Backpropagation and Anomaly Detection

CSIT375 AI for Cybersecurity

Dr Wei Zong

SCIT University of Wollongong

Disclaimer: The presentation materials come from various sources. For further information, check the references section

Outline

- Neural networks
    - Model representation (Last week)
    - Computation graph
    - Back propagation

- What are anomalies
- Types of anomalies
- Applications of anomaly detection
- Machine learning for intrusion detection
- Anomaly detection techniques

The Flow of Computations in Neural Networks

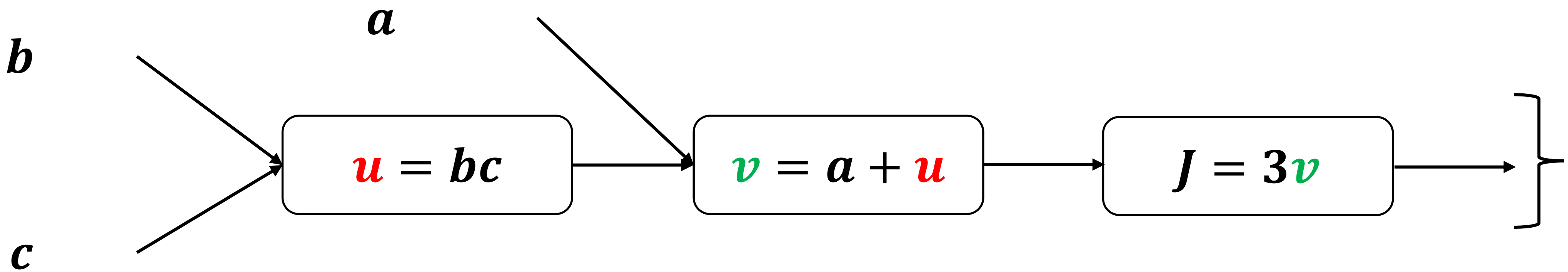
- The flow of computations in a neural network goes in two ways:
    1. Left-to-right: This is referred to as *forward propagation*, which results in computing the output of the network
    2. Right-to-left: This is referred to as *back propagation*, which results in computing the gradients (or derivatives) of the parameters in the network

- The intuition behind this 2-way flow of computations can be explained through the concept of "computation graphs"
    - What is a computation graph?

What is a Computation Graph?

- Let us assume we want to compute the following function J:

$$\left.\begin{array}{c} J(a,b,c) = 3\underbrace{(\underbrace{a+\underbrace{bc}_{u}}_{v})}_{J} \end{array}\right\} \quad \begin{aligned} u &= bc \\ v &= a+u \\ J &= 3v \end{aligned}$$



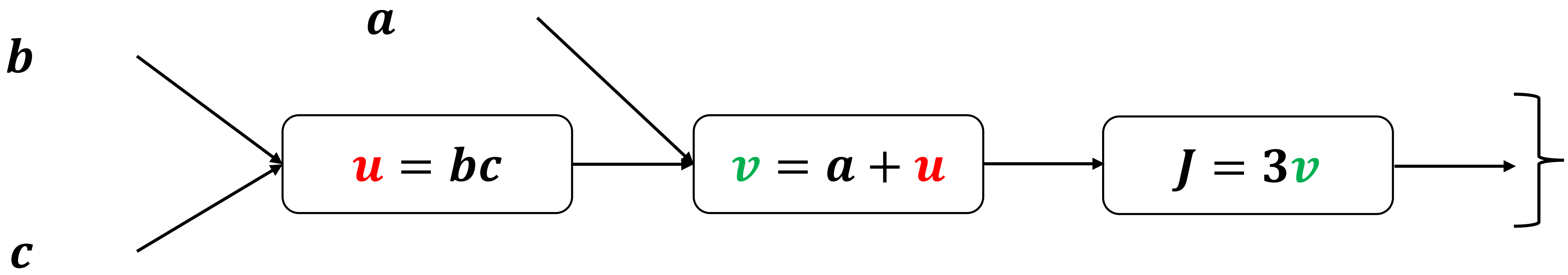
Computation Graph

Forward Propagation

- Let us assume we want to compute the following function J:

$$J(a,b,c) = 3\underbrace{(\underbrace{a+\underbrace{bc}_{u}}_{v})}_{J}$$

$$\left.\begin{aligned} u &= bc \\ v &= a + u \\ J &= 3v \end{aligned}\right.$$



b

c

a

$u = bc$

$v = a + u$

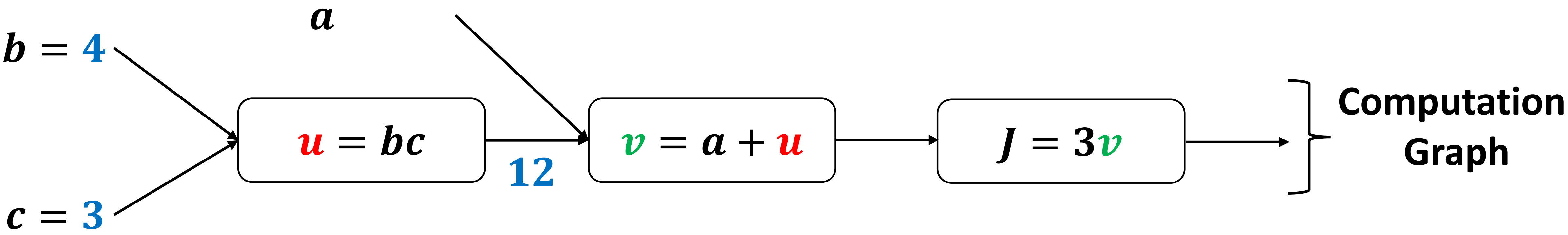
$J = 3v$

Computation Graph

Forward Propagation

- Let us assume we want to compute the following function J:

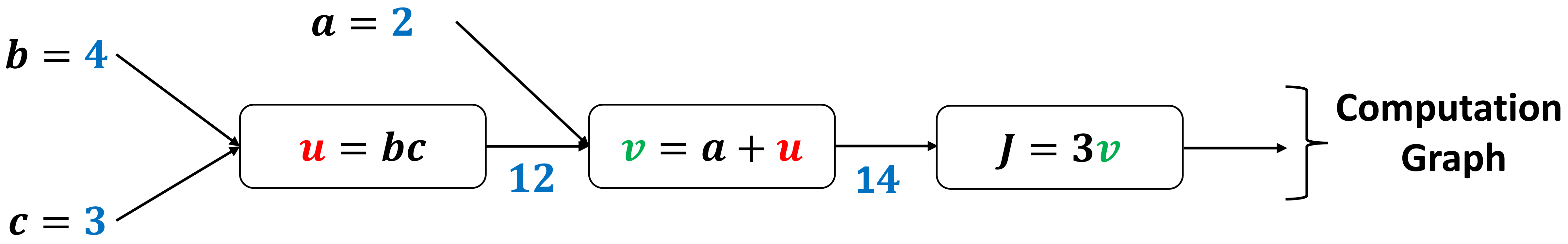
$$\left.\underbrace{J(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) = 3(\underbrace{\boldsymbol{a}+\underbrace{\boldsymbol{bc}}_{\boldsymbol{u}}}_{\boldsymbol{v}})}_{\boldsymbol{J}}\right\} \quad \begin{aligned} \boldsymbol{u} &= \boldsymbol{bc} \\ \boldsymbol{v} &= \boldsymbol{a}+\boldsymbol{u} \\ \boldsymbol{J} &= 3\boldsymbol{v} \end{aligned}$$



Forward Propagation

- Let us assume we want to compute the following function J:

$$\left.J(a,b,c) = 3\underbrace{(\underbrace{a+\underbrace{bc}_{u}}_{v})}_{J}\right\}\qquad \begin{aligned} u &= bc \\ v &= a+u \\ J &= 3v \end{aligned}$$



Forward Propagation

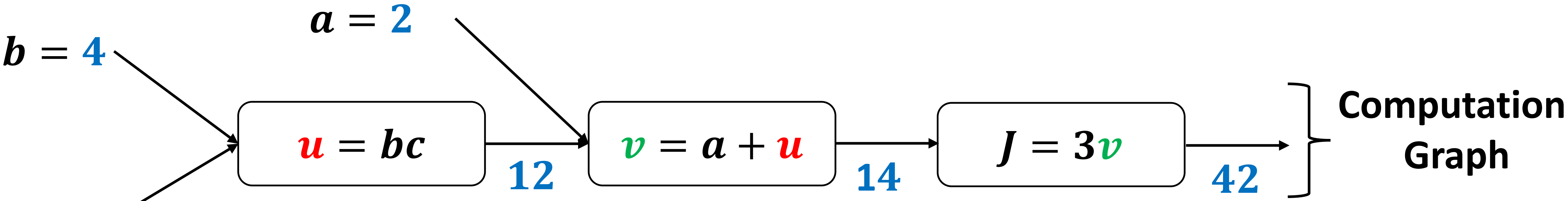
- Let us assume we want to compute the following function J:

$$\underbrace{J(a,b,c) = 3(\underbrace{a + \underbrace{bc}_{u}}_{v})}_{J}$$

$$u = bc$$

$$v = a + u$$

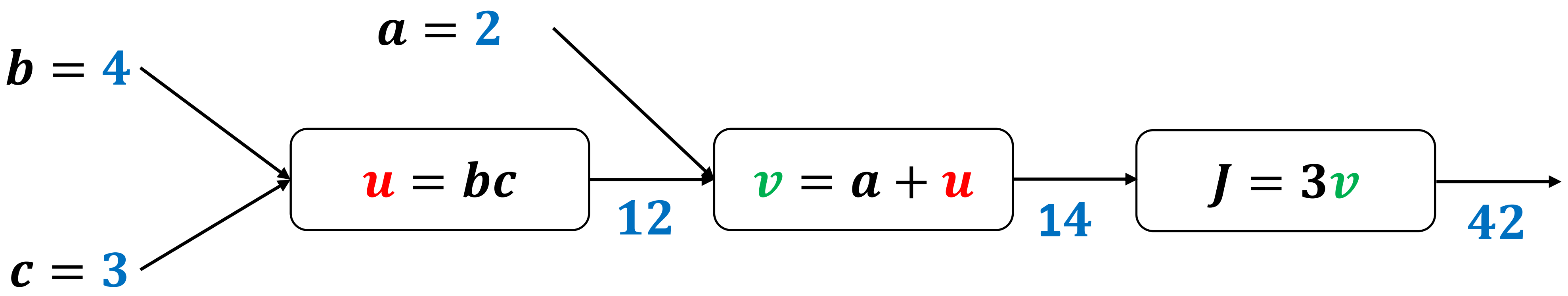
$$J = 3v$$



Forward propagation allows computing J

Backward Propagation

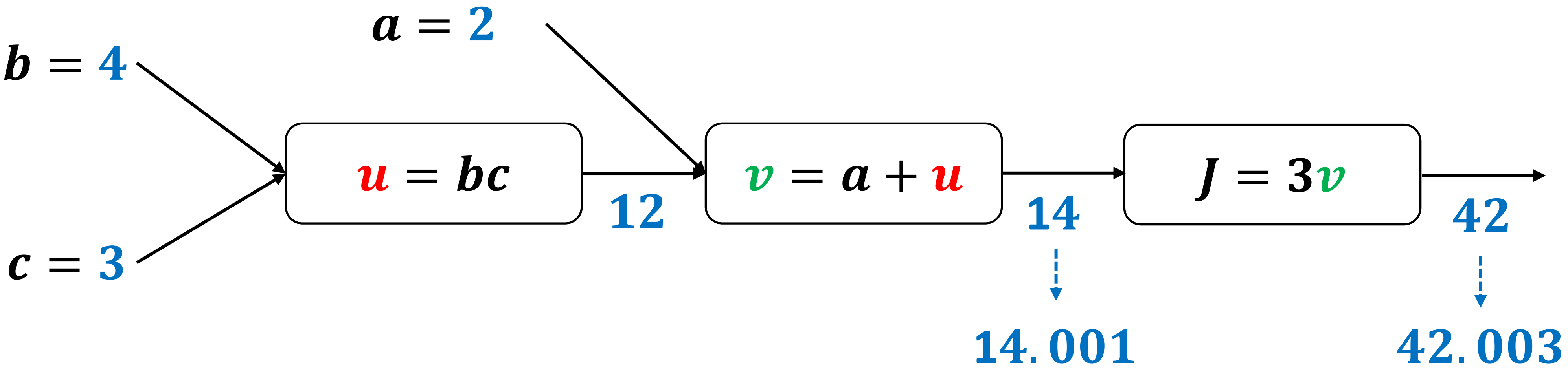
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{dv} = \text{Derivative of } J \text{ with respect to } v$$

Backward Propagation

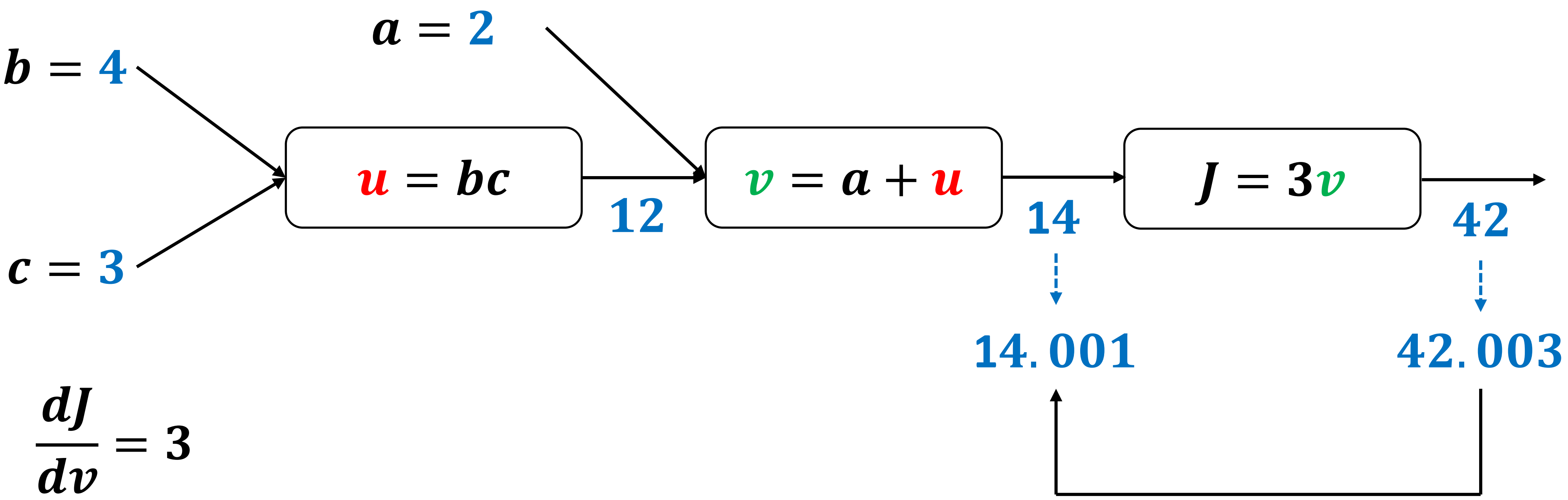
- Let us now compute the derivatives of the variables through the computation graph as follows:



$\frac{dJ}{dv}$ = If we change v a little bit, how would J change?

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

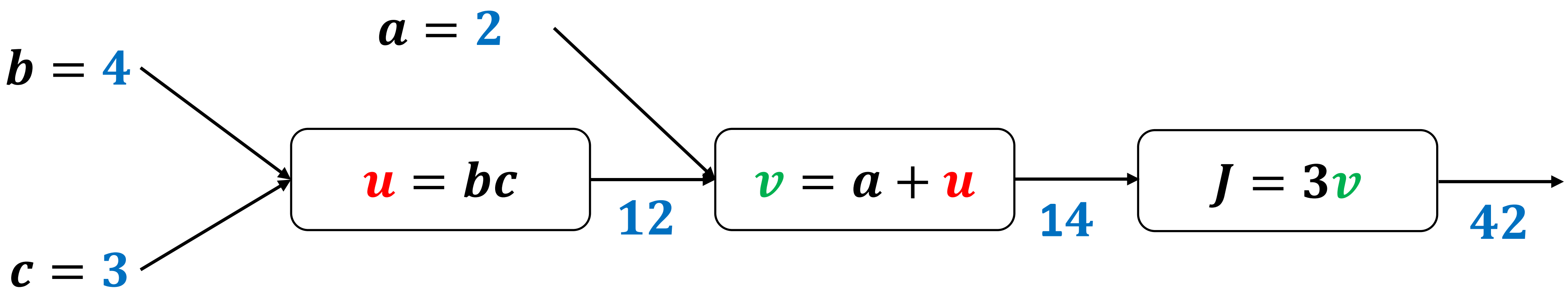


$$\frac{dJ}{dv} = 3$$

To compute the derivative of J with respect to v, we went *back* to v, nudged it, and measured the corresponding resultant increase on J

Backward Propagation

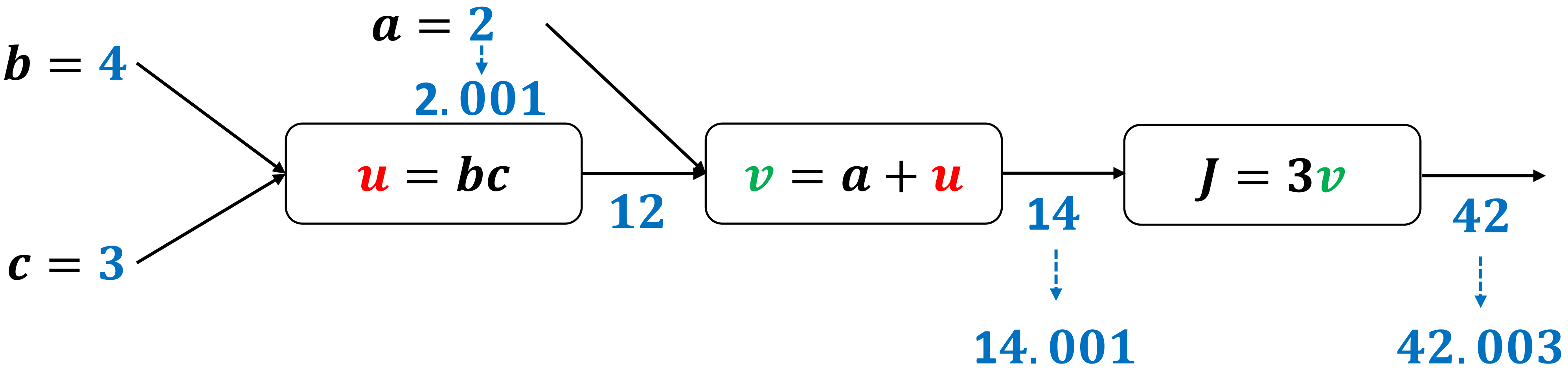
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{da} = \text{Derivative of } J \text{ with respect to } a$$

Backward Propagation

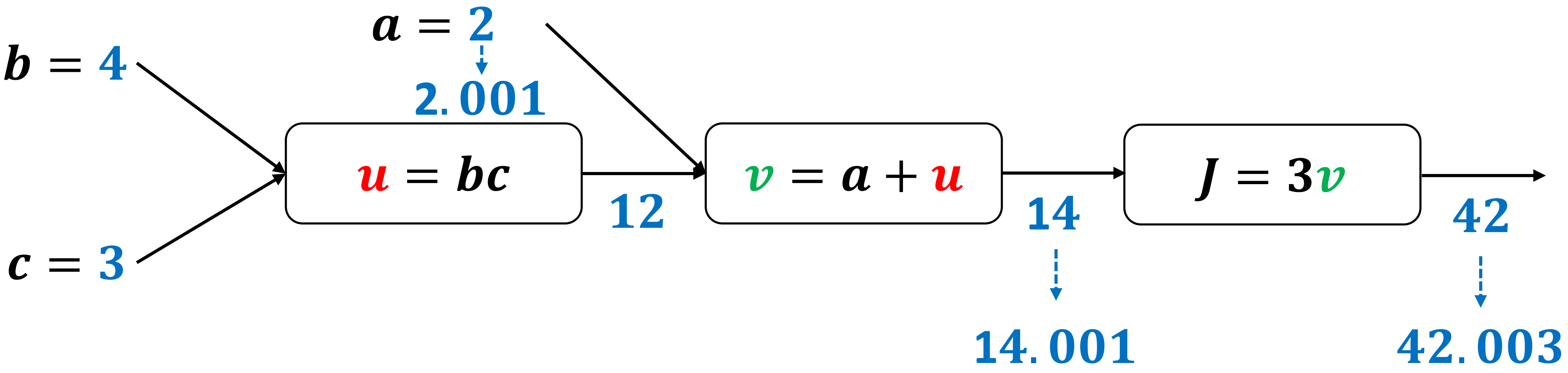
- Let us now compute the derivatives of the variables through the computation graph as follows:



$\frac{dJ}{da}$ = If we change a a little bit, how would J change?

