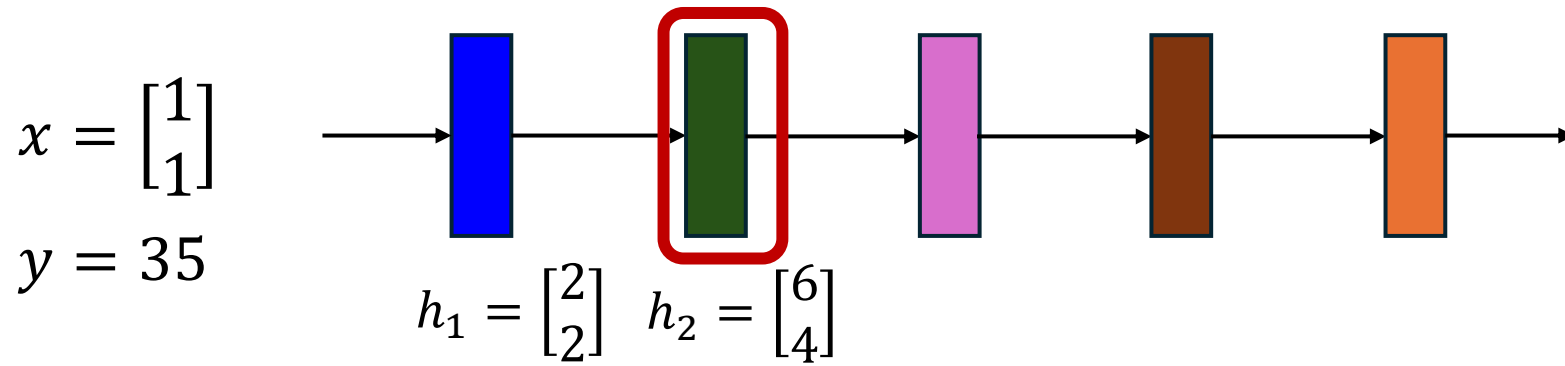
$$\frac{\partial l}{\partial \mathbf{W}^{(4)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{W}^{(4)}}$$

$$\widehat{\mathbf{W}}_1 \leftarrow \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$\widehat{\mathbf{W}}_2 \leftarrow \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}^{(2)}}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2)$$



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

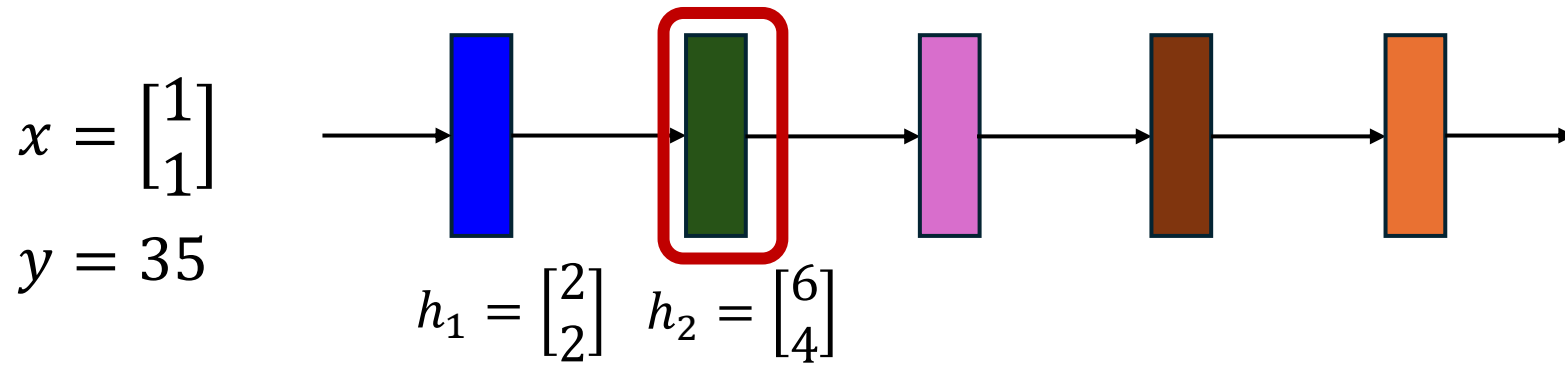
$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

$$\widehat{W}_1 \leftarrow \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$\widehat{W}_2 \leftarrow \begin{bmatrix} 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 \end{bmatrix}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial \mathbf{h}_4} = 2(\mathbf{h}_4 - 2)$$



$$\frac{\partial l}{\partial \mathbf{W}^{(1)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial \mathbf{W}^{(1)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(2)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}^{(2)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(3)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{W}^{(3)}}$$

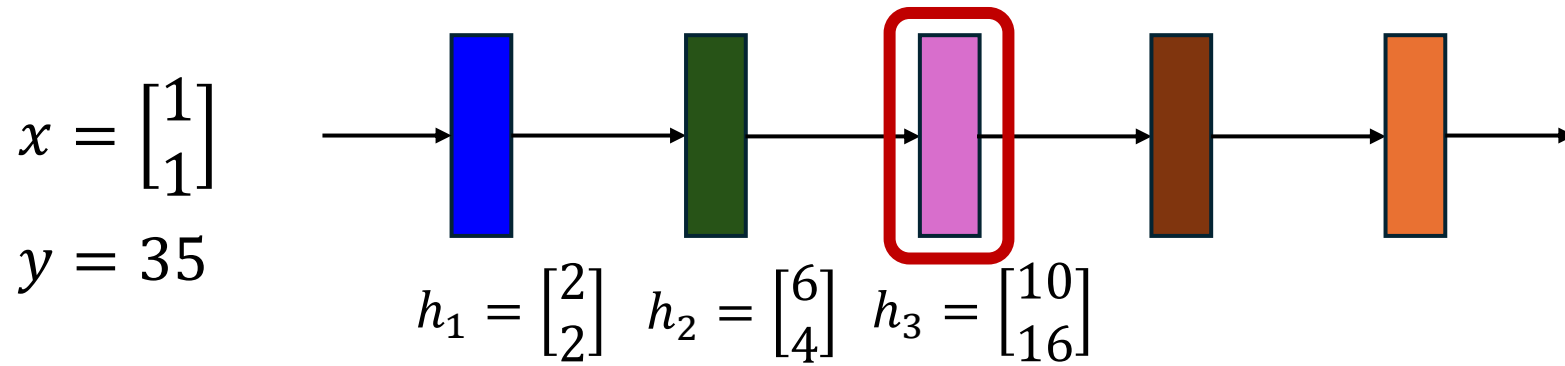
$$\frac{\partial l}{\partial \mathbf{W}^{(4)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{W}^{(4)}}$$

$$\widehat{\mathbf{W}}_1 \leftarrow \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\widehat{\mathbf{W}}_2 \leftarrow \begin{bmatrix} 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 \end{bmatrix}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2)$$



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

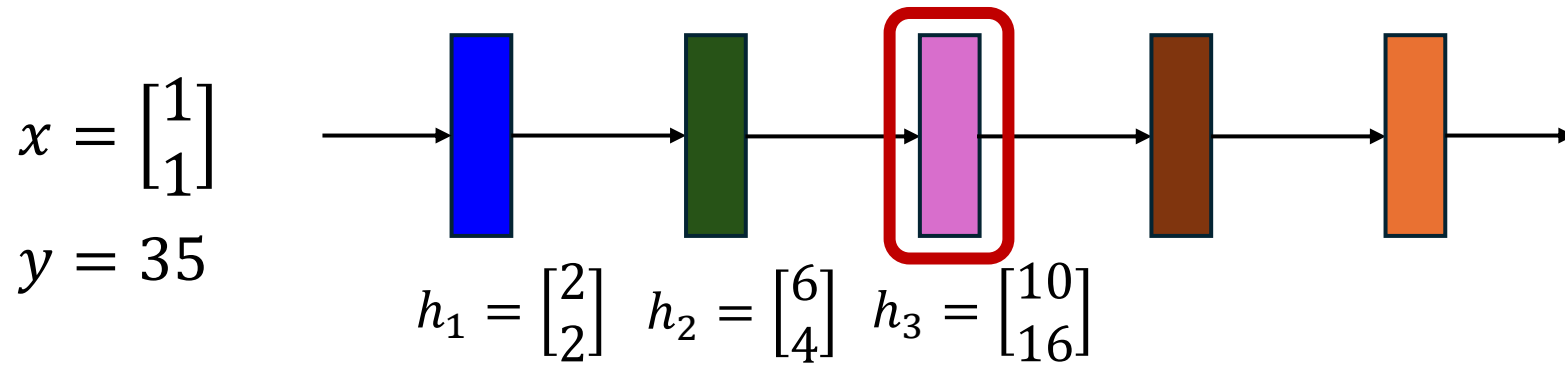
$$\widehat{W}_1 \leftarrow \frac{\partial h_3}{\partial h_2} \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\widehat{W}_2 \leftarrow \frac{\partial h_3}{\partial h_2} \begin{bmatrix} 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 \end{bmatrix}$$

$$\widehat{W}_3 \leftarrow \frac{\partial h_3}{\partial W^{(3)}}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial \mathbf{h}_4} = 2(\mathbf{h}_4 - 2)$$



$$\frac{\partial l}{\partial \mathbf{W}^{(1)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial \mathbf{W}^{(1)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(2)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{h}_2} \cdot \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}^{(2)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(3)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{h}_3} \cdot \frac{\partial \mathbf{h}_3}{\partial \mathbf{W}^{(3)}}$$

$$\frac{\partial l}{\partial \mathbf{W}^{(4)}} = \frac{\partial l}{\partial \mathbf{h}_4} \cdot \frac{\partial \mathbf{h}_4}{\partial \mathbf{W}^{(4)}}$$