Backward Propagation

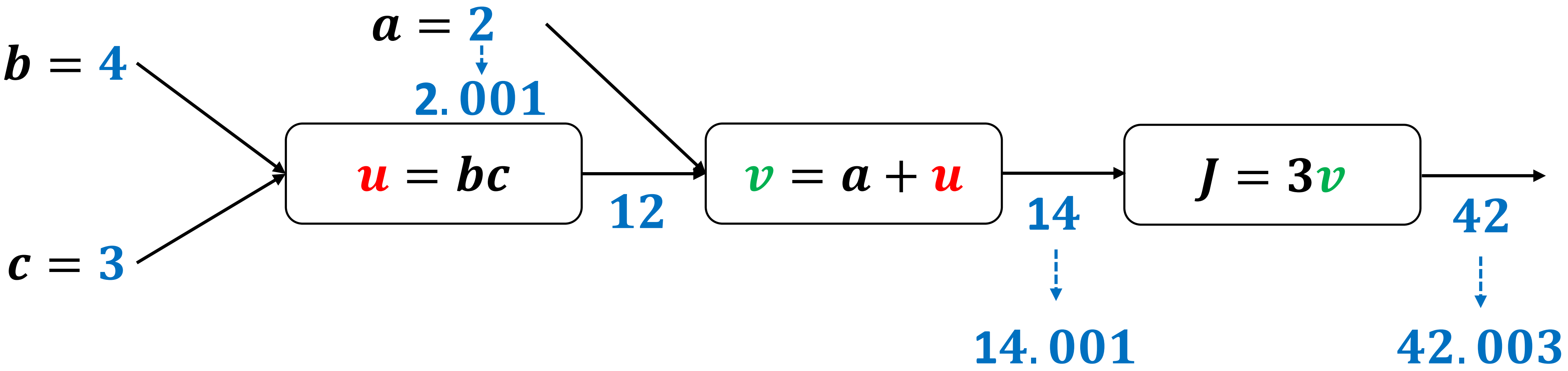
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{da} = 3$$

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

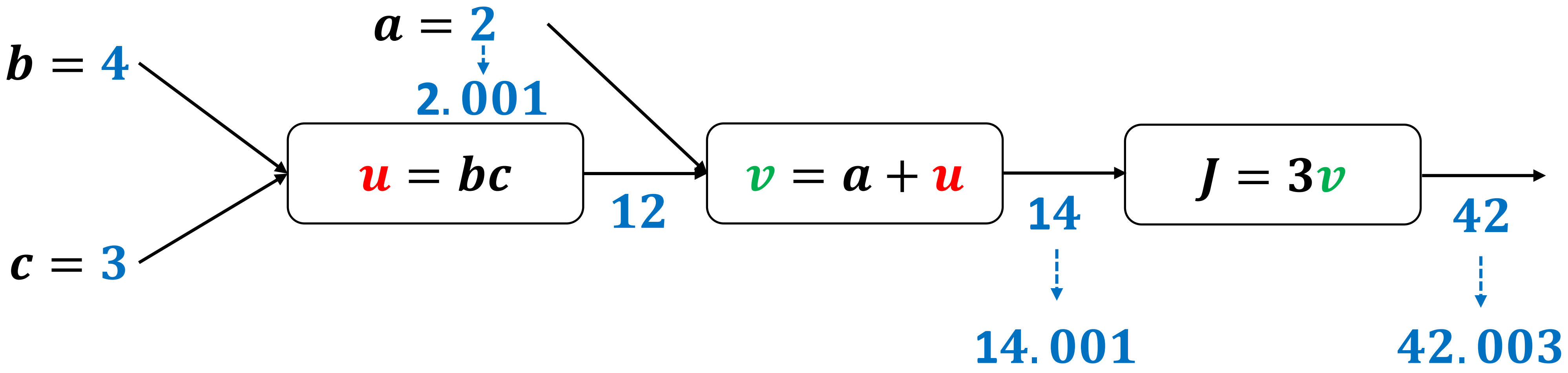


$$\frac{dJ}{da} = 3 = \qquad \frac{dv}{da}$$

The change in $\boldsymbol{a}$ caused a change in $\boldsymbol{v}$

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

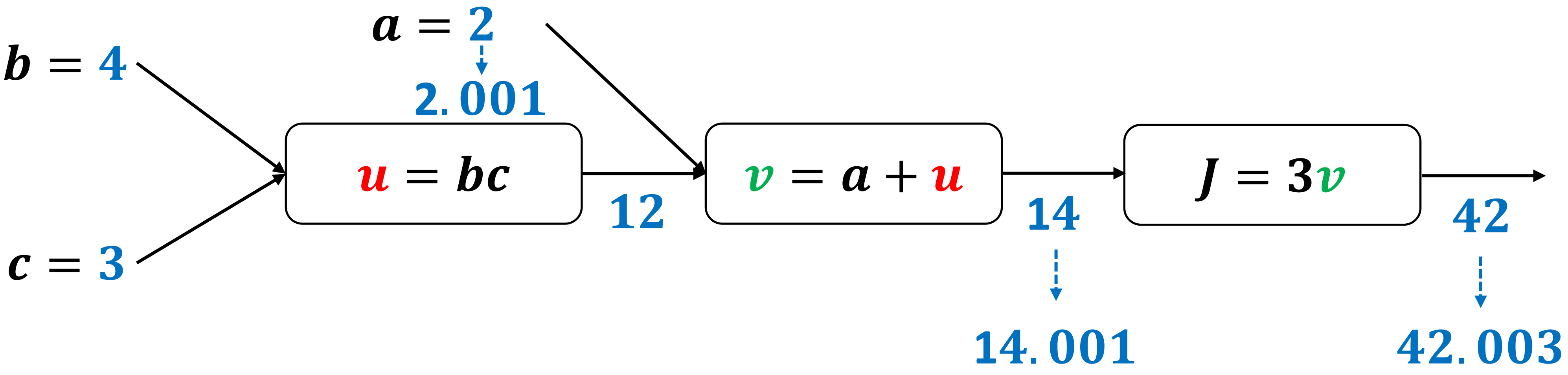


$$\frac{dJ}{da} = 3 = \frac{dJ}{dv} \times \frac{dv}{da}$$

And the change in v caused a change in J

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

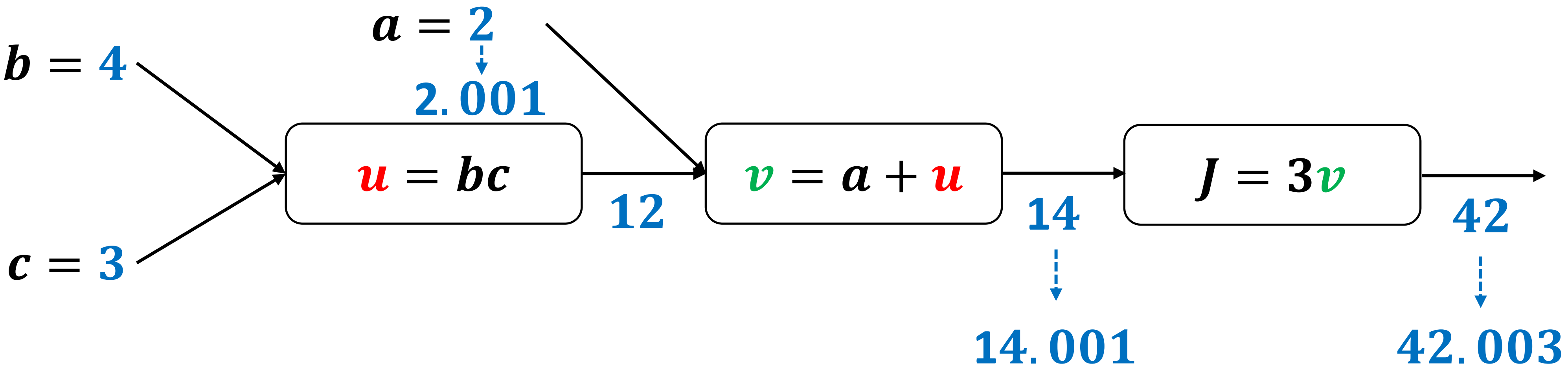


$$\frac{dJ}{da} = 3 = \frac{dJ}{dv} \times \frac{dv}{da}$$

This is denoted as ***the chain rule*** in calculus

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

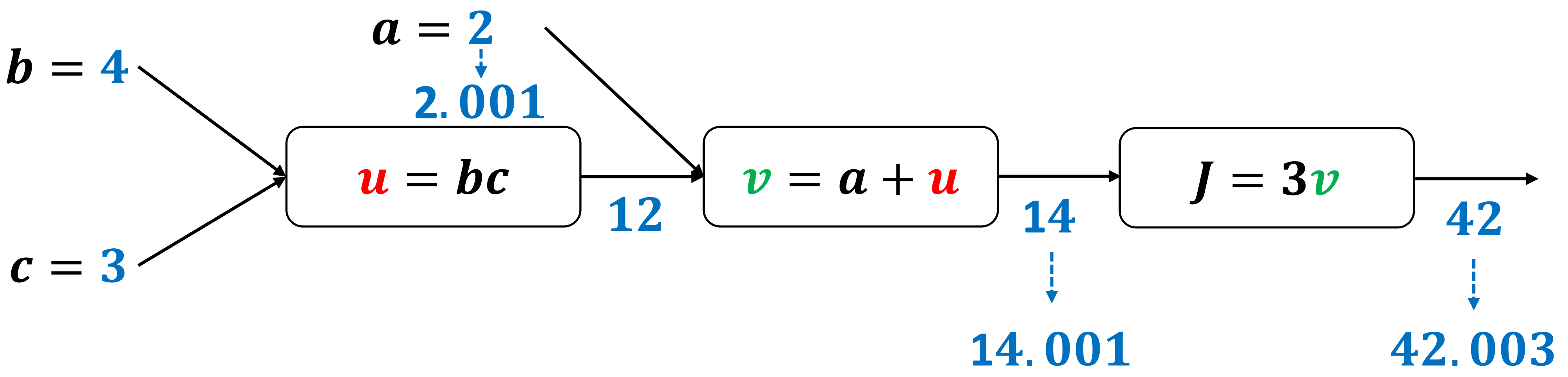


$\boldsymbol{a} = 2$
2.001
$\boldsymbol{b} = 4$
$\boldsymbol{c} = 3$
$\boldsymbol{u} = \boldsymbol{bc}$
12
$\boldsymbol{v} = \boldsymbol{a} + \boldsymbol{u}$
14
14.001
$\boldsymbol{J} = 3\boldsymbol{v}$
42
42.003

$$\frac{dJ}{da} = 3 = \frac{dJ}{dv} \times 1$$

Backward Propagation

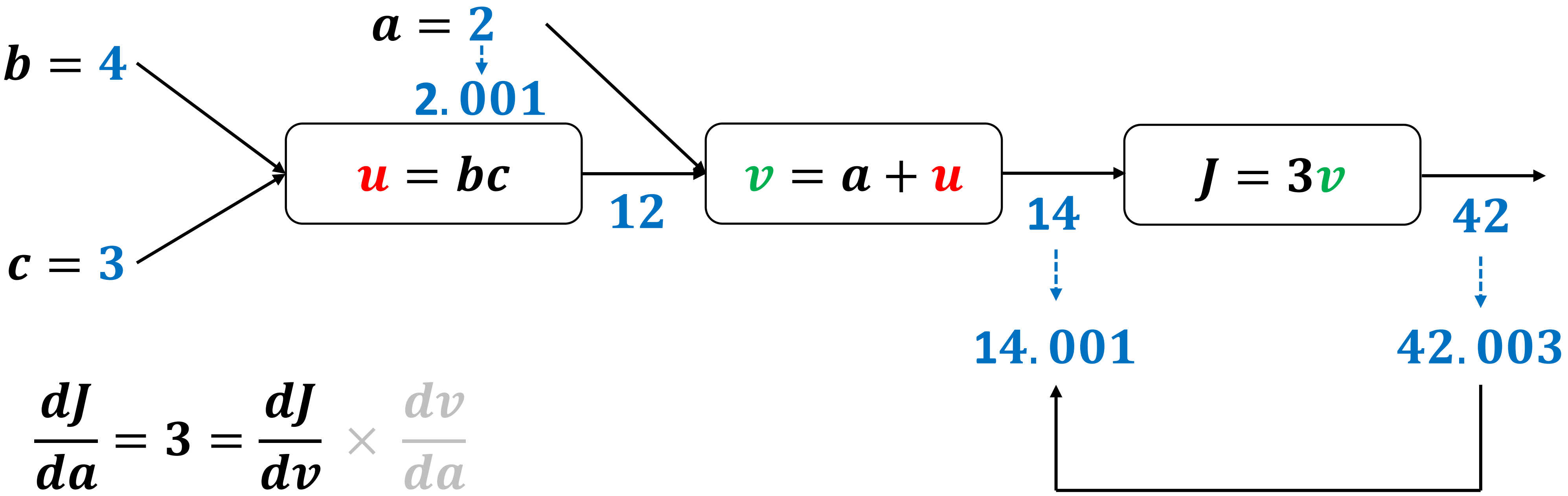
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{da} = 3 = 3 \times 1$$

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

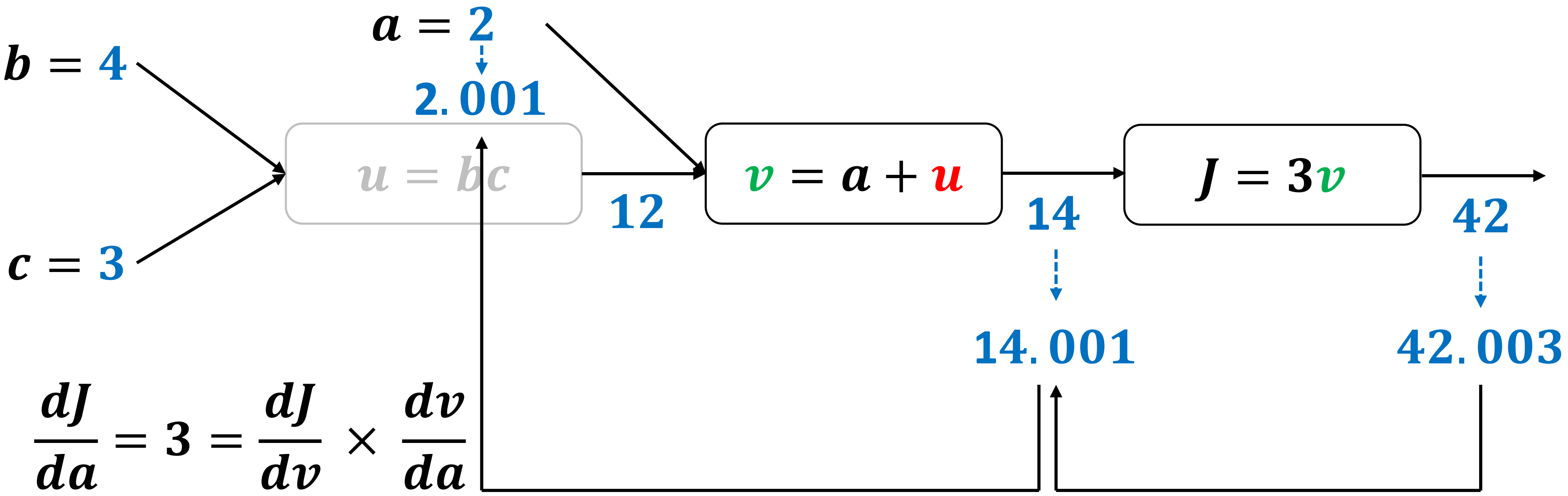


$$\frac{dJ}{da} = 3 = \frac{dJ}{dv} \times \frac{dv}{da}$$

In essence, to compute the derivative of J with respect to a, we had to go *back* to v, nudge it a little bit, and measure the corresponding resultant increase on J

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

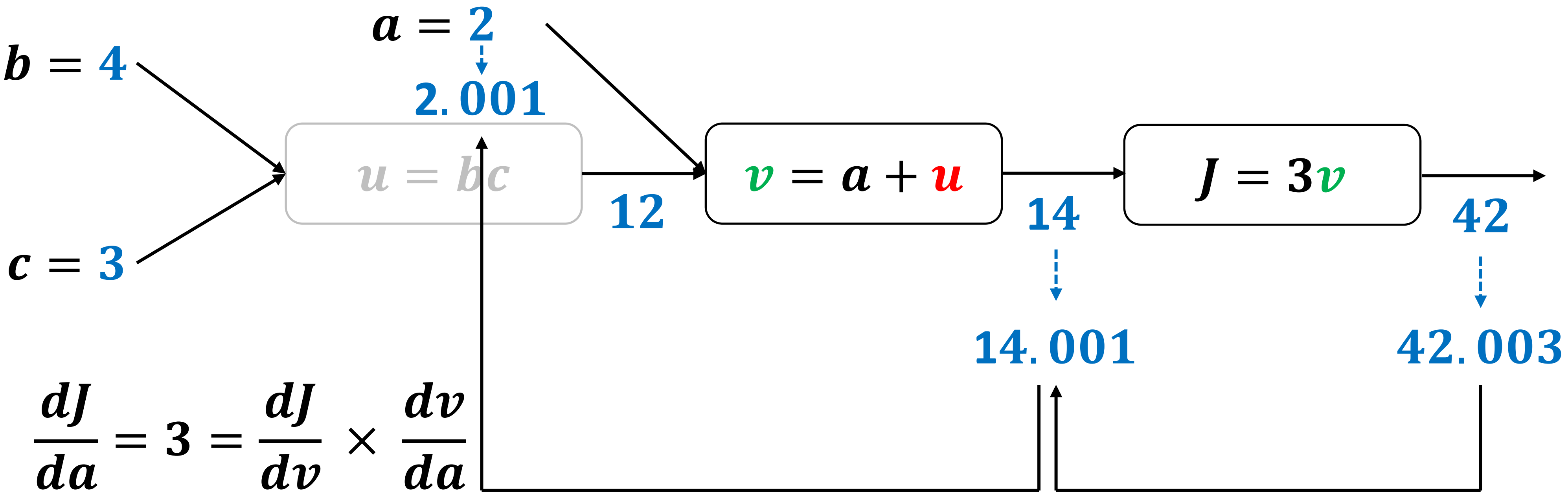


$$\frac{dJ}{da} = 3 = \frac{dJ}{dv} \times \frac{dv}{da}$$

Then, we had to go *back* to $\boldsymbol{a}$, nudge it a little bit, and measure the corresponding resultant increase on $\boldsymbol{v}$

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

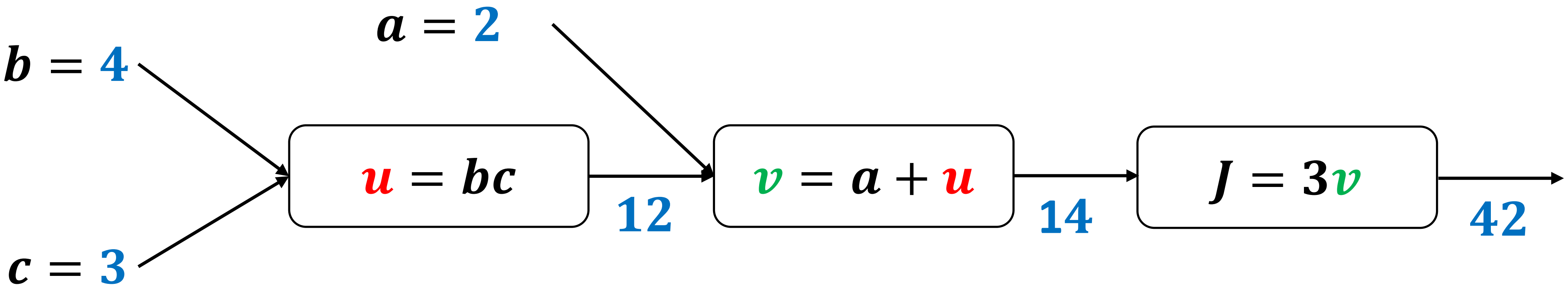


$$\frac{dJ}{da} = 3 = \frac{dJ}{dv} \times \frac{dv}{da}$$

Then, we multiplied the changes together (i.e., we applied the chain rule!)

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

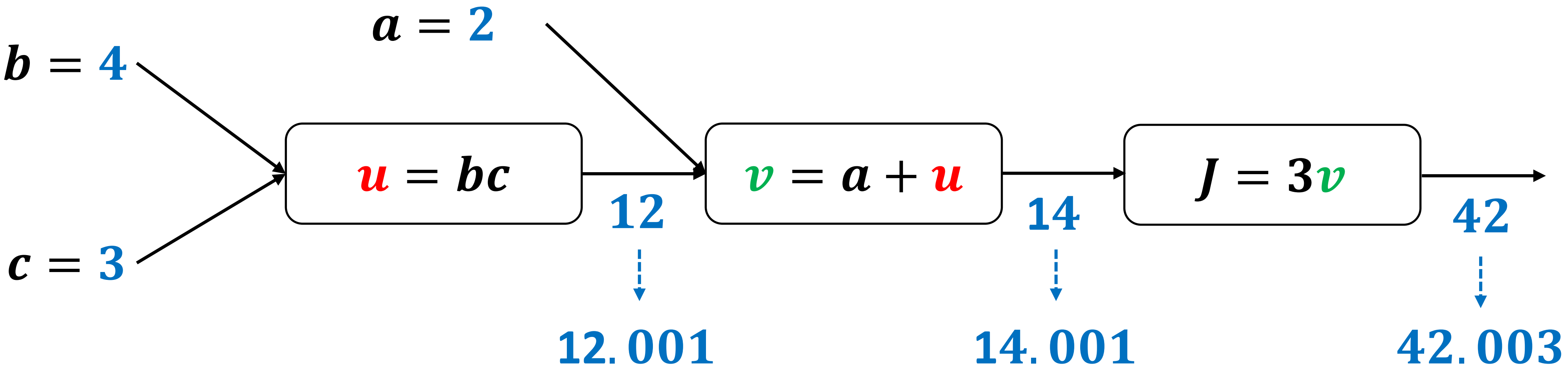


$\boldsymbol{a} = 2$
$\boldsymbol{b} = 4$
$\boldsymbol{c} = 3$
$\boldsymbol{u} = \boldsymbol{bc}$
12
$\boldsymbol{v} = \boldsymbol{a} + \boldsymbol{u}$
14
$\boldsymbol{J} = 3\boldsymbol{v}$
42

$\frac{\boldsymbol{dJ}}{\boldsymbol{du}}$ = Derivative of $\boldsymbol{J}$ with respect to $\boldsymbol{u}$

Backward Propagation

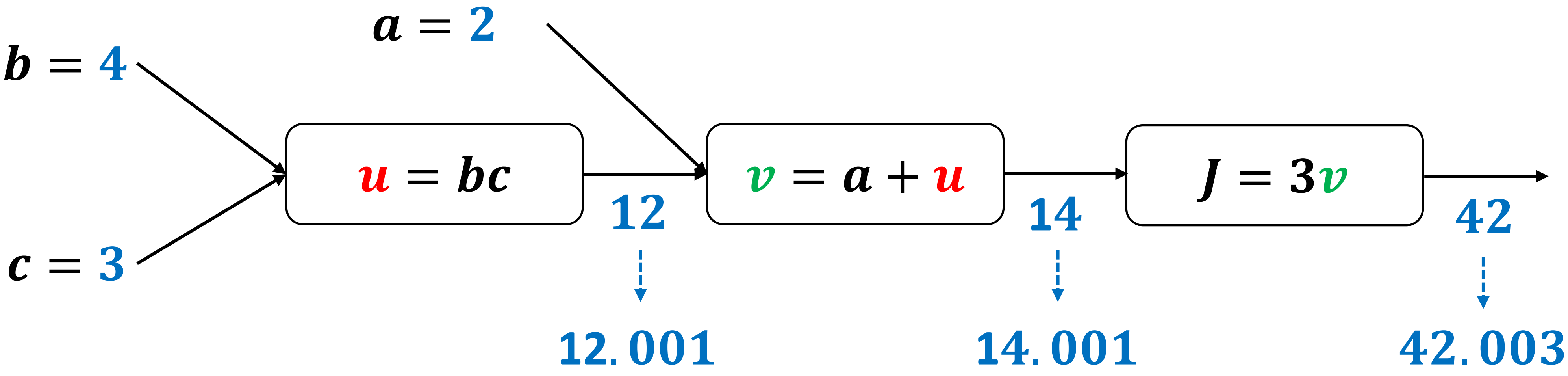
- Let us now compute the derivatives of the variables through the computation graph as follows:



$\frac{dJ}{du}$ = If we change u a little bit, how would J change?

Backward Propagation

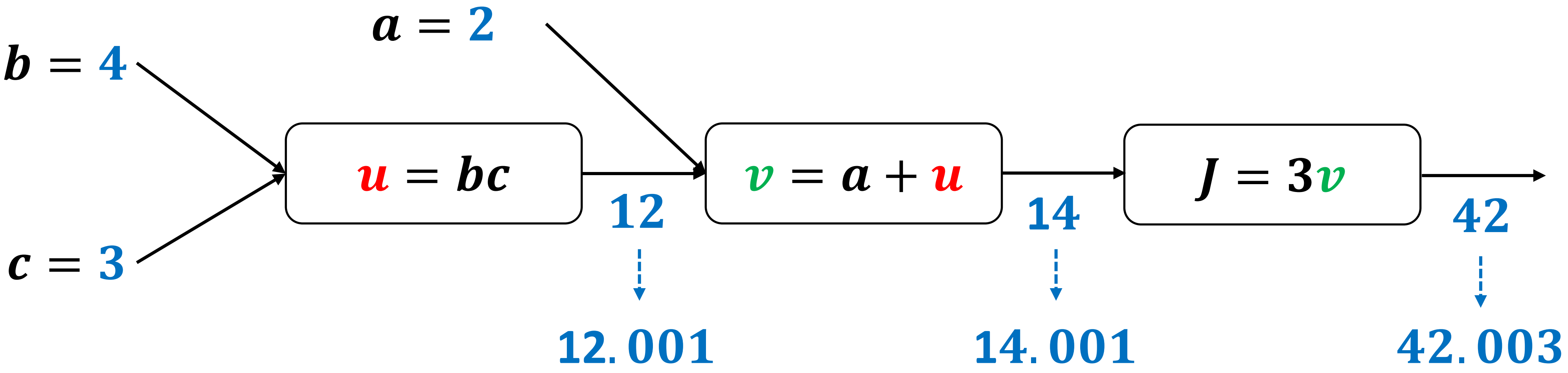
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{du} = 3$$

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

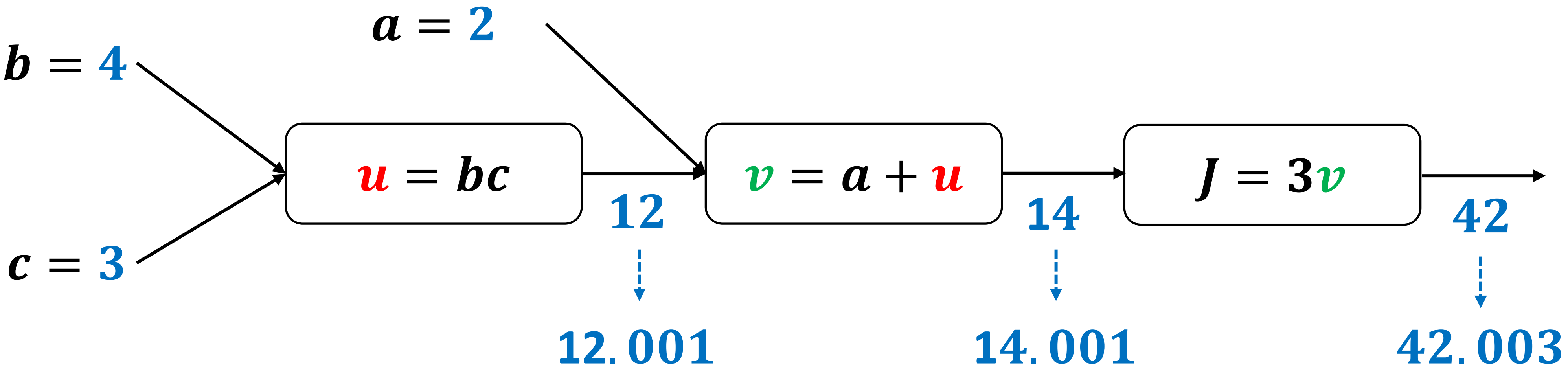


$$\frac{dJ}{du} = 3 = \qquad \frac{dv}{du}$$

The change in u caused a change in v

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

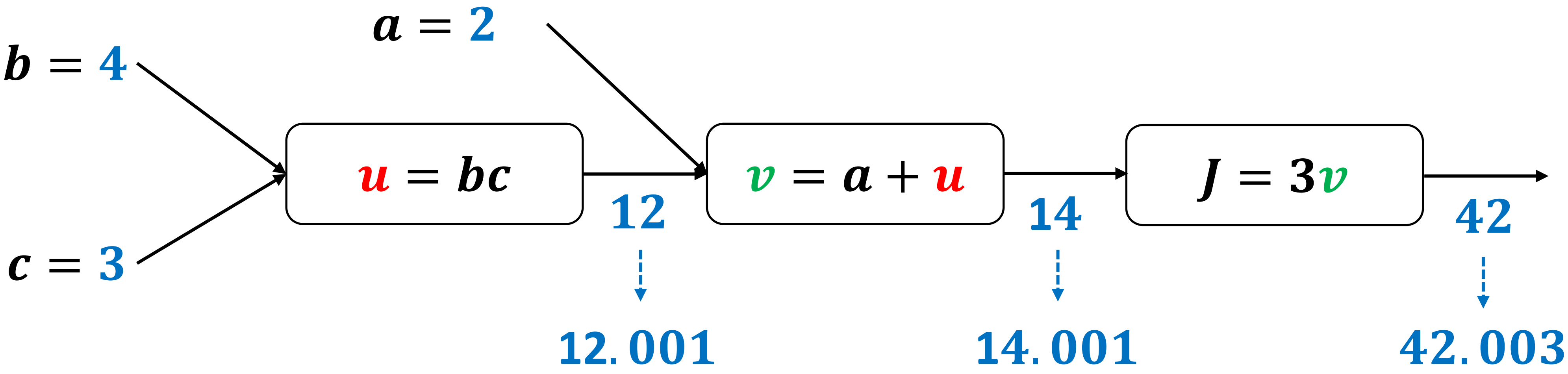


$$\frac{dJ}{du} = 3 = \frac{dJ}{dv} \times \frac{dv}{du}$$

And the change in v caused a change in J

Backward Propagation

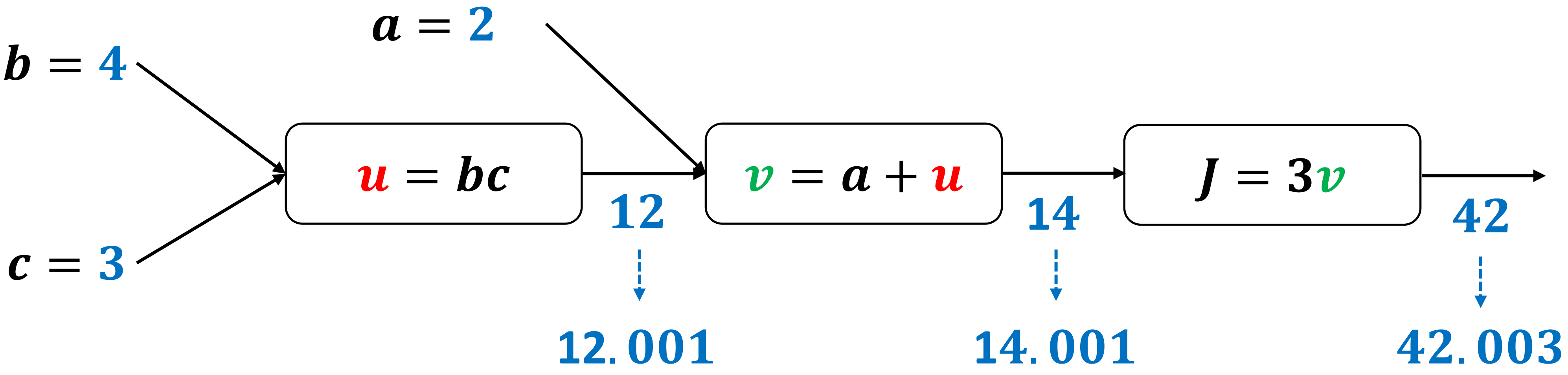
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{du} = 3 = \frac{dJ}{dv} \times 1$$