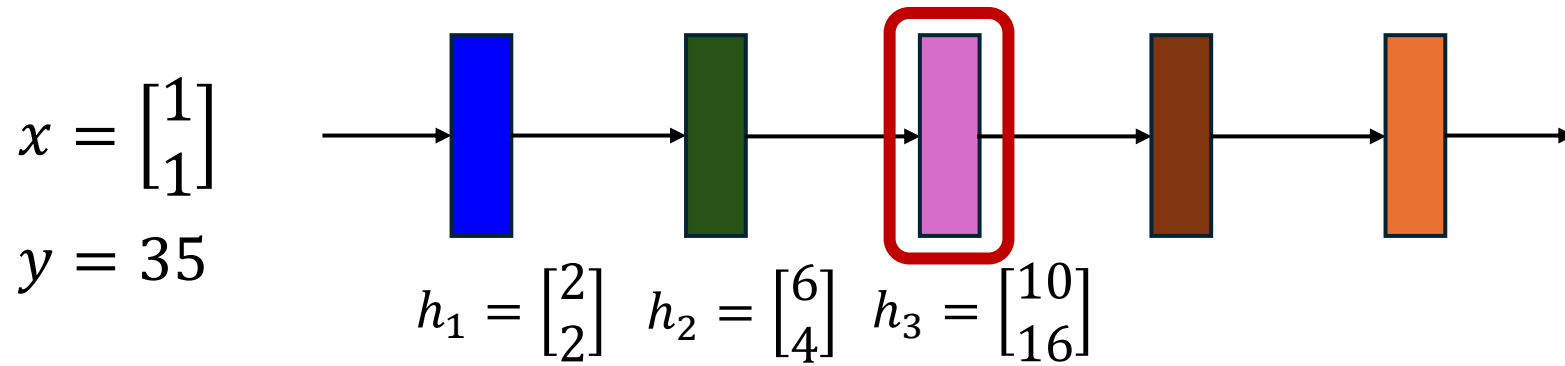
$$\widehat{\mathbf{W}}_1 \leftarrow \begin{bmatrix} 1 & 1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\widehat{\mathbf{W}}_2 \leftarrow \begin{bmatrix} 1 & 1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 \end{bmatrix}$$

$$\widehat{\mathbf{W}}_3 \leftarrow \begin{bmatrix} 6 & 4 & 0 & 0 \\ 0 & 0 & 6 & 4 \end{bmatrix}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2)$$



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

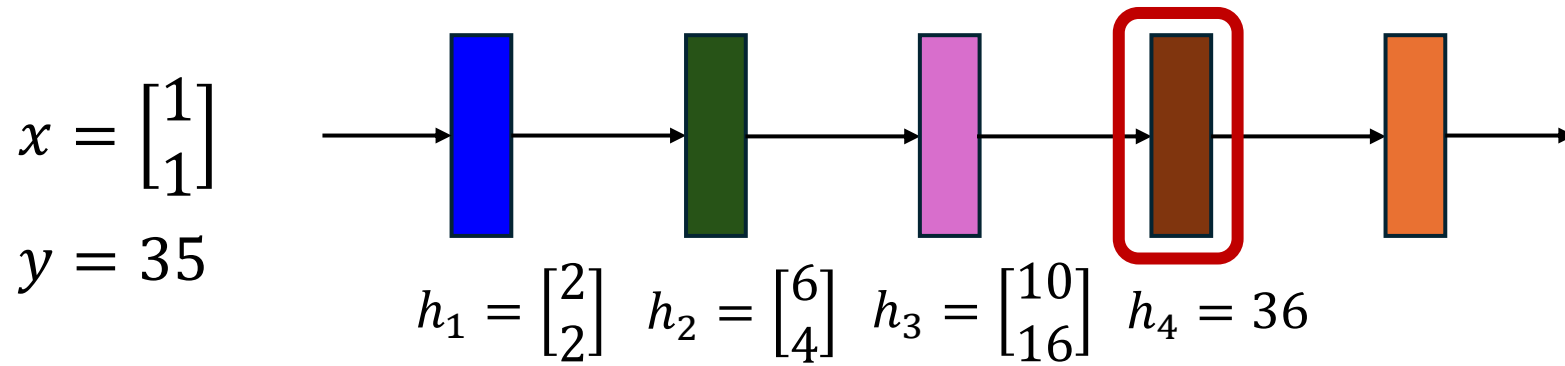
$$\widehat{W}_1 \leftarrow \begin{bmatrix} 2 & 2 & 3 & 3 \\ 3 & 3 & 5 & 5 \end{bmatrix}$$

$$\widehat{W}_2 \leftarrow \begin{bmatrix} 2 & 2 & 2 & 2 \\ 4 & 4 & 2 & 2 \end{bmatrix}$$

$$\widehat{W}_3 \leftarrow \begin{bmatrix} 6 & 4 & 0 & 0 \\ 0 & 0 & 6 & 4 \end{bmatrix}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2)$$



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

$$\widehat{W}_1 \leftarrow \frac{\partial h_4}{\partial h_3} \begin{bmatrix} 2 & 2 & 3 & 3 \\ 3 & 3 & 5 & 5 \end{bmatrix}$$

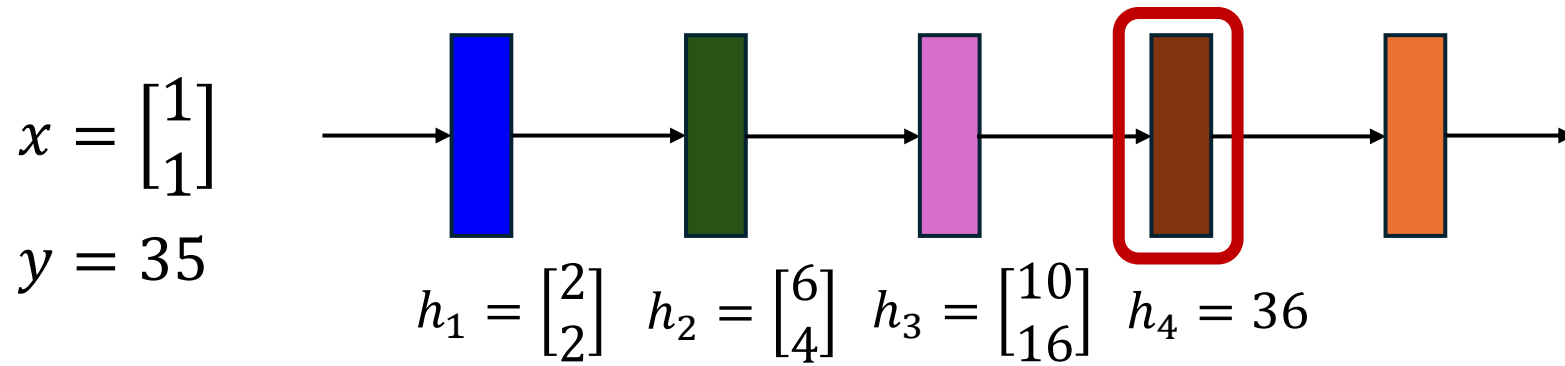
$$\widehat{W}_2 \leftarrow \frac{\partial h_4}{\partial h_3} \begin{bmatrix} 2 & 2 & 2 & 2 \\ 4 & 4 & 2 & 2 \end{bmatrix}$$

$$\widehat{W}_3 \leftarrow \frac{\partial h_4}{\partial h_3} \begin{bmatrix} 6 & 4 & 0 & 0 \\ 0 & 0 & 6 & 4 \end{bmatrix}$$

$$\widehat{W}_4 \leftarrow \frac{\partial h_4}{\partial W^{(4)}}$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2)$$



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

$$\widehat{W}_1 \leftarrow [2 \quad 1] \begin{bmatrix} 2 & 2 & 3 & 3 \\ 3 & 3 & 5 & 5 \end{bmatrix}$$

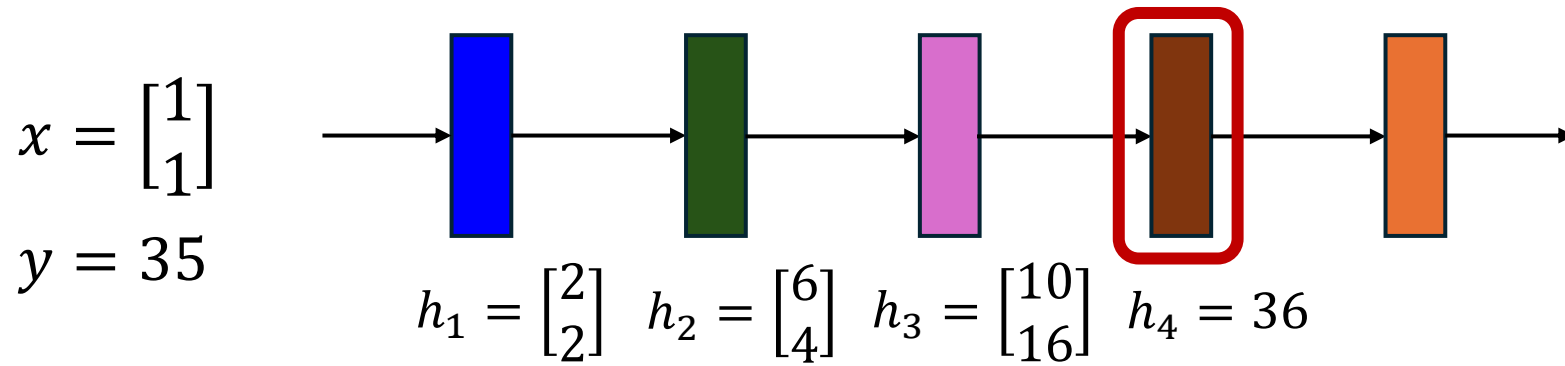
$$\widehat{W}_2 \leftarrow [2 \quad 1] \begin{bmatrix} 2 & 2 & 2 & 2 \\ 4 & 4 & 2 & 2 \end{bmatrix}$$

$$\widehat{W}_3 \leftarrow [2 \quad 1] \begin{bmatrix} 6 & 4 & 0 & 0 \\ 0 & 0 & 6 & 4 \end{bmatrix}$$

$$\widehat{W}_4 \leftarrow [10 \quad 16]$$

Automatic Differentiation: Forward Mode AD

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2)$$



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\widehat{W}_1 \leftarrow [7 \quad 7 \quad 11 \quad 11]$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\widehat{W}_2 \leftarrow [8 \quad 8 \quad 6 \quad 6]$$