Backward Propagation

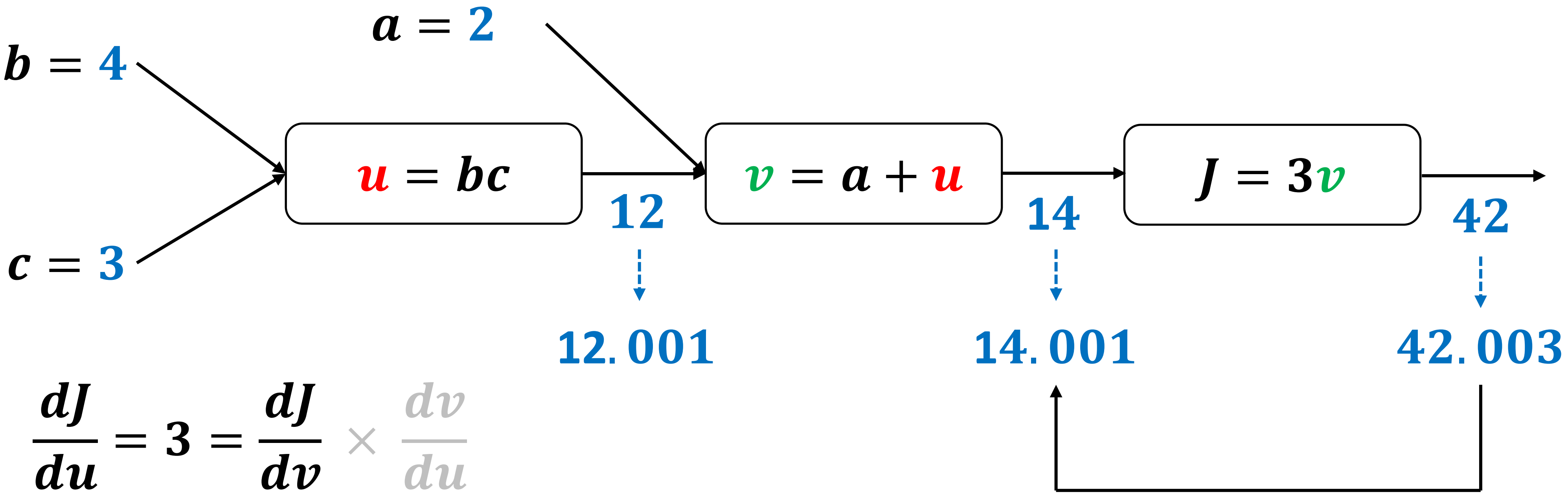
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{du} = 3 = 3 \times 1$$

Backward Propagation

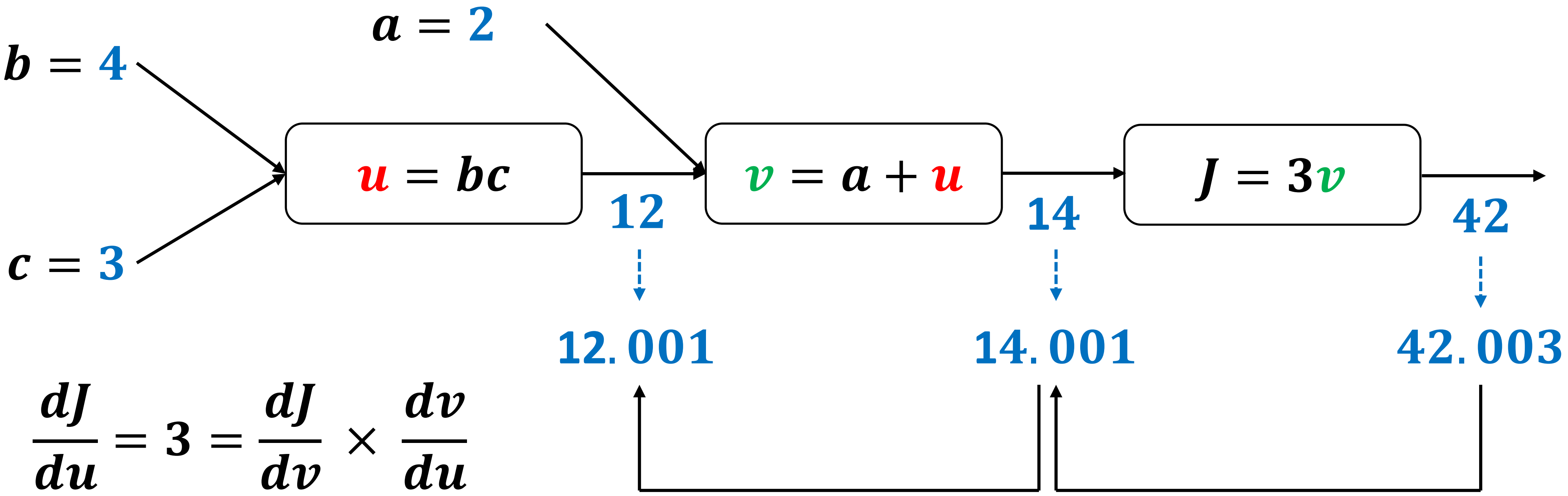
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{du} = 3 = \frac{dJ}{dv} \times \frac{dv}{du}$$

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

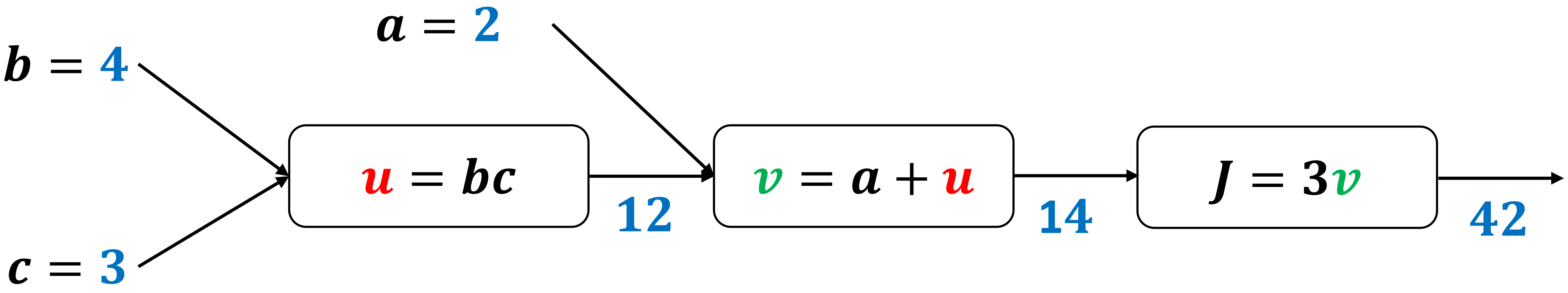


$$\frac{dJ}{du} = 3 = \frac{dJ}{dv} \times \frac{dv}{du}$$

Same as before, we had to go back to v then to u in order to compute the derivative of J

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

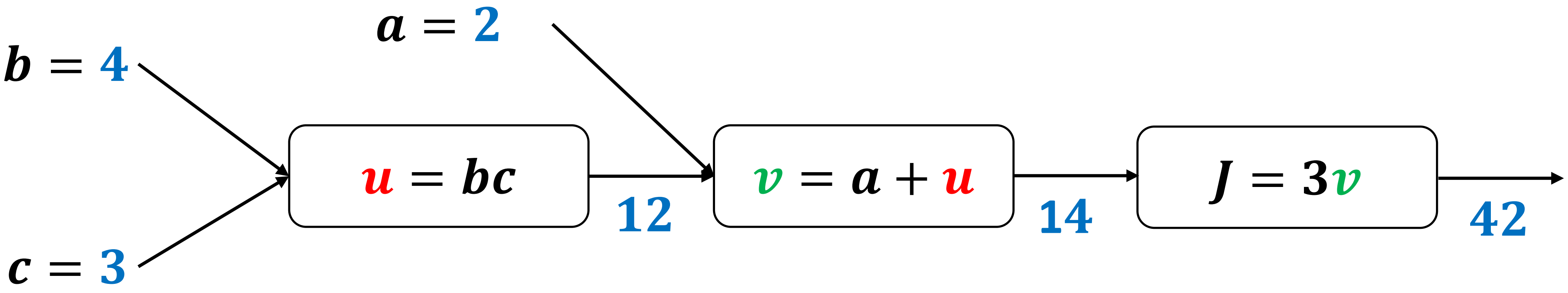


$\boldsymbol{a} = 2$
$\boldsymbol{b} = 4$
$\boldsymbol{c} = 3$
$\boldsymbol{u} = \boldsymbol{bc}$
12
$\boldsymbol{v} = \boldsymbol{a} + \boldsymbol{u}$
14
$\boldsymbol{J} = 3\boldsymbol{v}$
42

$\frac{\boldsymbol{dJ}}{\boldsymbol{db}}$ = Derivative of $\boldsymbol{J}$ with respect to $\boldsymbol{b}$

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

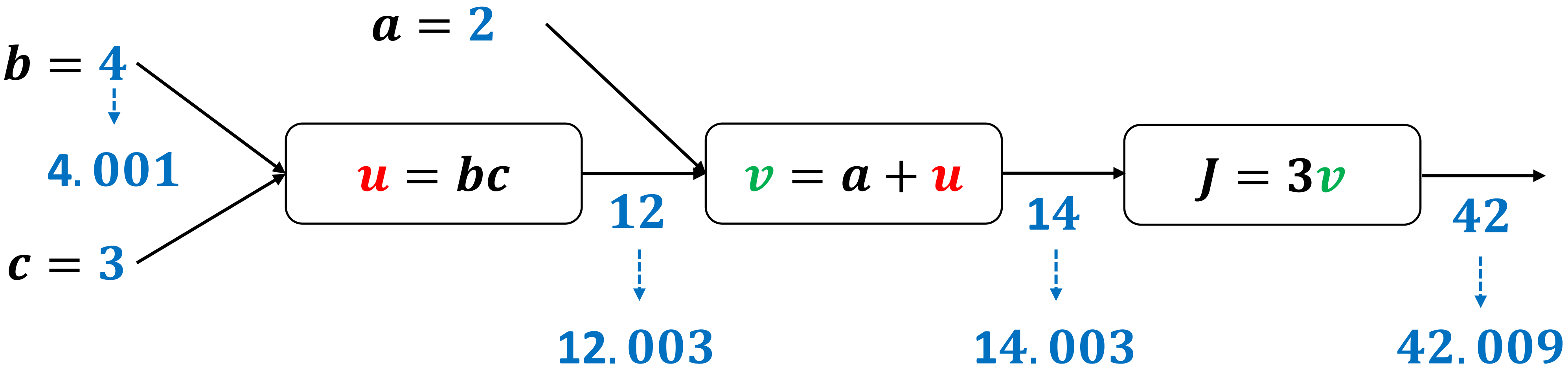


$\boldsymbol{a} = 2$
$\boldsymbol{b} = 4$
$\boldsymbol{c} = 3$
$\boldsymbol{u} = \boldsymbol{bc}$
12
$\boldsymbol{v} = \boldsymbol{a} + \boldsymbol{u}$
14
$\boldsymbol{J} = 3\boldsymbol{v}$
42

$$\frac{\boldsymbol{dJ}}{\boldsymbol{db}} = \text{If we change } \boldsymbol{b} \text{ a little bit, how would } \boldsymbol{J} \text{ change?}$$

Backward Propagation

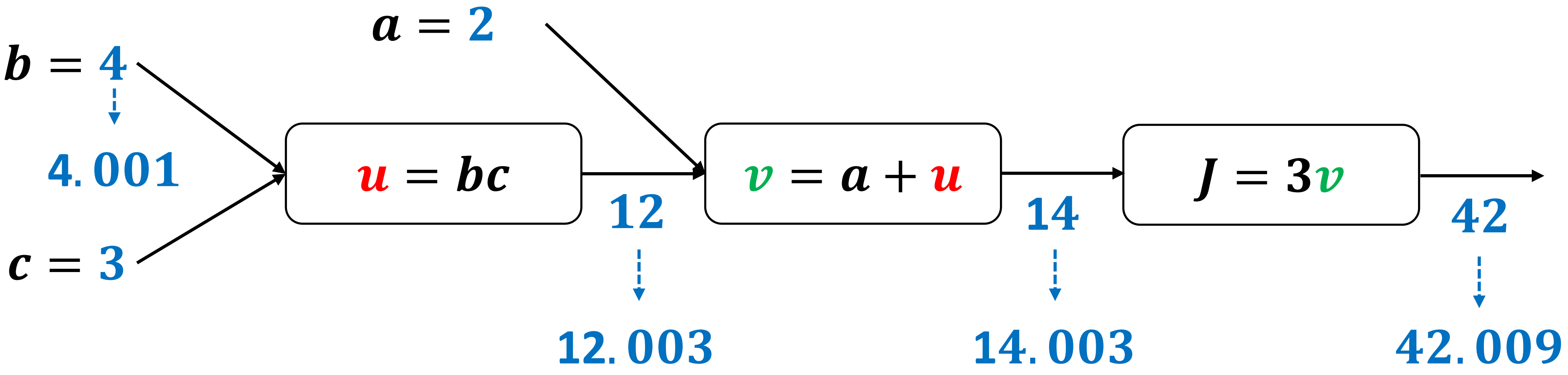
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{db} = ? = \underset{\substack{\downarrow \\ 3}}{\frac{dJ}{dv}} \times \underset{\substack{\downarrow \\ 1}}{\frac{dv}{du}} \times \underset{\substack{\downarrow \\ 3}}{\frac{du}{db}}$$

Backward Propagation

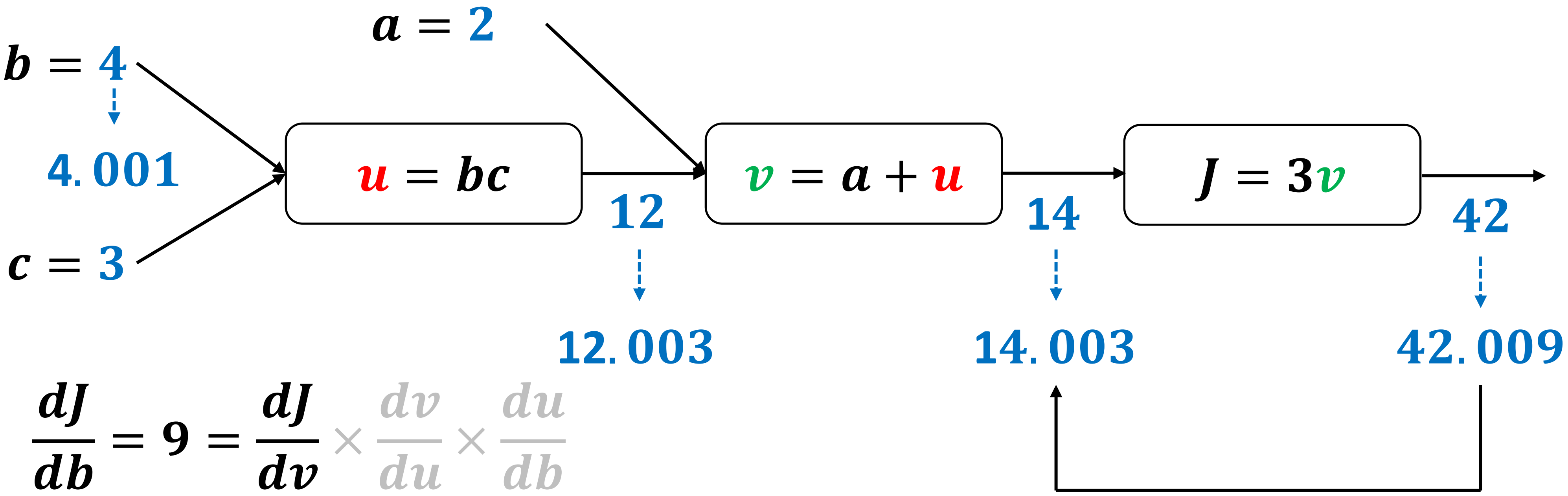
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{db} = 9 = \underset{3}{\frac{dJ}{dv}} \times \underset{1}{\frac{dv}{du}} \times \underset{3}{\frac{du}{db}}$$

Backward Propagation

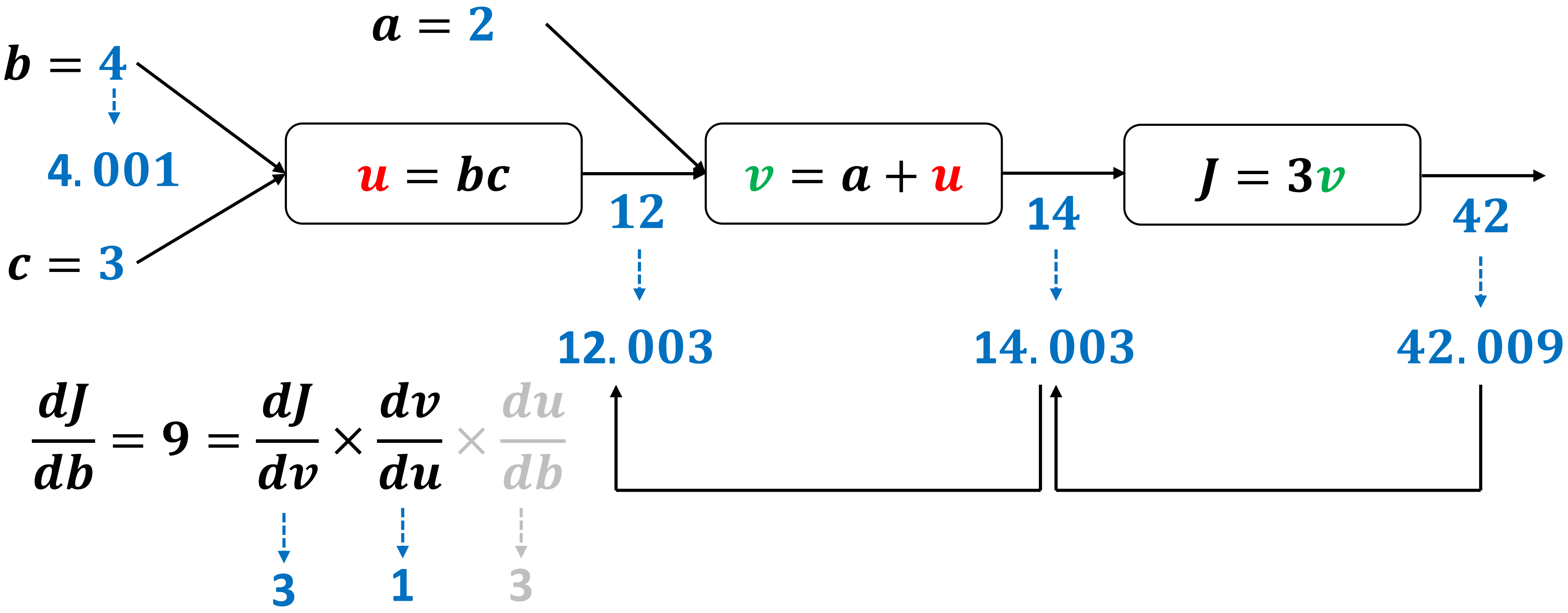
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{db} = 9 = \underset{3}{\frac{dJ}{dv}} \times \underset{1}{\frac{dv}{du}} \times \underset{3}{\frac{du}{db}}$$