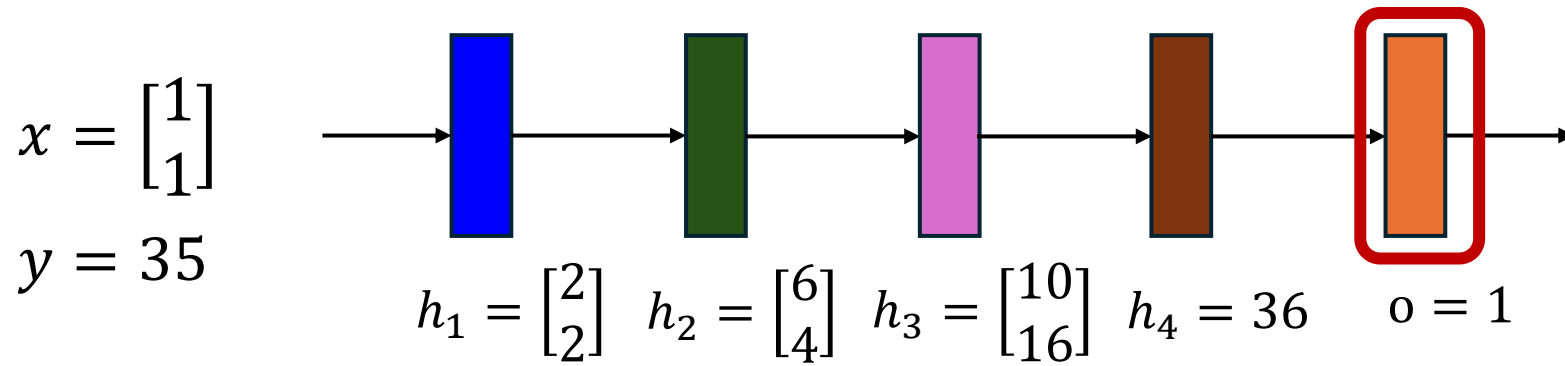
$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\widehat{W}_3 \leftarrow [12 \quad 8 \quad 6 \quad 4]$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

$$\widehat{W}_4 \leftarrow [10 \quad 16]$$

Automatic Differentiation: Forward Mode AD



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

$$\widehat{W}_1 \leftarrow [476 \quad 476 \quad 748 \quad 748]$$

$$\widehat{W}_2 \leftarrow [544 \quad 544 \quad 408 \quad 408]$$

$$\widehat{W}_3 \leftarrow [816 \quad 544 \quad 408 \quad 272]$$

$$\widehat{W}_4 \leftarrow [680 \quad 1088]$$

$$\frac{\partial l}{\partial h_4} = 2(h_4 - 2) = 68$$

If we have a sequence of matrix multiplications like below, where would you start in order to be efficient?

$$[2 \quad 1] \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

If we have a sequence of matrix multiplications like below, where would you start in order to be efficient?

$$[2 \quad 1] \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$



$2 \times 2 \times 4 = 16$ basic operations



$1 \times 2 \times 4 = 8$ basic operations

Total 24 basic operations

If we have a sequence of matrix multiplications like below, where would you start in order to be efficient?

$$[2 \quad 1] \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$



$1 \times 2 \times 2 = 4$ basic operations

Total 12 basic operations!



$1 \times 2 \times 4 = 8$ basic operations

Automatic Differentiation: Reverse Mode AD

$$\mathbf{h}_1 = f_1(\mathbf{x}, \mathbf{W}_1)$$

$$\mathbf{h}_2 = f_2(\mathbf{h}_1, \mathbf{W}_2)$$

\vdots

$$\mathbf{h}_l = f_l(\mathbf{h}_{l-1}, \mathbf{W}_l)$$

Because matrix multiplication is associative, we can perform the computations in any order. In **Forward-AD**, we proceeded from **the right to the left**, since it corresponds to the ordering in which the primitive functions were executed. However, we can do better by noting two interesting aspects:

What we want to compute:

$$\frac{\partial \mathbf{h}_l}{\partial \mathbf{W}_1} = \frac{\partial \mathbf{h}_l}{\partial \mathbf{h}_{l-1}} \cdot \frac{\partial \mathbf{h}_{l-1}}{\partial \mathbf{h}_{l-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial \mathbf{W}_1}$$

$$\frac{\partial \mathbf{h}_l}{\partial \mathbf{W}_2} = \frac{\partial \mathbf{h}_l}{\partial \mathbf{h}_{l-1}} \cdot \frac{\partial \mathbf{h}_{l-1}}{\partial \mathbf{h}_{l-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}_2}$$

\vdots

The leftmost term is a product between a vector and a matrix (which is a consequence of having a scalar term in output), which is computationally better than a product between two matrices.

Automatic Differentiation: Reverse Mode AD

$$\mathbf{h}_1 = f_1(\mathbf{x}, \mathbf{W}_1)$$

$$\mathbf{h}_2 = f_2(\mathbf{h}_1, \mathbf{W}_2)$$

\vdots

$$\mathbf{h}_l = f_l(\mathbf{h}_{l-1}, \mathbf{W}_l)$$

Because matrix multiplication is associative, we can perform the computations in any order. In **Forward-AD**, we proceeded from **the right to the left**, since it corresponds to the ordering in which the primitive functions were executed. However, we can do better by noting two interesting aspects:

What we want to compute:

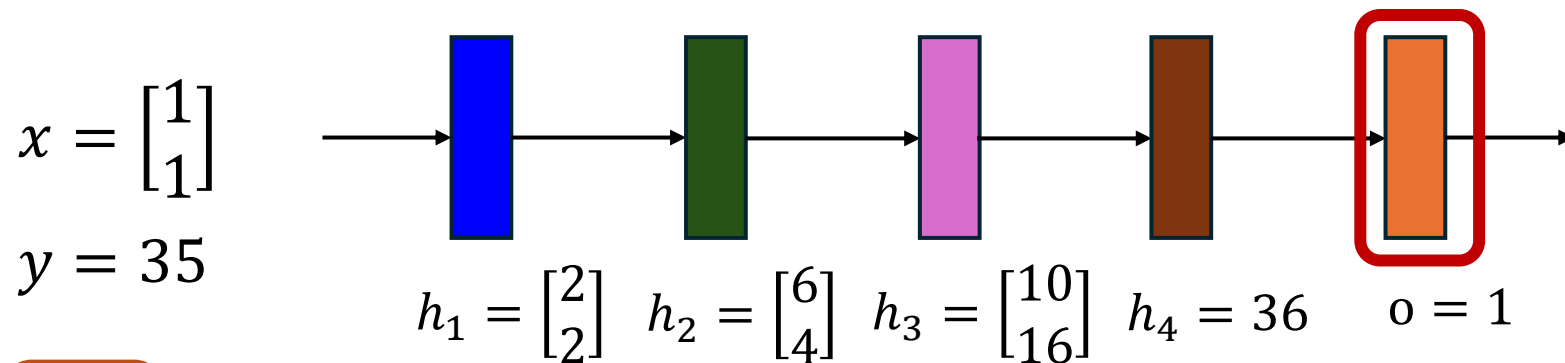
$$\frac{\partial \mathbf{h}_l}{\partial \mathbf{W}_1} = \frac{\partial \mathbf{h}_l}{\partial \mathbf{h}_{l-1}} \cdot \frac{\partial \mathbf{h}_{l-1}}{\partial \mathbf{h}_{l-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial \mathbf{W}_1}$$

$$\frac{\partial \mathbf{h}_l}{\partial \mathbf{W}_2} = \frac{\partial \mathbf{h}_l}{\partial \mathbf{h}_{l-1}} \cdot \frac{\partial \mathbf{h}_{l-1}}{\partial \mathbf{h}_{l-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{W}_2}$$

\vdots

The leftmost term is a product between a vector and a matrix (which is a consequence of having a scalar term in output), which is computationally better than a product between two matrices.

Automatic Differentiation: Reverse Mode AD



$$\begin{aligned} \frac{\partial l}{\partial W^{(1)}} &= \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial W^{(1)}} \\ \frac{\partial l}{\partial W^{(2)}} &= \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial W^{(2)}} \\ \frac{\partial l}{\partial W^{(3)}} &= \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial W^{(3)}} \\ \frac{\partial l}{\partial W^{(4)}} &= \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial W^{(4)}} \end{aligned}$$

Ini adalah contoh bagian yang cukup dikomputasi sekali lalu disimpan dan digabung untuk komputasi selanjutnya.

Automatic Differentiation: Reverse Mode AD

$$\mathbf{h}_1 = f_1(\mathbf{x}, \mathbf{W}_1)$$

$$\mathbf{h}_2 = f_2(\mathbf{h}_1, \mathbf{W}_2)$$

\vdots

$$\mathbf{h}_{l-1} = f_{l-1}(\mathbf{h}_{l-2}, \mathbf{W}_{l-1})$$

$$L = l(\mathbf{h}_{l-1}, \mathbf{W}_l)$$

Step 1: Forward Pass

Differently from F-AD, we start by executing the entire program to be differentiated, **storing all intermediate outputs**.

Step 2: Backward Pass

Initialize a vector $\tilde{\mathbf{h}} = \frac{\partial L}{\partial \mathbf{h}_{l-1}}$

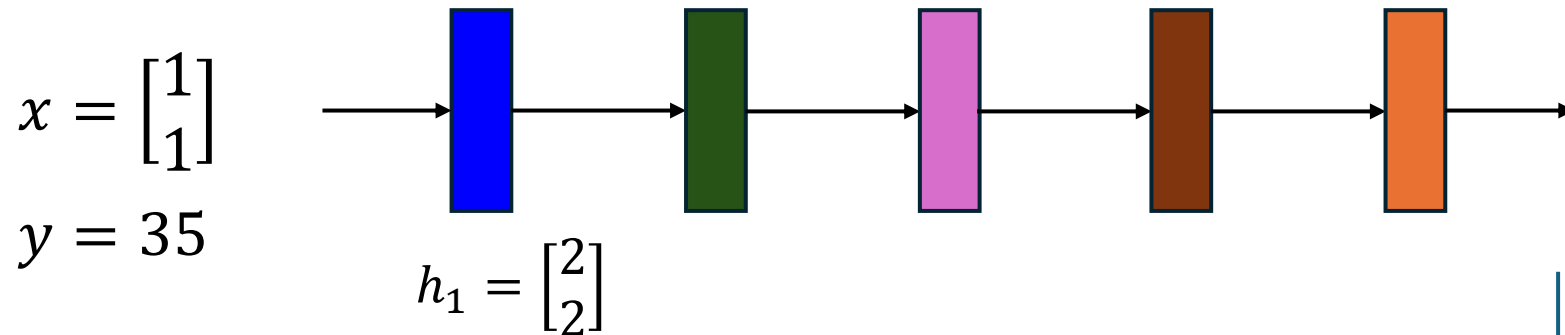
For index i ranging in $l, l-1, l-2, \dots, 2, 1$:

$$\frac{\partial L}{\partial \mathbf{W}^{(i)}} = \tilde{\mathbf{h}} \times \frac{\partial \mathbf{h}_i}{\partial \mathbf{W}^{(i)}}$$

$$\tilde{\mathbf{h}} \leftarrow \tilde{\mathbf{h}} \times \frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}}$$

We update our "back-propagated" input Jacobian.

Automatic Differentiation: Reverse Mode AD



Storage: [| | | |]

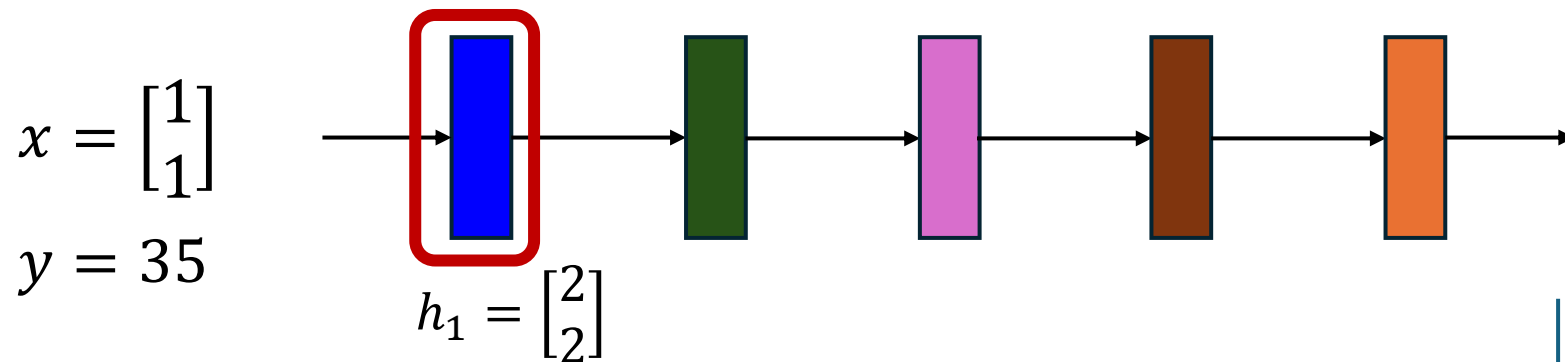
$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

Automatic Differentiation: Reverse Mode AD



Forward Pass

$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

Storage: $h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$

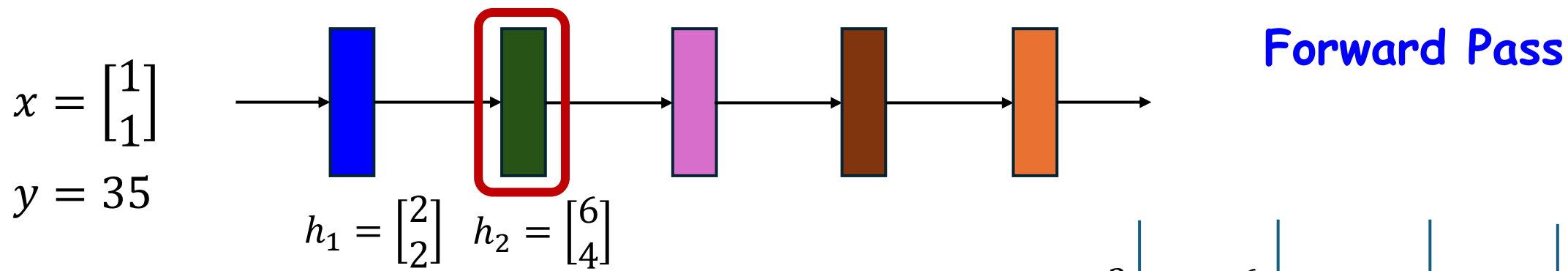
$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

$\begin{bmatrix} \\ \\ \end{bmatrix}$

Automatic Differentiation: Reverse Mode AD



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

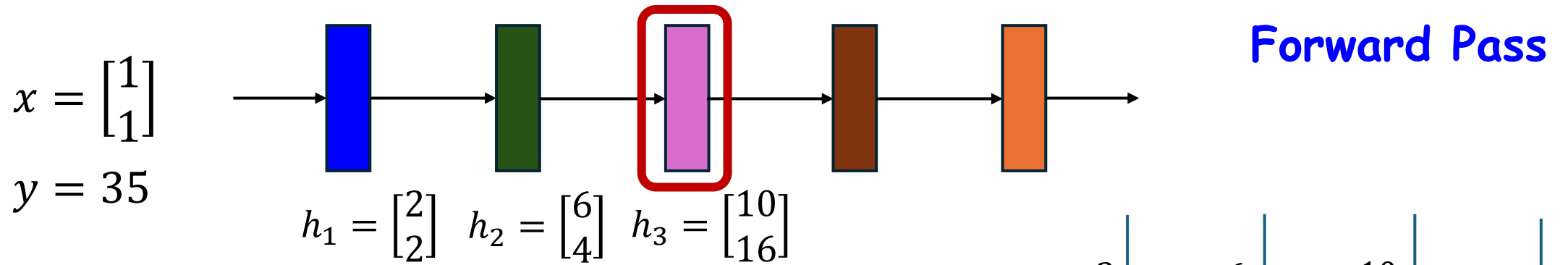
$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

Storage: $\left[h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \mid h_2 = \begin{bmatrix} 6 \\ 4 \end{bmatrix} \mid \mid \mid \right]$

Automatic Differentiation: Reverse Mode AD



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

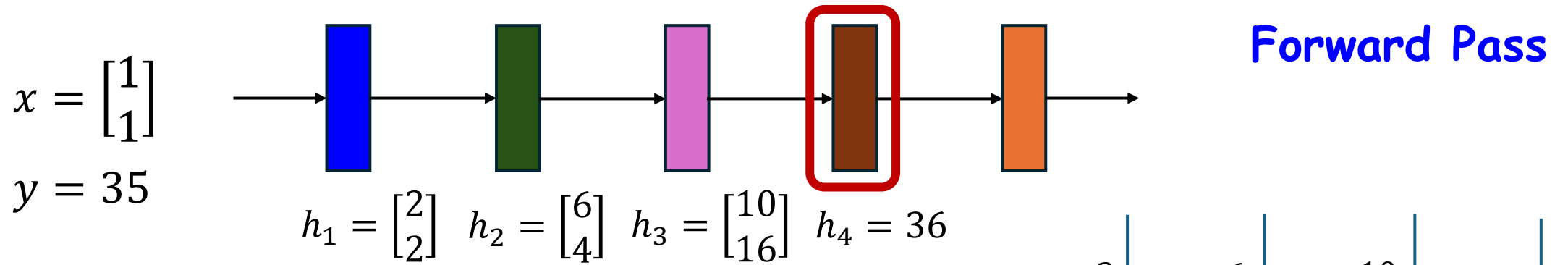
$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

Storage: $\left[h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \mid h_2 = \begin{bmatrix} 6 \\ 4 \end{bmatrix} \mid h_3 = \begin{bmatrix} 10 \\ 16 \end{bmatrix} \mid \right]$

Automatic Differentiation: Reverse Mode AD



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

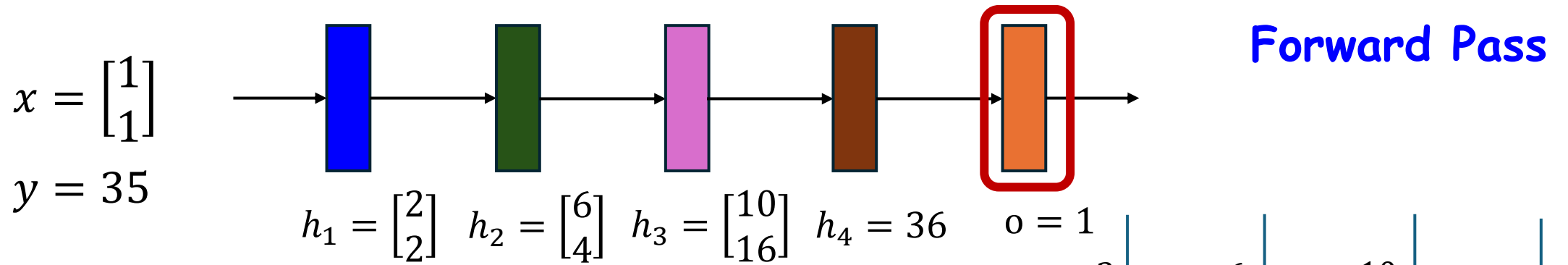
$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

Storage: $\left[h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \middle| h_2 = \begin{bmatrix} 6 \\ 4 \end{bmatrix} \middle| h_3 = \begin{bmatrix} 10 \\ 16 \end{bmatrix} \middle| h_4 = 36 \right]$

Automatic Differentiation: Reverse Mode AD



$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial W^{(3)}}$$

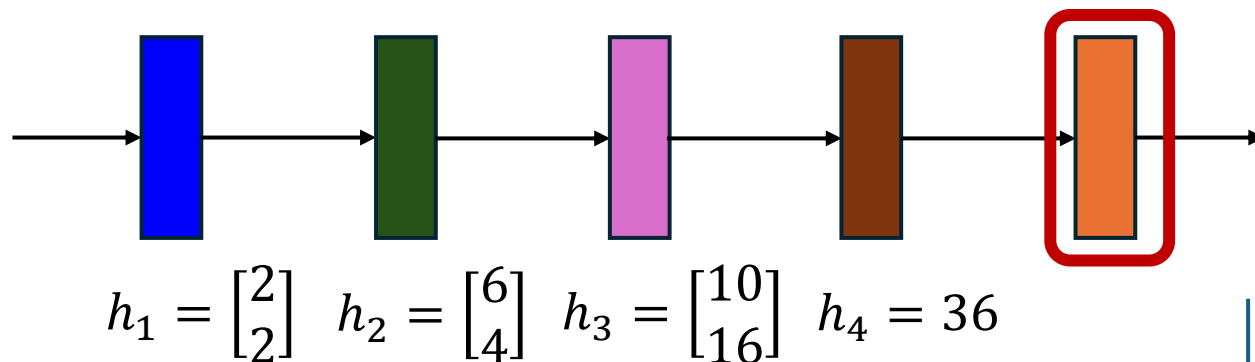
$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \cdot \frac{\partial h_4}{\partial W^{(4)}}$$