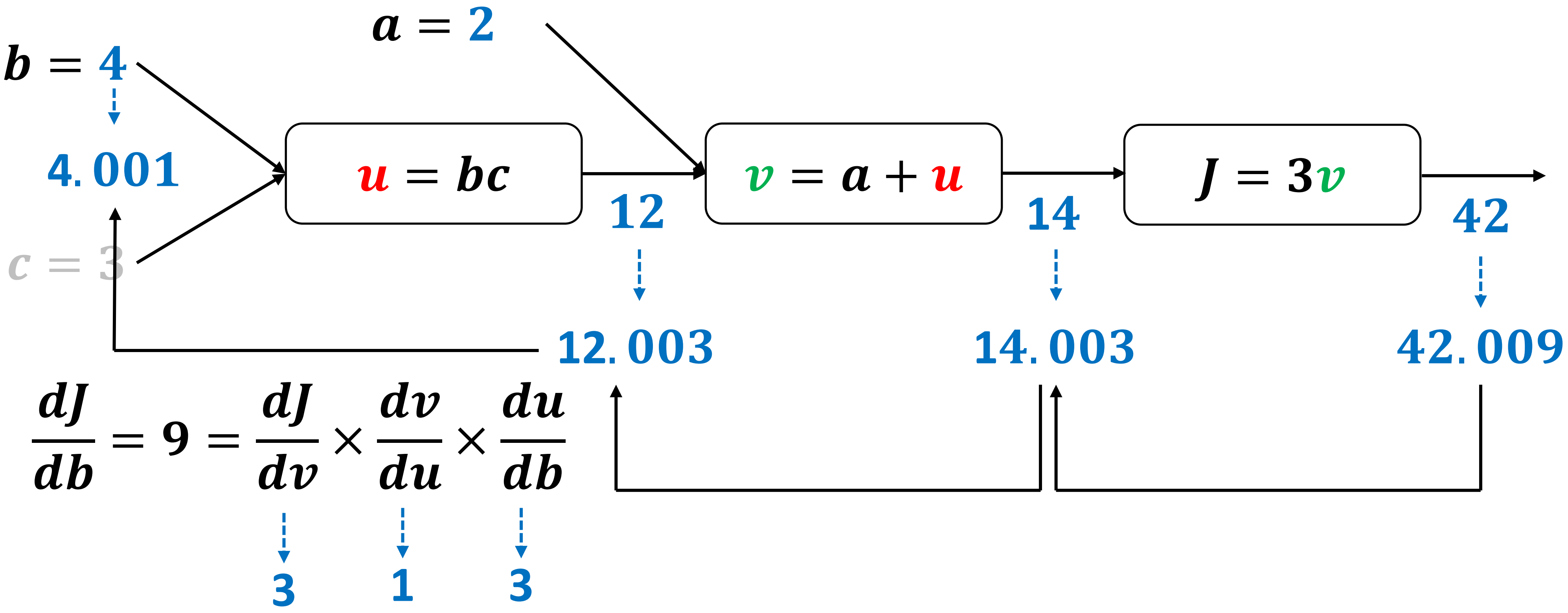
Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:



Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:



$a = 2$

$b = 4$

4.001

$c = 3$

$u = bc$

$v = a + u$

$J = 3v$

12

14

42

12.003

14.003

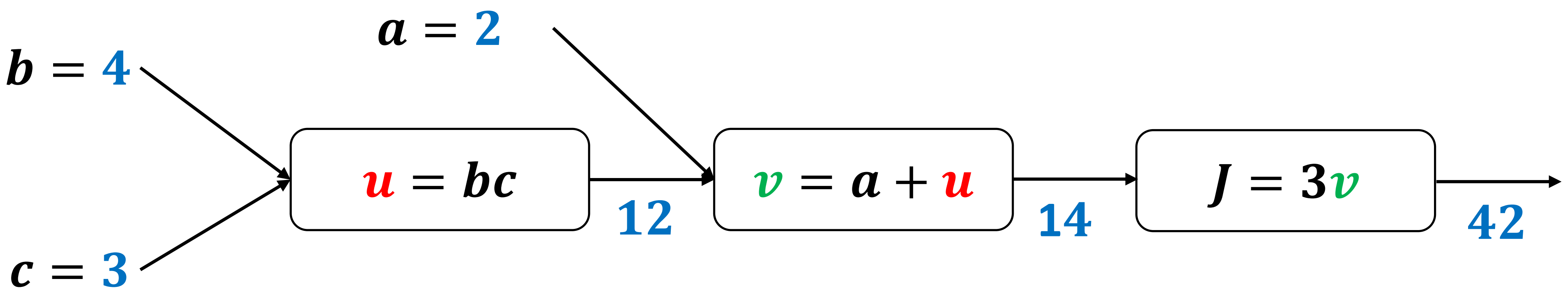
42.009

$$\frac{dJ}{db} = 9 = \frac{dJ}{dv} \times \frac{dv}{du} \times \frac{du}{db}$$

3 1 3

Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:

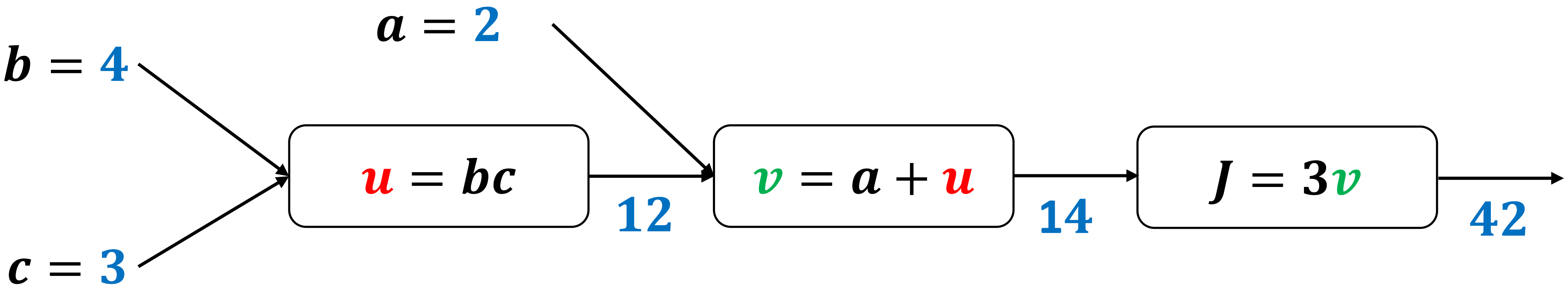


$\boldsymbol{a} = 2$
$\boldsymbol{b} = 4$
$\boldsymbol{c} = 3$
$\boldsymbol{u} = \boldsymbol{bc}$
12
$\boldsymbol{v} = \boldsymbol{a} + \boldsymbol{u}$
14
$\boldsymbol{J} = 3\boldsymbol{v}$
42

$\frac{\boldsymbol{dJ}}{\boldsymbol{dc}} =$ Derivative of $\boldsymbol{J}$ with respect to $\boldsymbol{c}$

Backward Propagation

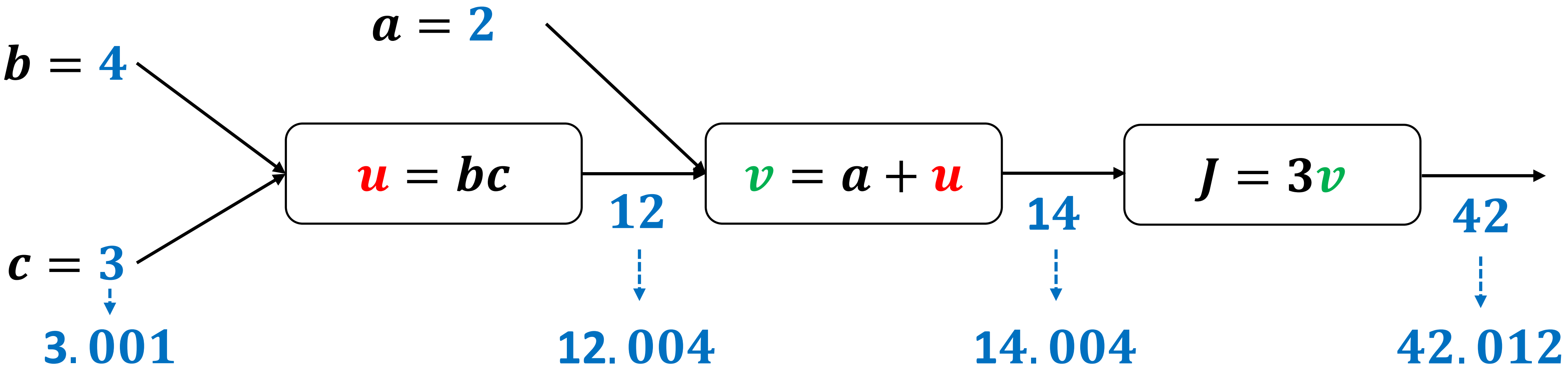
- Let us now compute the derivatives of the variables through the computation graph as follows:



$\frac{dJ}{dc} =$ If we change c a little bit, how would J change?

Backward Propagation

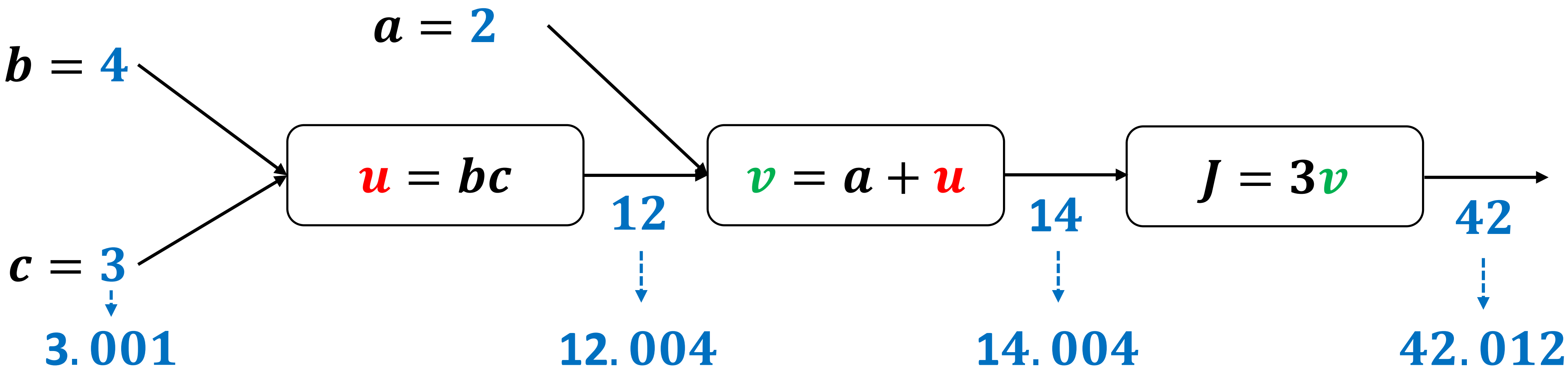
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{dc} = ? = \underset{\substack{\downarrow\\ 3}}{\frac{dJ}{dv}} \times \underset{\substack{\downarrow\\ 1}}{\frac{dv}{du}} \times \underset{\substack{\downarrow\\ 4}}{\frac{du}{dc}}$$

Backward Propagation

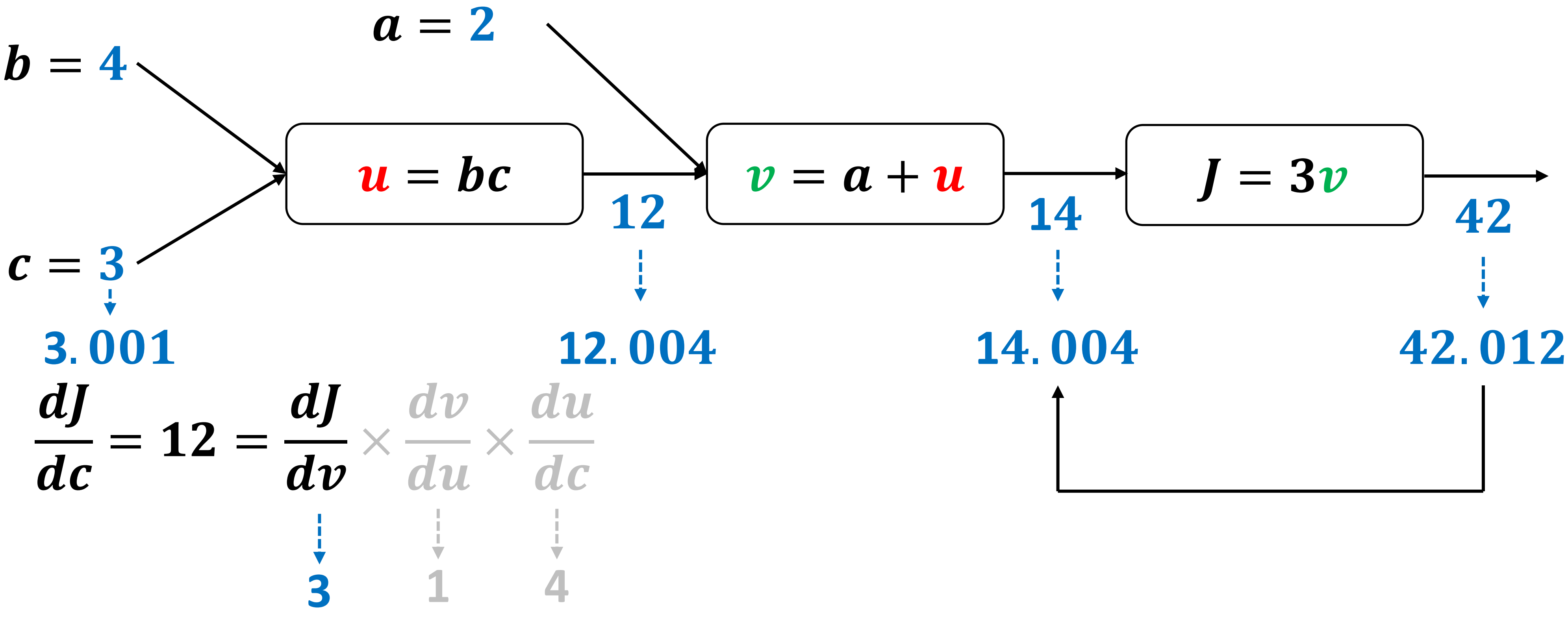
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{dc} = 12 = \underset{3}{\underset{\downarrow}{\frac{dJ}{dv}}} \times \underset{1}{\underset{\downarrow}{\frac{dv}{du}}} \times \underset{4}{\underset{\downarrow}{\frac{du}{dc}}}$$

Backward Propagation

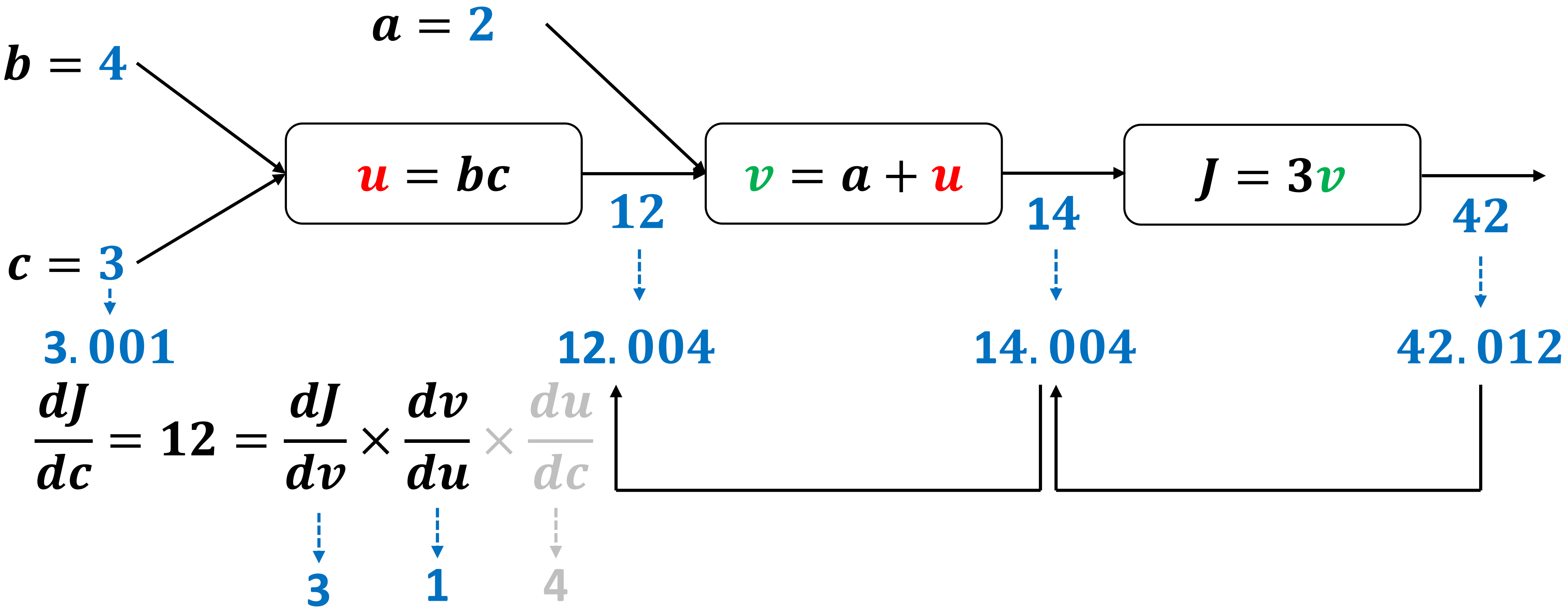
- Let us now compute the derivatives of the variables through the computation graph as follows:



$$\frac{dJ}{dc} = 12 = \underset{3}{\frac{dJ}{dv}} \times \underset{1}{\frac{dv}{du}} \times \underset{4}{\frac{du}{dc}}$$