Storage: $\left[h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \right] h_2 = \begin{bmatrix} 6 \\ 4 \end{bmatrix} \left[h_3 = \begin{bmatrix} 10 \\ 16 \end{bmatrix} \right] h_4 = 36 \left[o = 1 \right]$

Automatic Differentiation: Reverse Mode AD

$$x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$y = 35$$



Backward Pass

$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial W^{(1)}}$$

$$\tilde{h} \leftarrow \partial l / \partial h_4$$

Storage: $[h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} | h_2 = \begin{bmatrix} 6 \\ 4 \end{bmatrix} | h_3 = \begin{bmatrix} 10 \\ 16 \end{bmatrix} | h_4 = 36 | o = 1]$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial W^{(2)}}$$

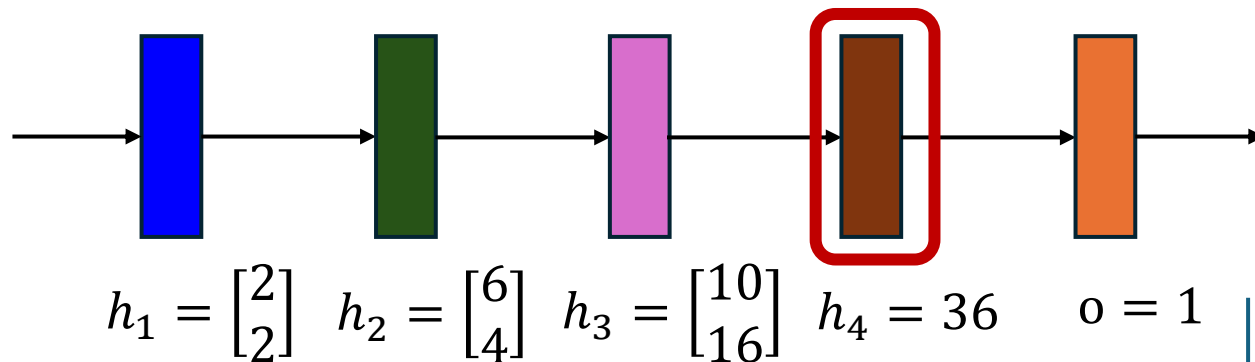
$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial W^{(4)}}$$

Automatic Differentiation: Reverse Mode AD

$$x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$y = 35$$



Backward Pass

$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial W^{(3)}}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial W^{(4)}} \text{ DONE!}$$

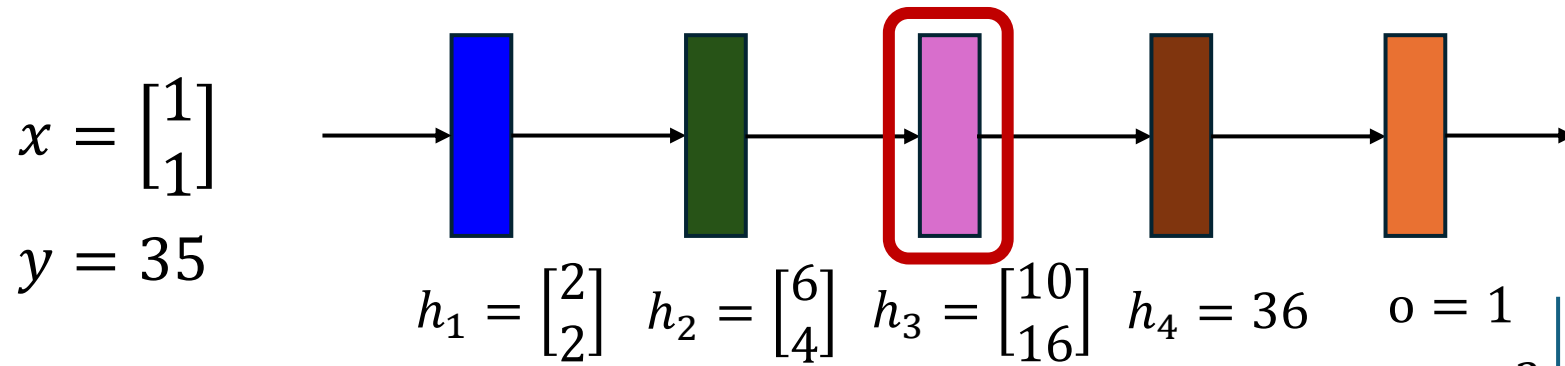
Storage: $[h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \mid h_2 = \begin{bmatrix} 6 \\ 4 \end{bmatrix} \mid h_3 = \begin{bmatrix} 10 \\ 16 \end{bmatrix} \mid h_4 = 36 \mid o = 1]$

$$\tilde{h} \leftarrow \partial l / \partial h_4$$

$$\partial_{W^{(4)}} l = \tilde{h} \times \partial h_4 / \partial W^{(4)}$$

$$\tilde{h} \leftarrow \tilde{h} \times \partial h_4 / \partial h_3$$

Automatic Differentiation: Reverse Mode AD



Backward Pass

$$\frac{\partial l}{\partial W^{(1)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial W^{(1)}}$$

$$\frac{\partial l}{\partial W^{(2)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial W^{(2)}}$$

$$\frac{\partial l}{\partial W^{(3)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial W^{(3)}} \text{ DONE!}$$

$$\frac{\partial l}{\partial W^{(4)}} = \frac{\partial l}{\partial h_4} \frac{\partial h_4}{\partial W^{(4)}} \text{ DONE!}$$

Storage: $\left[h_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \mid h_2 = \begin{bmatrix} 6 \\ 4 \end{bmatrix} \mid h_3 = \begin{bmatrix} 10 \\ 16 \end{bmatrix} \mid h_4 = 36 \mid o = 1 \right]$

$$\tilde{h} \leftarrow \partial l / \partial h_4$$

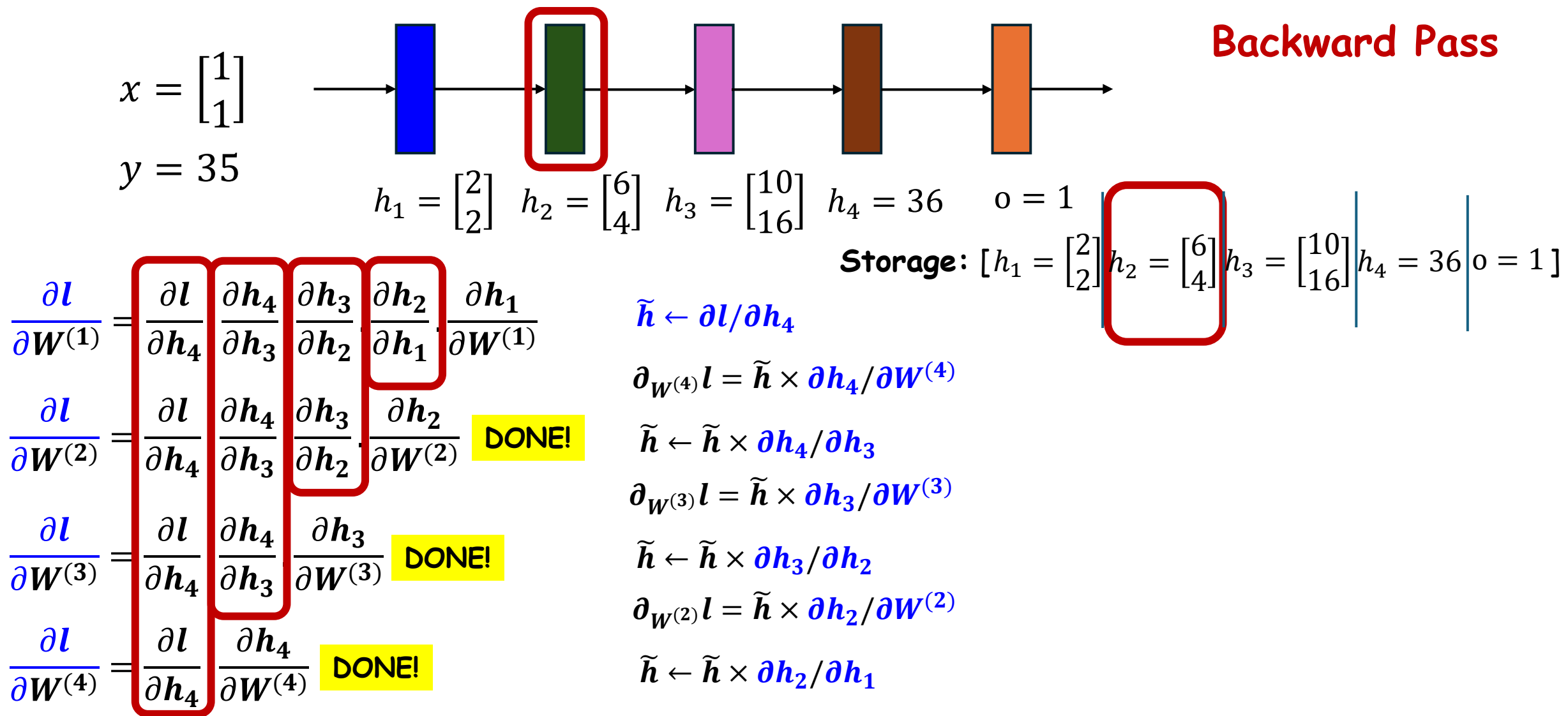
$$\partial_{W^{(4)}} l = \tilde{h} \times \partial h_4 / \partial W^{(4)}$$

$$\tilde{h} \leftarrow \tilde{h} \times \partial h_4 / \partial h_3$$

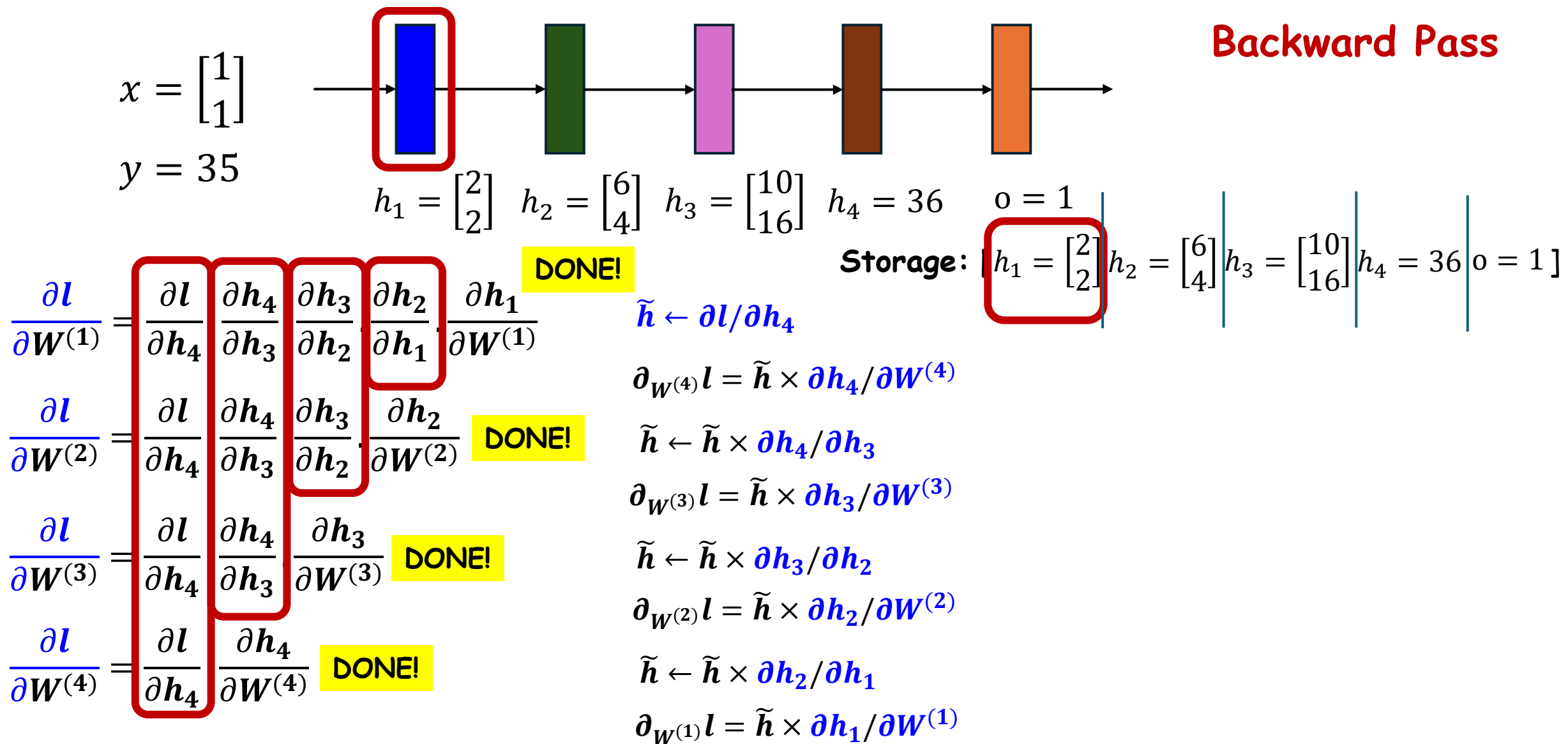
$$\partial_{W^{(3)}} l = \tilde{h} \times \partial h_3 / \partial W^{(3)}$$

$$\tilde{h} \leftarrow \tilde{h} \times \partial h_3 / \partial h_2$$

Automatic Differentiation: Reverse Mode AD



Automatic Differentiation: Reverse Mode AD



Automatic Differentiation

Computationally, Reverse-Mode AD is significantly more efficient than Forward-Mode AD.

In particular, both operations in **backward steps** at Reverse-Mode AD are **vector-matrix products** scaling only linearly in all shape quantities.

The tradeoff is that executing **Reverse-Mode AD requires a large amount of memory**, since all intermediate values of the primal program must be stored on disk with a suitable strategy

Automatic Differentiation

Specific techniques, such as **gradient checkpointing**, can be used to improve on this tradeoff by increasing computations and partially reducing the memory requirements.

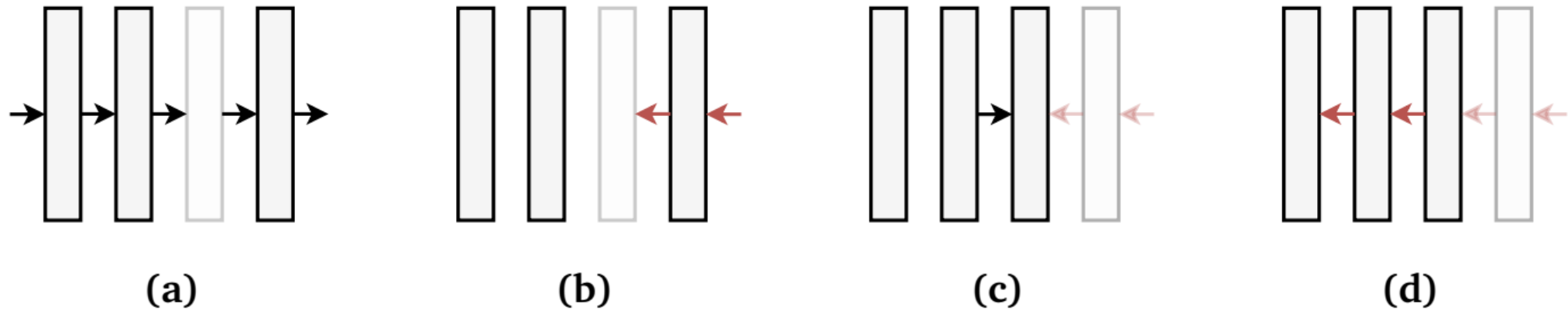


Figure F.6.1: An example of **gradient checkpointing**. (a) We execute a forward pass, but we only store the outputs of the first, second, and fourth blocks (**checkpoints**). (b) The backward pass (red arrows) stops at the third block, whose activations are not available. (c) We run a second forward pass starting from the closest checkpoint to materialize again the activations. (d) We complete the forward pass. Compared to a standard backward pass, this requires 1.25x more computations. In general, the less checkpoints are stored, the higher the computational cost of the backward pass.

Vector-Jacobian Products

Looking at the backward step in the R-AD algorithm, we can make an interesting observation: the only operation we need is a product between a row vector \mathbf{v} and a Jacobian of \mathbf{f} (either the input or the weight Jacobian). We call these two operations the vector-Jacobian products (VJPs) of $\mathbf{f}(\mathbf{x})$.

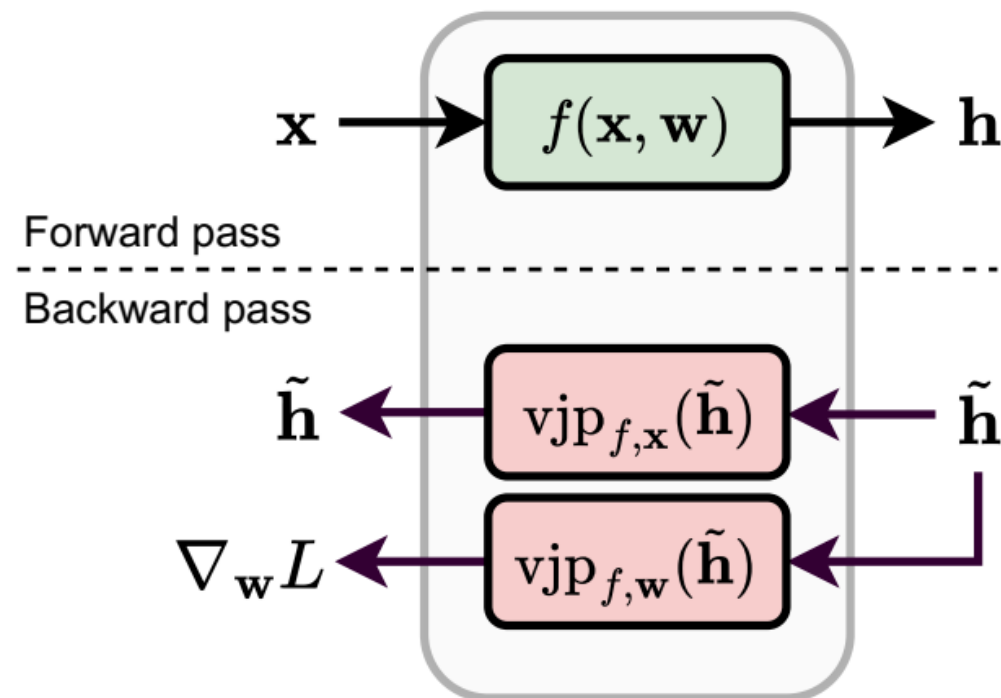
Definition D.6.2 (Vector-Jacobian product (VJP)) Given a function $\mathbf{y} = \mathbf{f}(\mathbf{x})$, with $\mathbf{x} \sim (c)$ and $\mathbf{y} \sim (c')$, its VJP is another function defined as:

$$\text{vjp}_f(\mathbf{v}) = \mathbf{v}^\top \partial \mathbf{f}(\mathbf{x}) \quad (\text{E.6.4})$$

where $\mathbf{v} \sim (c')$. If f has multiple parameters $\mathbf{f}(\mathbf{x}_1, \dots, \mathbf{x}_n)$, we can define n individual VJPs denoted as $\text{vjp}_{f, \mathbf{x}_1}(\mathbf{v})$, ..., $\text{vjp}_{f, \mathbf{x}_n}(\mathbf{v})$.

Vector-Jacobian Products: How to implement R-AD?

Figure F.6.2: For performing R-AD, primitives must be augmented with two VJP operations to be able to perform a backward pass, corresponding to the input VJP (E.6.5) and the weight VJP (E.6.6). One call for each is sufficient to perform the backward pass through the primitive, corresponding to (E.6.7)-(E.6.8).