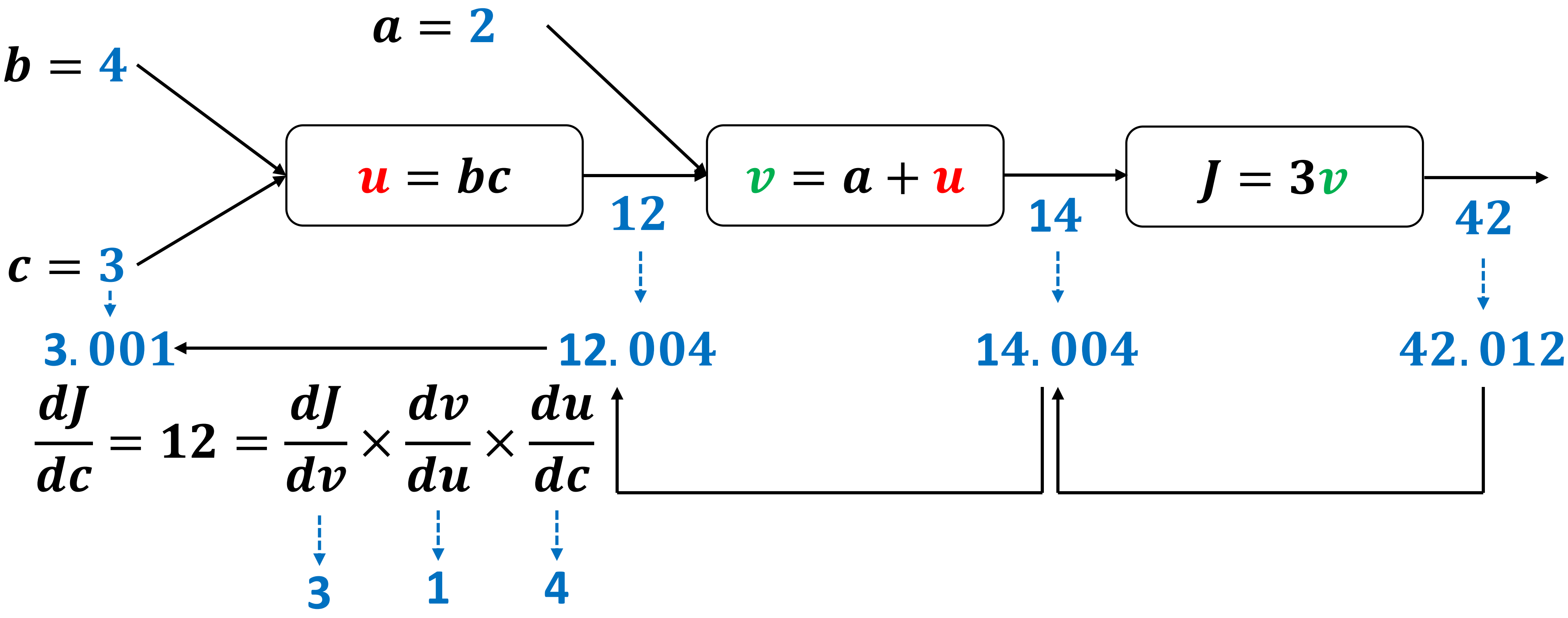
Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:



Backward Propagation

- Let us now compute the derivatives of the variables through the computation graph as follows:



$b = 4$

$a = 2$

$c = 3$

$u = bc$

$v = a + u$

$J = 3v$

12

14

42

3.001

12.004

14.004

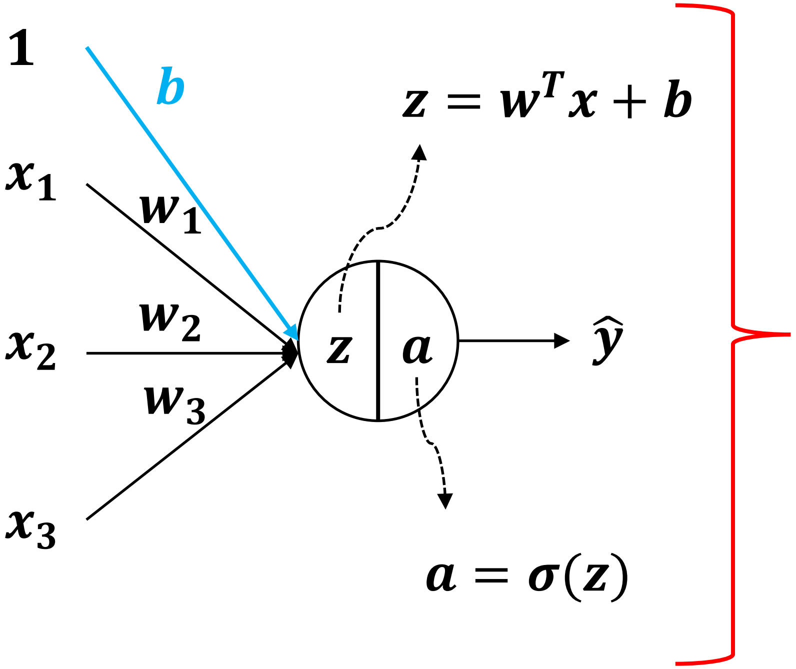
42.012

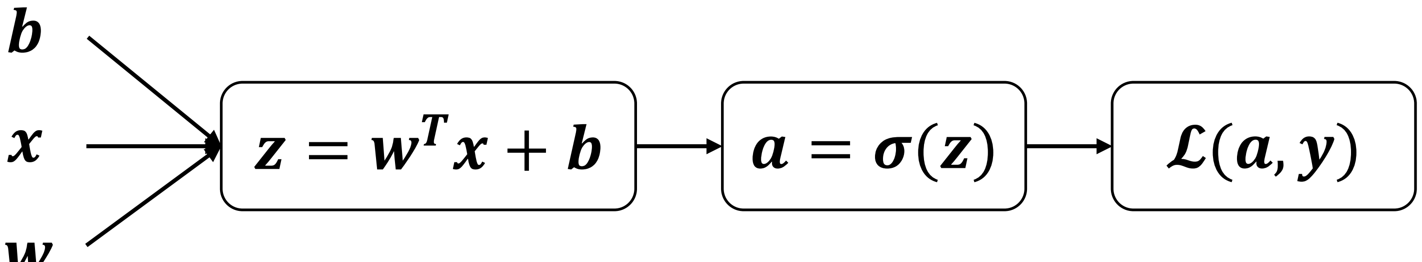
$$\frac{dJ}{dc} = 12 = \frac{dJ}{dv} \times \frac{dv}{du} \times \frac{du}{dc}$$

3 1 4

Computation Graph of Logistic Regression

- Let us translate logistic regression (which is a **neural network** with only 1 neuron) into a computation graph

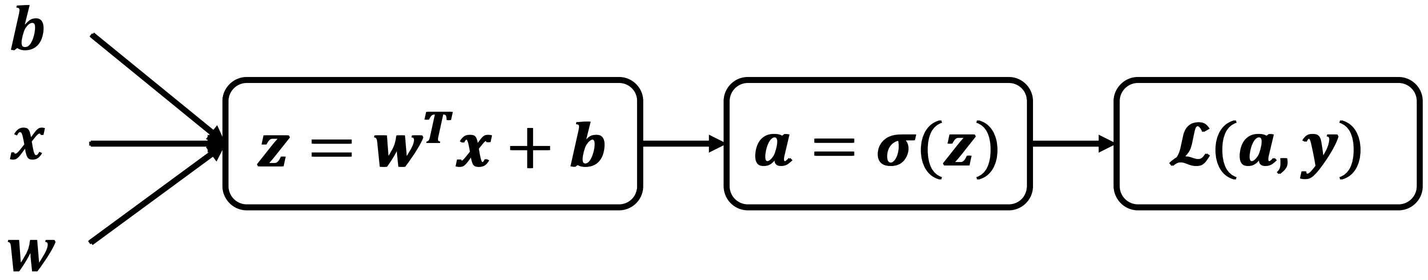




Where $b = 1$, $w = [w_1, w_2, w_3]$, $x = [x_1, x_2, x_3]$, and $\mathcal{L}(a, y)$ is the cost (or ***loss***) function (Cross-entropy loss)

Forward Propagation

- The loss function can be computed by moving from left to right



$\boldsymbol{b}$

$\boldsymbol{x}$

$\boldsymbol{w}$

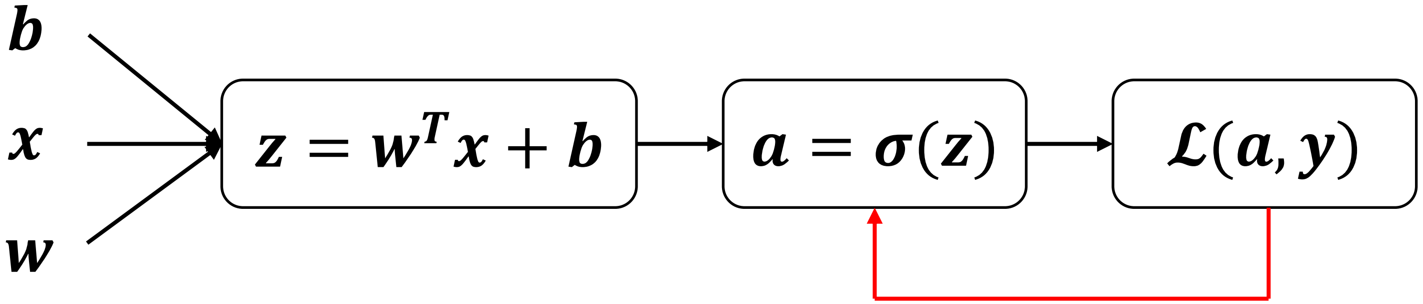
$\boldsymbol{z} = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + \boldsymbol{b}$

$\boldsymbol{a} = \boldsymbol{\sigma}(\boldsymbol{z})$

$\boldsymbol{\mathcal{L}}(\boldsymbol{a}, \boldsymbol{y})$

Backward Propagation

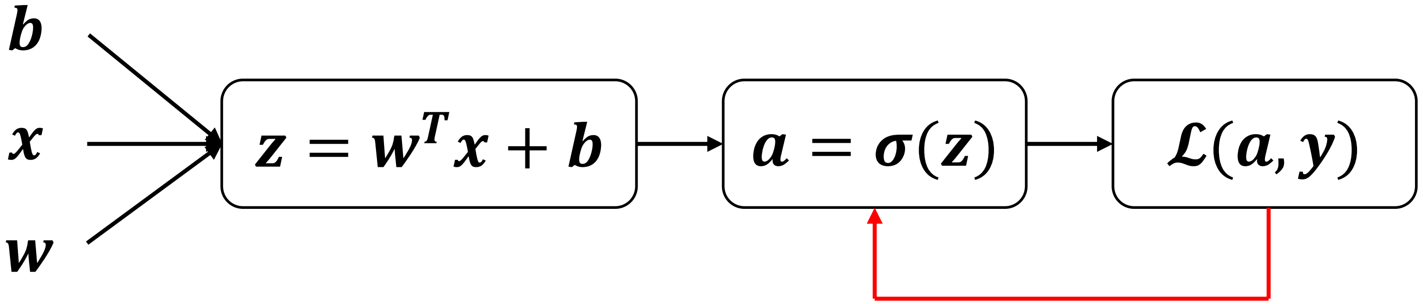
- The derivatives can be computed by moving from right to left



$$\frac{\partial \mathcal{L}}{\partial a} = \text{Partial derivative of } \mathcal{L} \text{ with respect to } a$$

Backward Propagation

- The derivatives can be computed by moving from right to left



$\boldsymbol{b}$

$\boldsymbol{x}$

$\boldsymbol{w}$

$\boldsymbol{z = w^T x + b}$

$\boldsymbol{a = \sigma(z)}$

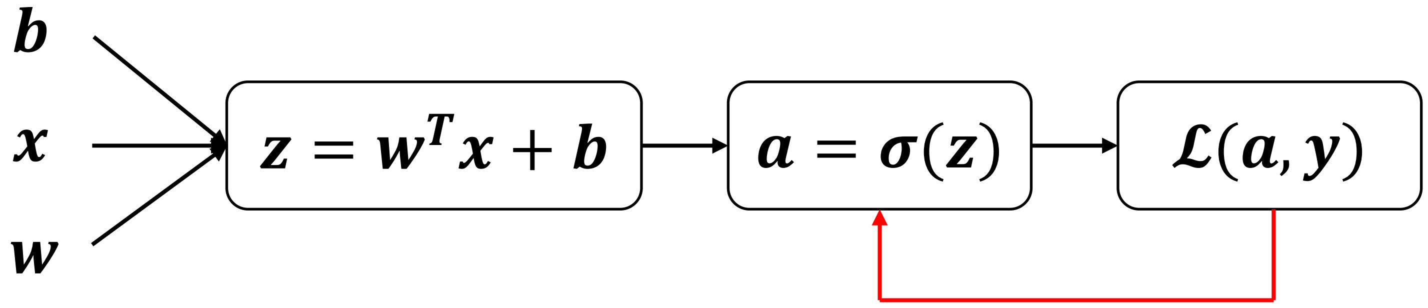
$\boldsymbol{\mathcal{L}(a, y)}$

$$\frac{\partial \mathcal{L}}{\partial a} = \frac{\partial}{\partial a}\left(-y log(a) - (1 - y) log(1 - a)\right)$$

Here we are using the cross-entropy loss

Backward Propagation

- The derivatives can be computed by moving from right to left



b

x

w

$z = w^T x + b$

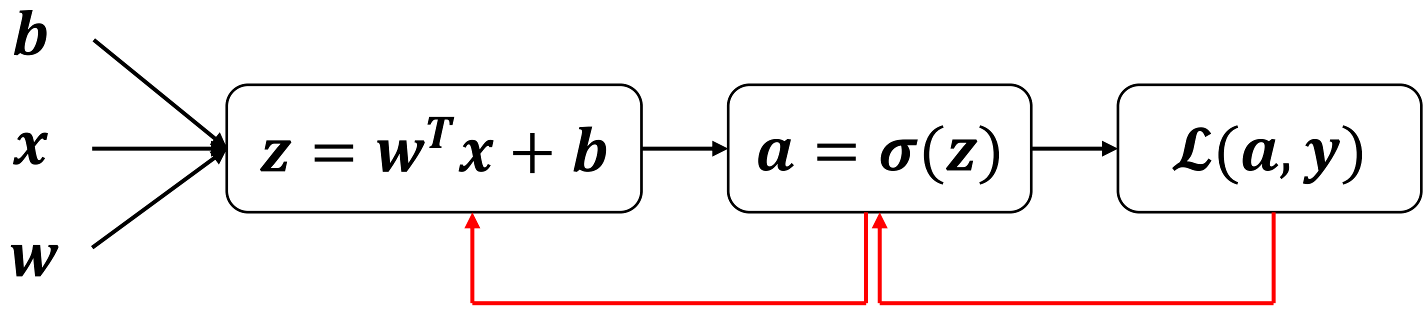
$a = \sigma(z)$

$\mathcal{L}(a, y)$

$$\frac{\partial \mathcal{L}}{\partial a} = \frac{-y}{a} + \frac{(1 - y)}{(1 - a)}$$

Backward Propagation

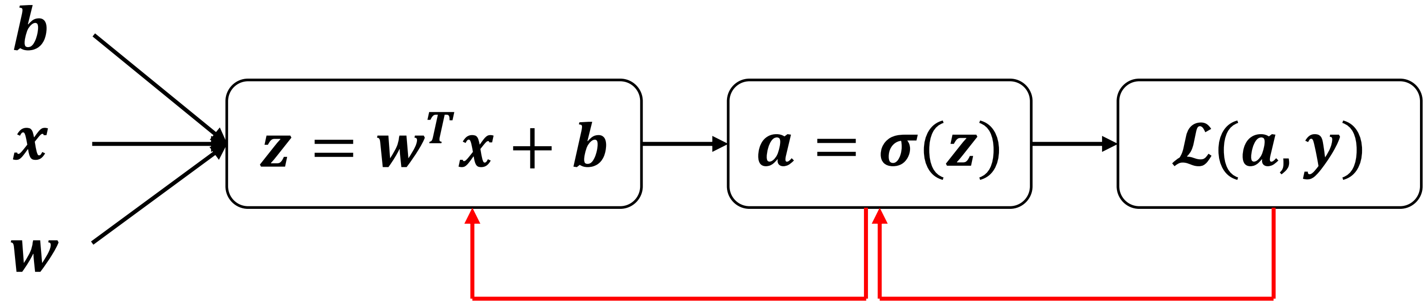
- The derivatives can be computed by moving from right to left



$\frac{\partial \mathcal{L}}{\partial z}$ = Partial derivative of $\mathcal{L}$ with respect to z

Backward Propagation

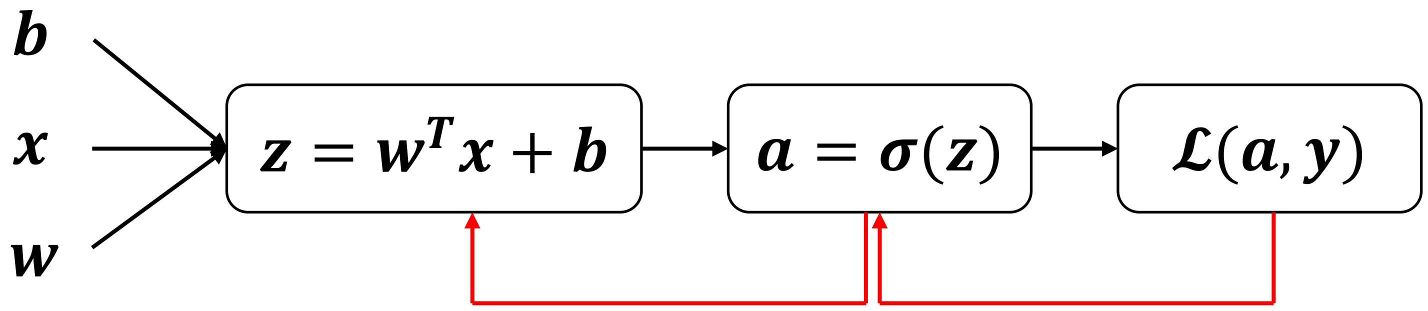
- The derivatives can be computed by moving from right to left



$$\frac{\partial \mathcal{L}}{\partial z} = \frac{\partial \mathcal{L}}{\partial a} \times \frac{\partial a}{\partial z} = \left(\frac{-y}{a} + \frac{(1-y)}{(1-a)}\right) \times \frac{\partial a}{\partial z} = \left(\frac{-y}{a} + \frac{(1-y)}{(1-a)}\right) \times \; a(1-a)$$