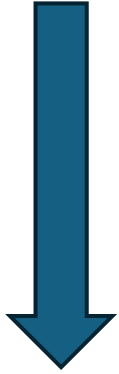


$$\text{vjp}_{f,\mathbf{x}}(\mathbf{v}) = \mathbf{v}^\top \partial_{\mathbf{x}} f(\mathbf{x}, \mathbf{w})$$
$$\text{vjp}_{f,\mathbf{w}}(\mathbf{v}) = \mathbf{v}^\top \partial_{\mathbf{w}} f(\mathbf{x}, \mathbf{w})$$

Contoh implementasi sederhana Reverse-Mode AD

Contoh kode dari A. Karpathy
<https://github.com/karpathy/micrograd>

$$f(a, b) = a^2 + 2ab + b^2$$



$$g(a, b) = a + b$$

$$h(g) = g^2$$

$$f(a, b) = h(g(a, b))$$

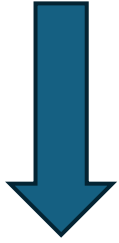
$$\frac{\partial f(a, b)}{\partial a} = 2a + 2b$$

$$\frac{\partial f(a, b)}{\partial b} = 2a + 2b$$

$$\frac{\partial f(a, b)}{\partial a} = \frac{\partial h}{\partial a} = \frac{\partial h}{\partial g} \cdot \frac{\partial g}{\partial a}$$

$$\frac{\partial f(a, b)}{\partial b} = \frac{\partial h}{\partial b} = \frac{\partial h}{\partial g} \cdot \frac{\partial g}{\partial b}$$

$$f(a, b) = a^2 + 2ab + b^2$$



```
a = Value(...)  
b = Value(...)  
c = a + b  
d = c ** 2  
d.backward()  
print(a.grad)      #  $\partial d / \partial a$   
print(b.grad)      #  $\partial d / \partial b$ 
```

<https://github.com/karpathy/micrograd>

```
class Value:
    """ stores a single scalar value and its gradient """

    def __init__(self, data, _children=()):
        self.data = data
        self.grad = 0
        # internal variables used for autograd graph construction
        self._backward = lambda: None
        self._prev = set(_children)

    def __add__(self, other):
        other = other if isinstance(other, Value) else Value(other)
        out = Value(self.data + other.data, (self, other), '+')

        def _backward():
            self.grad += out.grad
            other.grad += out.grad
            out._backward = _backward

        return out
```

<https://github.com/karpathy/micrograd>

```
def __mul__(self, other):  
    other = other if isinstance(other, Value) else Value(other)  
    out = Value(self.data * other.data, (self, other))
```

```
def _backward():  
    self.grad += other.data * out.grad  
    other.grad += self.data * out.grad  
    out._backward = _backward
```

```
    return out
```

```
def __pow__(self, other):  
    assert isinstance(other, (int, float)), "only supporting int/float powers for now"  
    out = Value(self.data**other, (self,))
```

```
def _backward():  
    self.grad += (other * self.data**(other-1)) * out.grad  
    out._backward = _backward
```

```
    return out
```

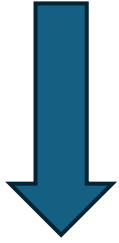
<https://github.com/karpathy/micrograd>

```
def backward(self):

    # topological order all of the children in the graph
    topo = []
    visited = set()
    def build_topo(v):
        if v not in visited:
            visited.add(v)
            for child in v._prev:
                build_topo(child)
            topo.append(v)
    build_topo(self)

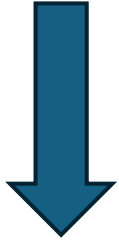
    # go one variable at a time and apply the chain rule to get its gradient
    self.grad = 1
    for v in reversed(topo):
        v._backward()
```

$$f(a, b) = a^2 + 2ab + b^2$$



```
a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 
```

$$f(a, b) = a^2 + 2ab + b^2$$

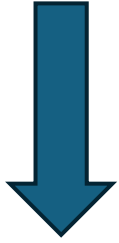


```
a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)    #  $\partial d / \partial a$ 
print(b.grad)    #  $\partial d / \partial b$ 
```

```
_prev = ()
data = 2
grad = 0
_backward =
    lambda: None
```

a

$$f(a, b) = a^2 + 2ab + b^2$$



```
a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)    #  $\partial d / \partial a$ 
print(b.grad)    #  $\partial d / \partial b$ 
```

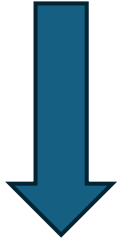
```
_prev = ()
data = 2
grad = 0
_backward =
    lambda: None
```

a

```
_prev = ()
data = 3
grad = 0
_backward =
    lambda: None
```

b

$$f(a, b) = a^2 + 2ab + b^2$$



```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```

a

```

_prev = ()
data = 2
grad = 0
_backward =
    lambda: None

```

b

```

_prev = ()
data = 3
grad = 0
_backward =
    lambda: None

```

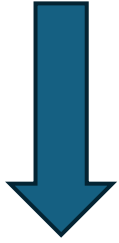
c

```

_prev = (a, b)
data = a.data + b.data = 2 + 3 = 5
grad = 0
_backward =
    lambda: a.grad += self.grad
           b.grad += self.grad

```

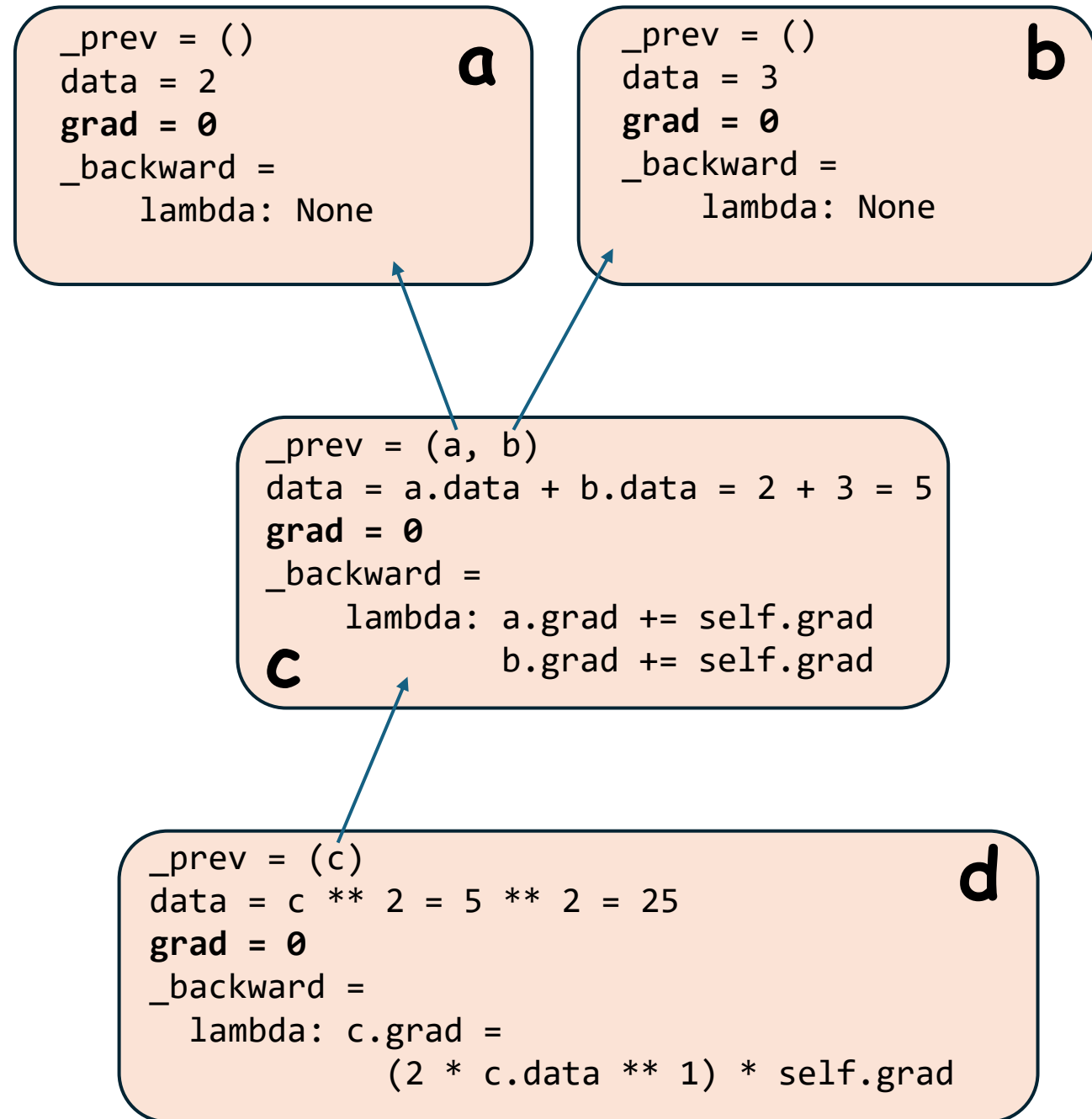

$$f(a, b) = a^2 + 2ab + b^2$$



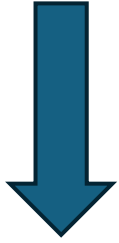
```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```