Backward Propagation

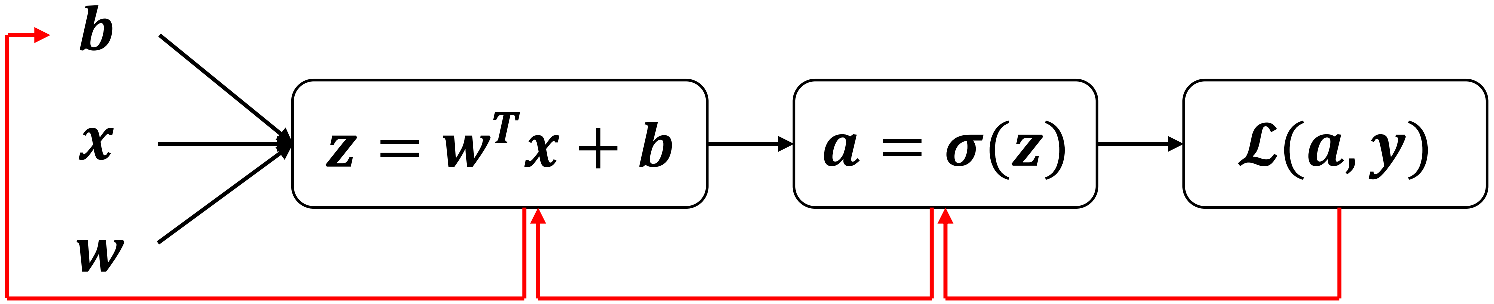
- The derivatives can be computed by moving from right to left



$$\frac{\partial \mathcal{L}}{\partial z} = a - y$$

Backward Propagation

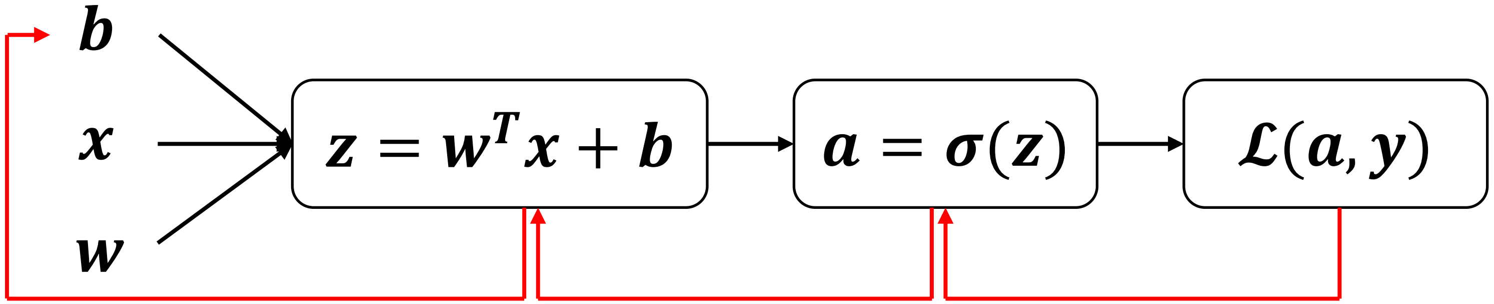
- The derivatives can be computed by moving from right to left


$\boldsymbol{b}$
$\boldsymbol{x}$
$\boldsymbol{w}$
$\boldsymbol{z = w^T x + b}$
$\boldsymbol{a = \sigma(z)}$
$\boldsymbol{\mathcal{L}(a, y)}$

$\frac{\partial \boldsymbol{\mathcal{L}}}{\partial \boldsymbol{b}}$ = Partial derivative of $\boldsymbol{\mathcal{L}}$ with respect to $\boldsymbol{b}$

Backward Propagation

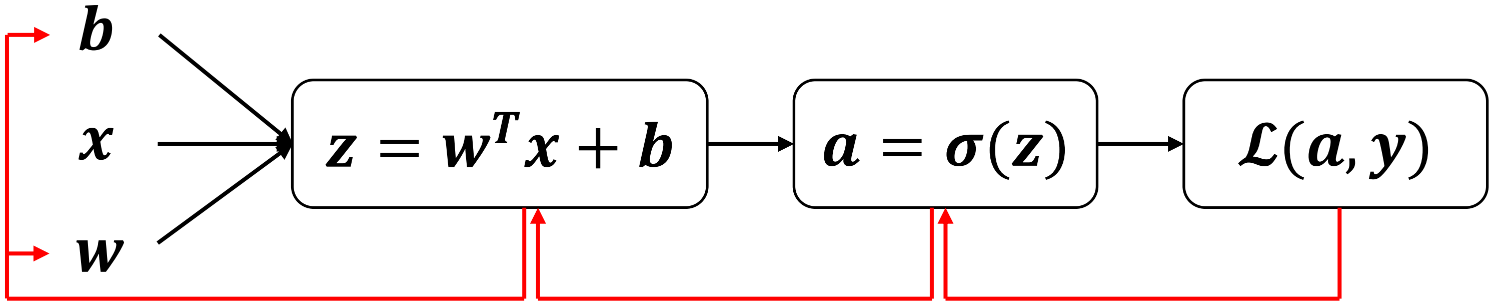
- The derivatives can be computed by moving from right to left



$$\frac{\partial \mathcal{L}}{\partial b} = \frac{\partial \mathcal{L}}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial b} = (a - y) \times \frac{\partial z}{\partial b} = (a - y) \times 1 = (a - y)$$

Backward Propagation

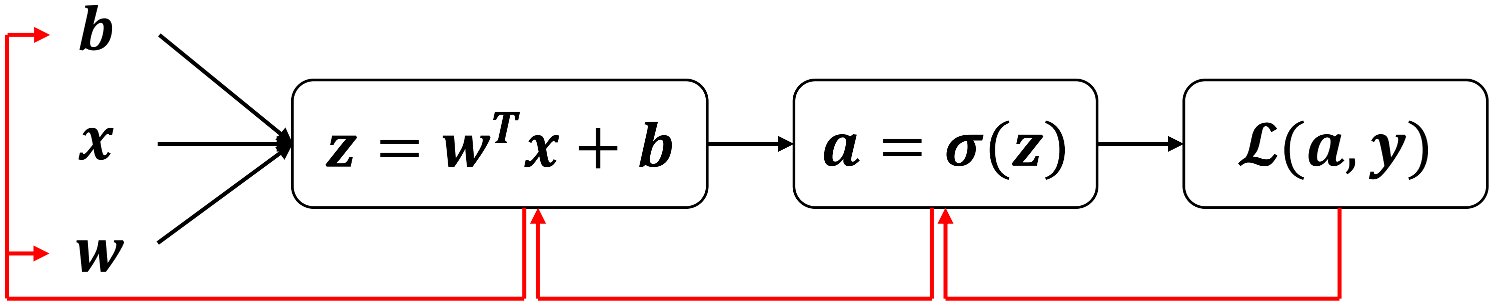
- The derivatives can be computed by moving from right to left



$\frac{\partial \mathcal{L}}{\partial w}$ = Partial derivative of $\mathcal{L}$ with respect to w

Backward Propagation

- The derivatives can be computed by moving from right to left



$$\frac{\partial \mathcal{L}}{\partial w} = \frac{\partial \mathcal{L}}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w} = (a - y) \times \frac{\partial z}{\partial w} = (a - y)x$$

Backward Propagation: Summary

- Here is the summary of the gradients in logistic regression:

$$\boldsymbol{dz} = \underbrace{\frac{\boldsymbol{\partial \mathcal{L}}}{\boldsymbol{\partial z}}} = \boldsymbol{a} - \boldsymbol{y}$$

We will denote this as $\boldsymbol{dz}$ for simplicity

Backward Propagation: Summary

- Here is the summary of the gradients in logistic regression:

$$\boldsymbol{dz} = \frac{\partial \mathcal{L}}{\partial z} = a - y$$

$$\boldsymbol{db} = \frac{\partial \mathcal{L}}{\partial b} = a - y$$

$$\boldsymbol{dw} = \frac{\partial \mathcal{L}}{\partial w} = (a - y)x$$

Gradient Descent For Logistic Regression

- **Outline**:
  - Have a loss function $\boldsymbol{\mathcal{L}(w, b)}$, where $\boldsymbol{w}$ = $[\boldsymbol{w_1}, \dots, \boldsymbol{w_m}]$ and $\boldsymbol{b} = \boldsymbol{w_0}$
  - Start with some guesses for $\boldsymbol{w_1}, \dots, \boldsymbol{w_m}$
    - It does not really matter what values you start with for $\boldsymbol{w_1}, \dots, \boldsymbol{w_m}$, but a common choice is to set them all initially to zero or random values
  - Repeat until convergence{

$$\left.\begin{aligned} w_j &= w_j - \alpha \frac{\partial \boldsymbol{\mathcal{L}(w, b)}}{\partial\, w_j} \\ b &= b - \alpha \frac{\partial \boldsymbol{\mathcal{L}(w, b)}}{\partial b} \end{aligned}\right]$$

Let us focus on this part

}

Gradient Descent For Logistic Regression

- **Outline**:
  - Repeat until convergence{

Assuming n examples

```
for i = 1 to n:
    $z^{(i)} = w^T x^{(i)} + b$
    $a^{(i)} = \sigma(z^{(i)})$
    $dz^{(i)} = a^{(i)} - y^{(i)}$
    $dw = dw + dz^{(i)} x^{(i)}$
    $db = db + dz^{(i)}$
$dw = dw/n$
$db = db/n$
$w = w - \alpha dw$
$b = b - \alpha db$
}
```

Forward propagation

Backward propagation

Outside the for loop

$$Z = w^T X + b$$

$$A = \sigma(Z)$$

$$dZ = A - Y$$

$$dw = \frac{1}{n} X dZ^T$$

$$db = \frac{1}{n} \sum_{i=1}^{n} dz^{(i)}$$

$$w = w - \alpha dw$$

$$b = b - \alpha db$$

Vectorized version

Gradient Descent For Logistic Regression

- **Outline**:
  - Repeat until convergence{

$$\boldsymbol{Z = w^T X + b}$$

$$\boldsymbol{A = \sigma(Z)}$$

$$\boldsymbol{dZ = A - Y}$$

$$\boldsymbol{dw = \frac{1}{n} X dZ^T}$$

$$\boldsymbol{db = \frac{1}{n} \sum_{i=1}^{n} dz^{(i)}}$$

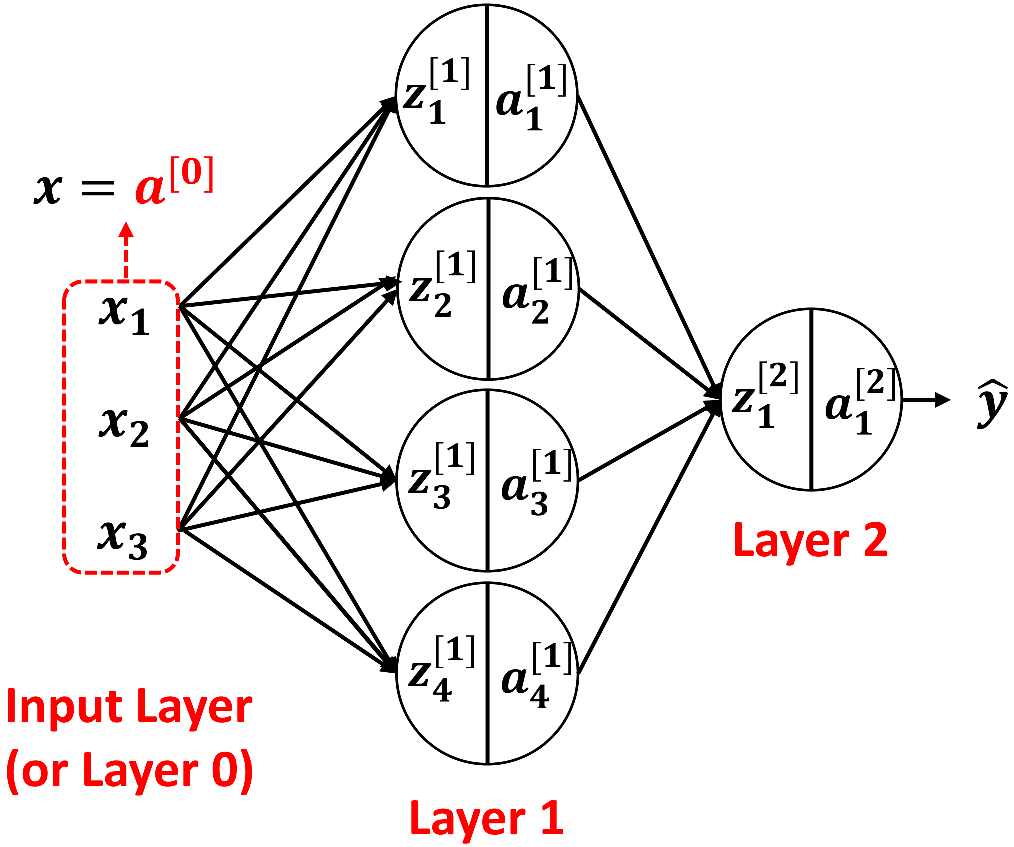
$$\boldsymbol{w = w - \alpha dw}$$
$$\boldsymbol{b = b - \alpha db}$$

}

Computation Graph of A Neural Network

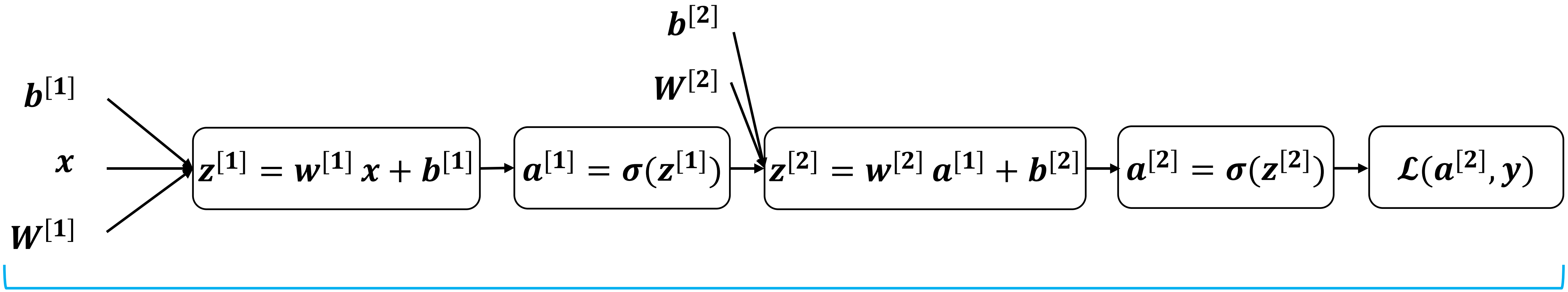
- Similar to logistic regression, we can represent any neural network in terms of a computation graph

A neural network with 2 layers



Computation Graph of A Neural Network

- Similar to logistic regression, we can represent any neural network in terms of a computation graph



$\boldsymbol{b}^{[1]}$

$\boldsymbol{x}$

$\boldsymbol{W}^{[1]}$

$\boldsymbol{z}^{[1]} = \boldsymbol{w}^{[1]}\,\boldsymbol{x} + \boldsymbol{b}^{[1]}$

$\boldsymbol{a}^{[1]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[1]})$

$\boldsymbol{b}^{[2]}$

$\boldsymbol{W}^{[2]}$

$\boldsymbol{z}^{[2]} = \boldsymbol{w}^{[2]}\,\boldsymbol{a}^{[1]} + \boldsymbol{b}^{[2]}$

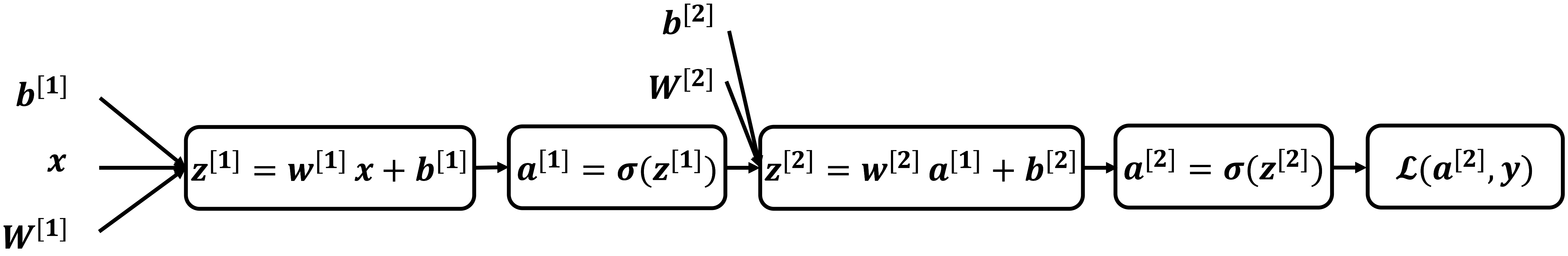
$\boldsymbol{a}^{[2]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[2]})$

$\mathcal{L}(\boldsymbol{a}^{[2]}, \boldsymbol{y})$

The corresponding computation graph

Forward Propagation

- The loss function can be computed by moving from left to right



$\boldsymbol{b}^{[1]}$

$\boldsymbol{x}$

$\boldsymbol{W}^{[1]}$

$\boldsymbol{z}^{[1]} = \boldsymbol{w}^{[1]}\,\boldsymbol{x} + \boldsymbol{b}^{[1]}$