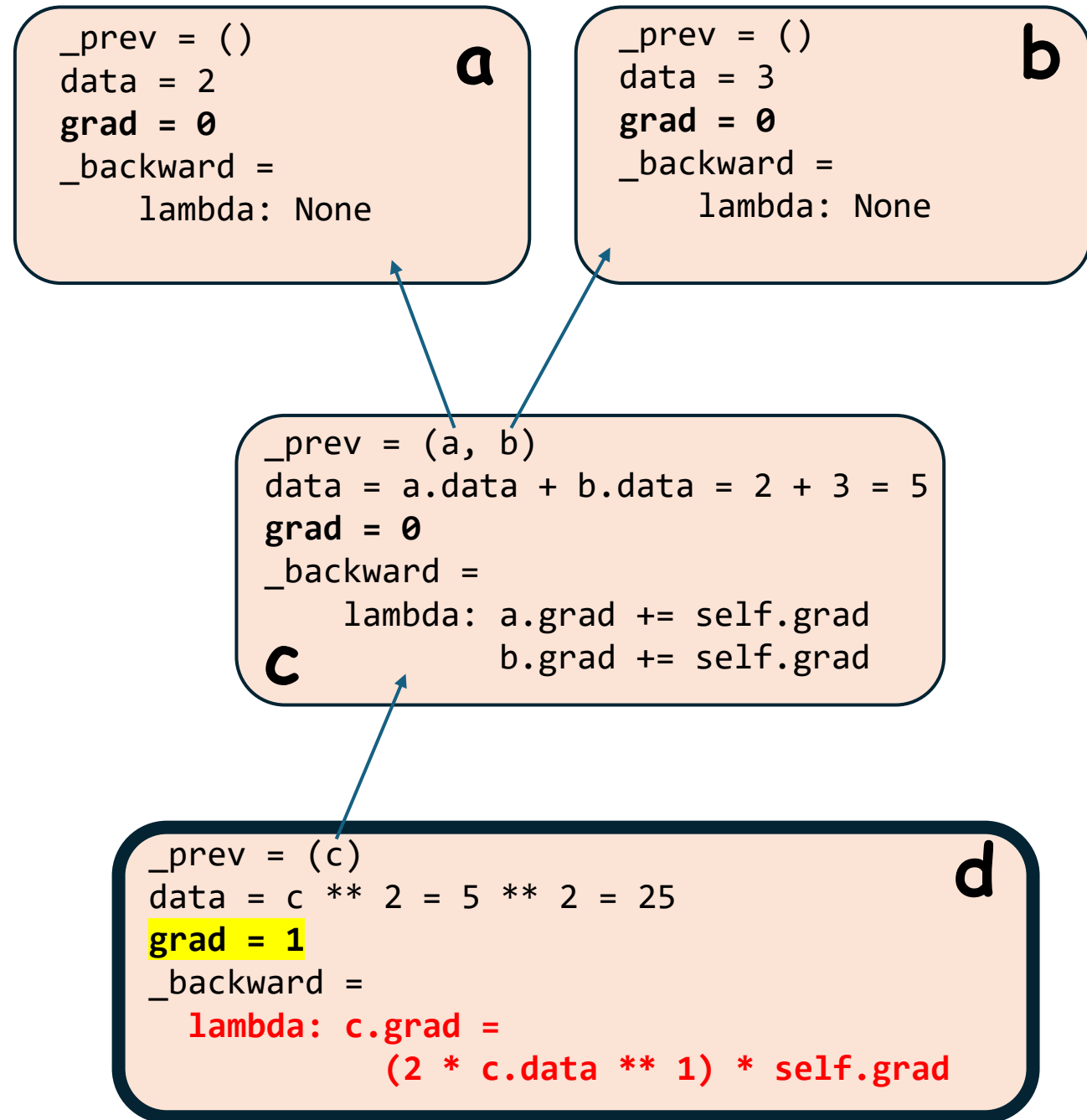
$$f(a, b) = a^2 + 2ab + b^2$$



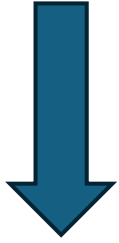
```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```



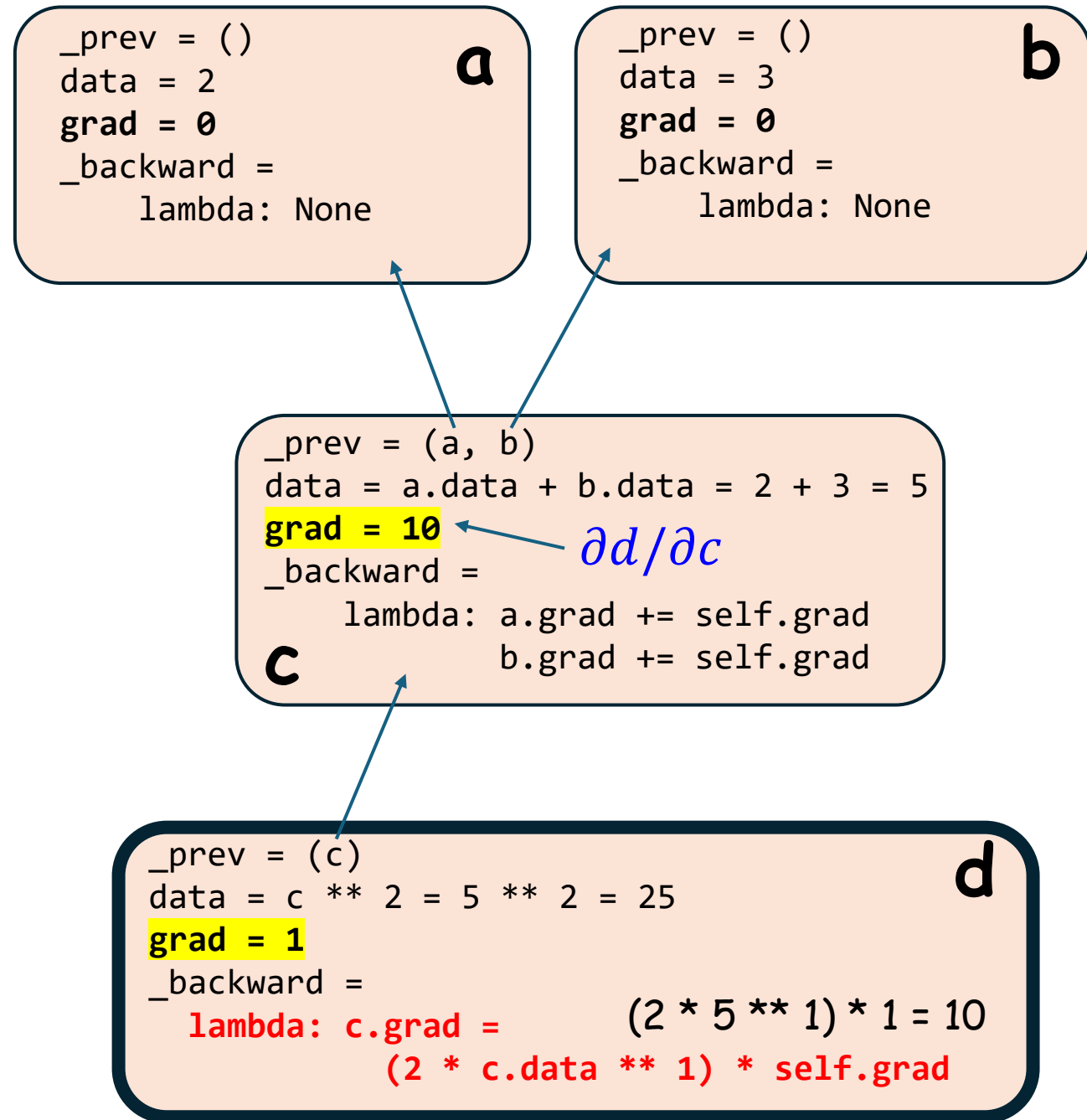
$$f(a, b) = a^2 + 2ab + b^2$$



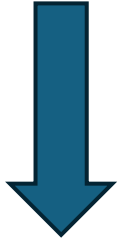
```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```



$$f(a, b) = a^2 + 2ab + b^2$$



```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```

a

```

_prev = ()
data = 2
grad = 0
_backward =
    lambda: None

```

b

```

_prev = ()
data = 3
grad = 0
_backward =
    lambda: None

```

c

```

_prev = (a, b)
data = a.data + b.data = 2 + 3 = 5
grad = 10
_backward =
    lambda: a.grad += self.grad
           b.grad += self.grad

```

$\partial d / \partial c$ points to `grad = 10`

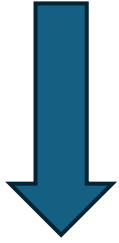
d

```

_prev = (c)
data = c ** 2 = 5 ** 2 = 25
grad = 1
_backward =
    lambda: c.grad = (2 * 5 ** 1) * 1 = 10
                (2 * c.data ** 1) * self.grad

```

$$f(a, b) = a^2 + 2ab + b^2$$



```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```

a

```

_prev = ()
data = 2
grad = 10
_backward =
lambda: None

```

$\partial d / \partial a$

b

```

_prev = ()
data = 3
grad = 10
_backward =
lambda: None

```

$\partial d / \partial b$

c

```

_prev = (a, b)
data = a.data + b.data = 2 + 3 = 5
grad = 10
_backward =
lambda: a.grad += self.grad
        b.grad += self.grad

```

$\partial d / \partial c$

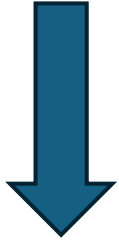
d

```

_prev = (c)
data = c ** 2 = 5 ** 2 = 25
grad = 1
_backward =
lambda: c.grad = (2 * 5 ** 1) * 1 = 10
        (2 * c.data ** 1) * self.grad

```

$$f(a, b) = a^2 + 2ab + b^2$$



```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```

a

```

_prev = ()
data = 2
grad = 10
_backward =
lambda: None

```

$\partial d / \partial a$

b

```

_prev = ()
data = 3
grad = 10
_backward =
lambda: None

```

$\partial d / \partial b$

c

```

_prev = (a, b)
data = a.data + b.data = 2 + 3 = 5
grad = 10
_backward =
lambda: a.grad += self.grad
        b.grad += self.grad

```

$\partial d / \partial c$

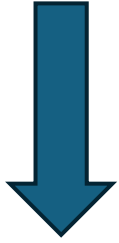
d

```

_prev = (c)
data = c ** 2 = 5 ** 2 = 25
grad = 1
_backward =
lambda: c.grad = (2 * 5 ** 1) * 1 = 10
        (2 * c.data ** 1) * self.grad

```

$$f(a, b) = a^2 + 2ab + b^2$$



```

a = Value(2)
b = Value(3)
c = a + b
d = c ** 2
d.backward()
print(a.grad)      #  $\partial d / \partial a$ 
print(b.grad)      #  $\partial d / \partial b$ 

```

a

```

_prev = ()
data = 2
grad = 10
_backward =
  lambda: None

```

$\partial d / \partial a$

b

```

_prev = ()
data = 3
grad = 10
_backward =
  lambda: None

```

$\partial d / \partial b$

c

```

_prev = (a, b)
data = a.data + b.data = 2 + 3 = 5
grad = 10
_backward =
  lambda: a.grad += self.grad
         b.grad += self.grad

```

$\partial d / \partial c$

d

```

_prev = (c)
data = c ** 2 = 5 ** 2 = 25
grad = 1
_backward =
  lambda: c.grad = (2 * 5 ** 1) * 1 = 10
             (2 * c.data ** 1) * self.grad

```