$\boldsymbol{a}^{[1]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[1]})$

$\boldsymbol{b}^{[2]}$

$\boldsymbol{W}^{[2]}$

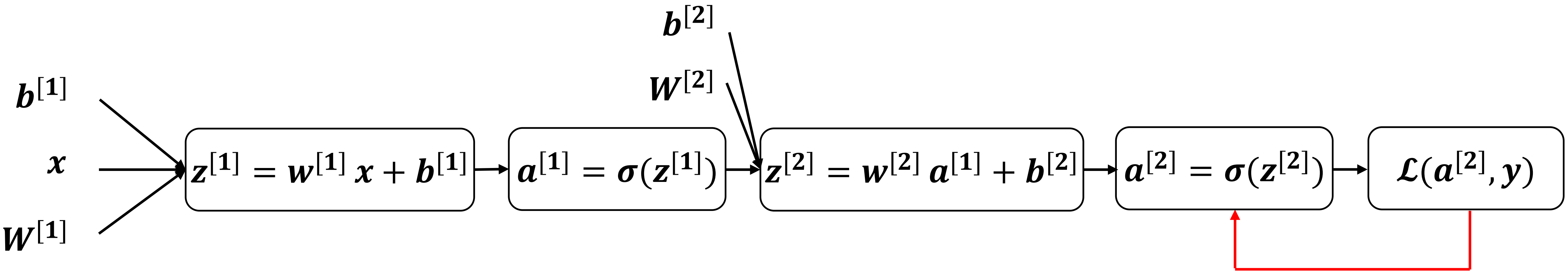
$\boldsymbol{z}^{[2]} = \boldsymbol{w}^{[2]}\,\boldsymbol{a}^{[1]} + \boldsymbol{b}^{[2]}$

$\boldsymbol{a}^{[2]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[2]})$

$\boldsymbol{\mathcal{L}}(\boldsymbol{a}^{[2]}, \boldsymbol{y})$

Backward Propagation

- The derivatives can be computed by moving from right to left



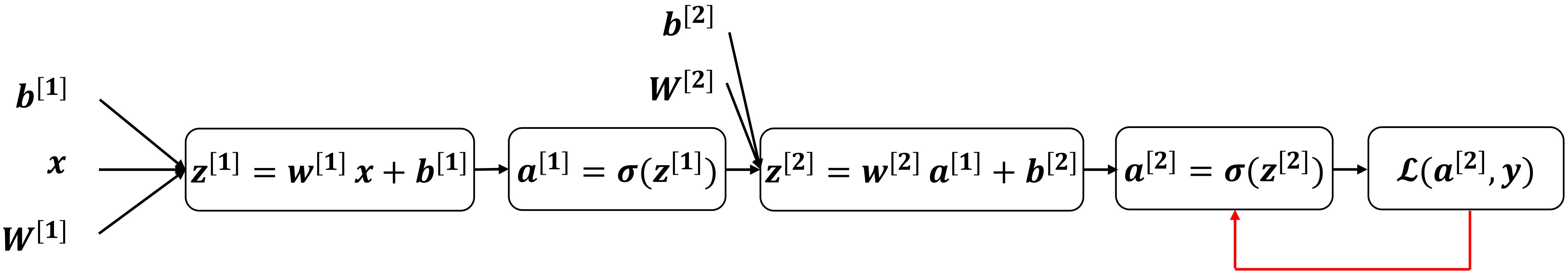
$b^{[2]}$
$W^{[2]}$
$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial a^{[2]}} = \text{Partial derivative of } \mathcal{L} \text{ with respect to } a^{[2]}$$

We are using the cross entropy loss as cost function for illustration

Backward Propagation

- The derivatives can be computed by moving from right to left

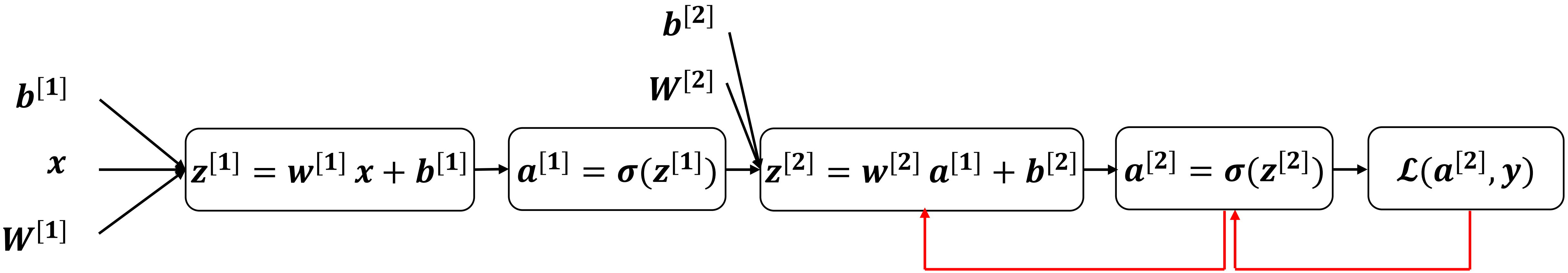


$\boldsymbol{b}^{[1]}$
$\boldsymbol{x}$
$\boldsymbol{W}^{[1]}$
$\boldsymbol{z}^{[1]} = \boldsymbol{w}^{[1]}\boldsymbol{x} + \boldsymbol{b}^{[1]}$
$\boldsymbol{a}^{[1]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[1]})$
$\boldsymbol{b}^{[2]}$
$\boldsymbol{W}^{[2]}$
$\boldsymbol{z}^{[2]} = \boldsymbol{w}^{[2]}\boldsymbol{a}^{[1]} + \boldsymbol{b}^{[2]}$
$\boldsymbol{a}^{[2]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[2]})$
$\boldsymbol{\mathcal{L}}(\boldsymbol{a}^{[2]}, \boldsymbol{y})$

$$\frac{\partial \boldsymbol{\mathcal{L}}}{\partial \boldsymbol{a}^{[2]}} = \frac{-\boldsymbol{y}}{\boldsymbol{a}^{[2]}} + \frac{(\boldsymbol{1} - \boldsymbol{y})}{(\boldsymbol{1} - \boldsymbol{a}^{[2]})}$$

Backward Propagation

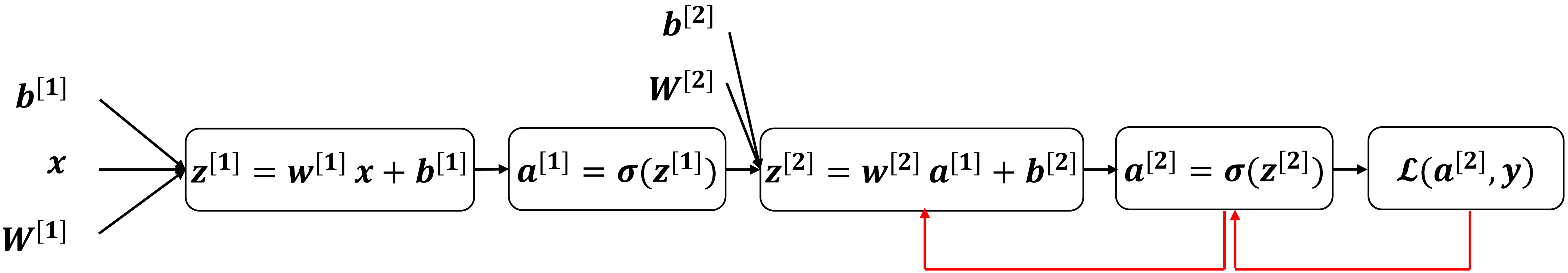
- The derivatives can be computed by moving from right to left


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial z^{[2]}} = \text{Partial derivative of } \mathcal{L} \text{ with respect to } z^{[2]}$$

Backward Propagation

- The derivatives can be computed by moving from right to left

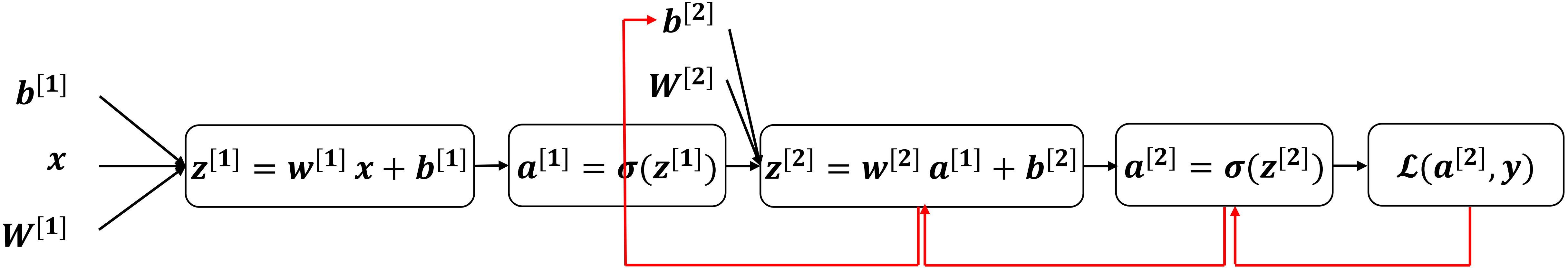


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial z^{[2]}} = \frac{\partial \mathcal{L}}{\partial a^{[2]}} \times \frac{\partial a^{[2]}}{\partial z^{[2]}} = a^{[2]} - y$$

Backward Propagation

- The derivatives can be computed by moving from right to left

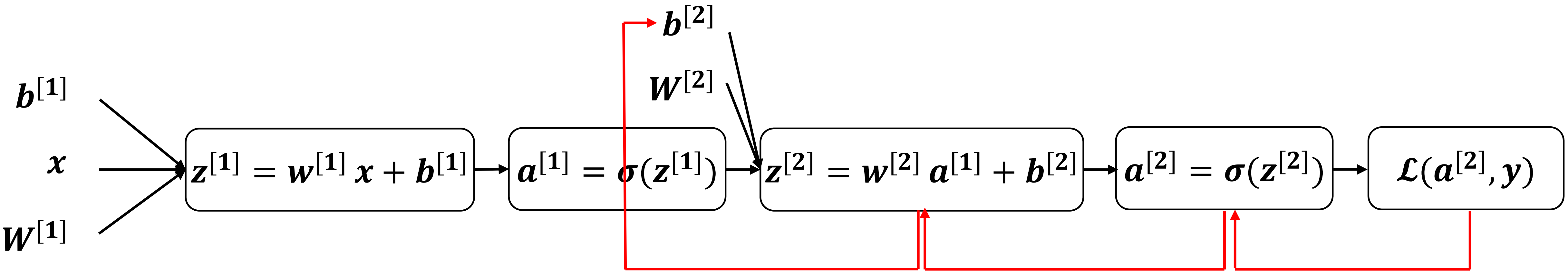


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial b^{[2]}} = \text{Partial derivative of } \mathcal{L} \text{ with respect to } b^{[2]}$$

Backward Propagation

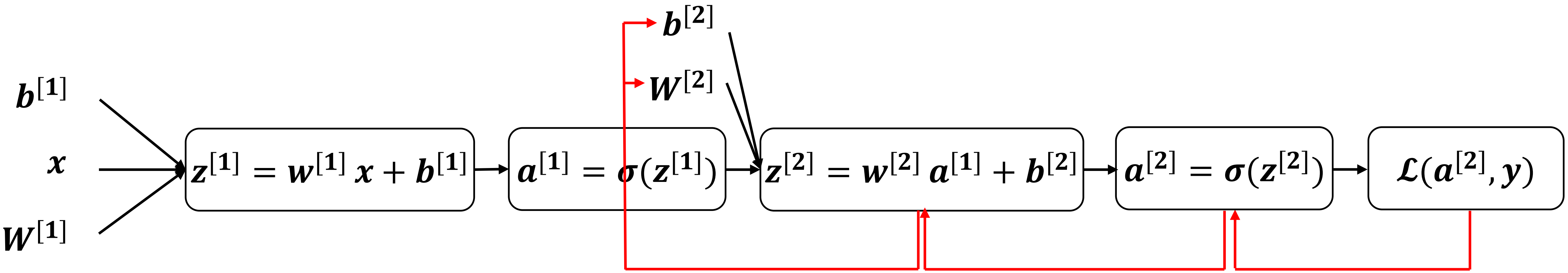
- The derivatives can be computed by moving from right to left


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]} x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]} a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial b^{[2]}} = \frac{\partial \mathcal{L}}{\partial a^{[2]}} \times \frac{\partial a^{[2]}}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial b^{[2]}} = a^{[2]} - y$$

Backward Propagation

- The derivatives can be computed by moving from right to left



$\boldsymbol{b}^{[1]}$

$\boldsymbol{x}$

$\boldsymbol{W}^{[1]}$

$\boldsymbol{z}^{[1]} = \boldsymbol{w}^{[1]}\,\boldsymbol{x} + \boldsymbol{b}^{[1]}$

$\boldsymbol{a}^{[1]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[1]})$

$\boldsymbol{b}^{[2]}$

$\boldsymbol{W}^{[2]}$

$\boldsymbol{z}^{[2]} = \boldsymbol{w}^{[2]}\,\boldsymbol{a}^{[1]} + \boldsymbol{b}^{[2]}$

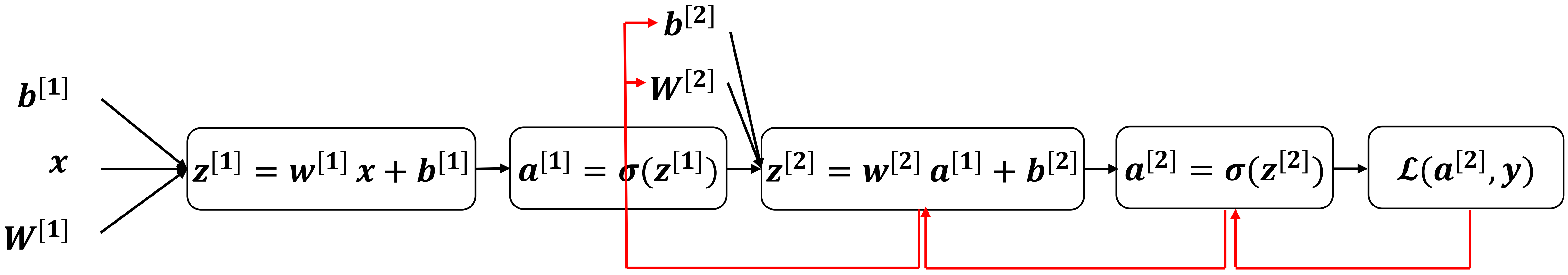
$\boldsymbol{a}^{[2]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[2]})$

$\boldsymbol{\mathcal{L}}(\boldsymbol{a}^{[2]}, \boldsymbol{y})$

$$\frac{\boldsymbol{\partial \mathcal{L}}}{\boldsymbol{\partial W}^{[2]}} = \text{Partial derivative of } \boldsymbol{\mathcal{L}} \text{ with respect to } \boldsymbol{W}^{[2]}$$

Backward Propagation

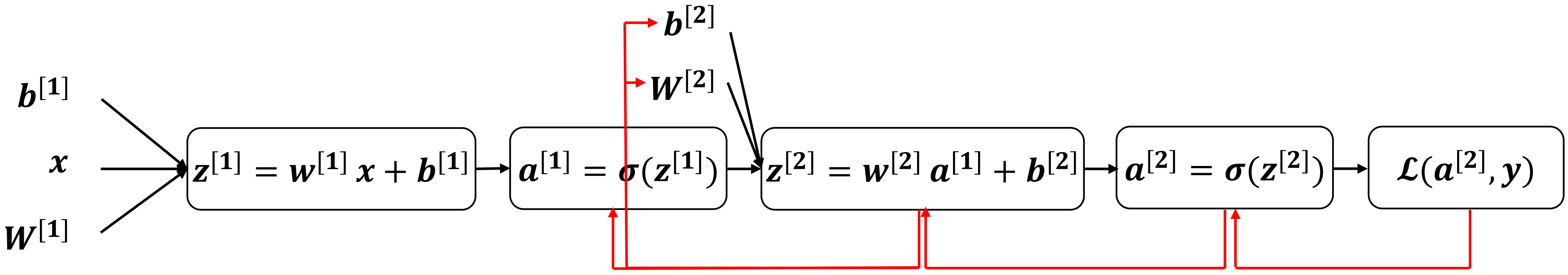
- The derivatives can be computed by moving from right to left


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]} x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]} a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial W^{[2]}} = \frac{\partial \mathcal{L}}{\partial a^{[2]}} \times \frac{\partial a^{[2]}}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial W^{[2]}} = (a^{[2]} - y) a^{[1]T}$$

Backward Propagation

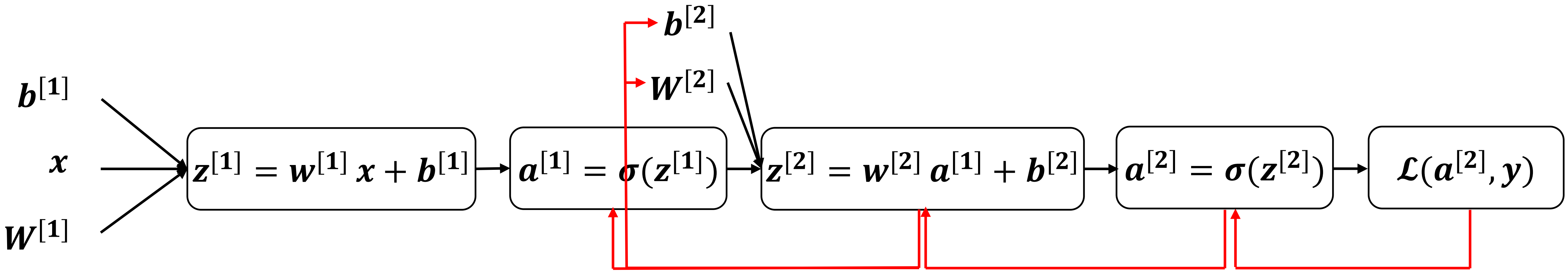
- The derivatives can be computed by moving from right to left


$\boldsymbol{b}^{[1]}$
$\boldsymbol{x}$
$\boldsymbol{W}^{[1]}$
$\boldsymbol{z}^{[1]} = \boldsymbol{w}^{[1]}\,\boldsymbol{x} + \boldsymbol{b}^{[1]}$
$\boldsymbol{a}^{[1]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[1]})$
$\boldsymbol{b}^{[2]}$
$\boldsymbol{W}^{[2]}$
$\boldsymbol{z}^{[2]} = \boldsymbol{w}^{[2]}\,\boldsymbol{a}^{[1]} + \boldsymbol{b}^{[2]}$
$\boldsymbol{a}^{[2]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[2]})$
$\mathcal{L}(\boldsymbol{a}^{[2]}, \boldsymbol{y})$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{a}^{[1]}} = \text{Partial derivative of } \mathcal{L} \text{ with respect to } \boldsymbol{a}^{[1]}$$

Backward Propagation

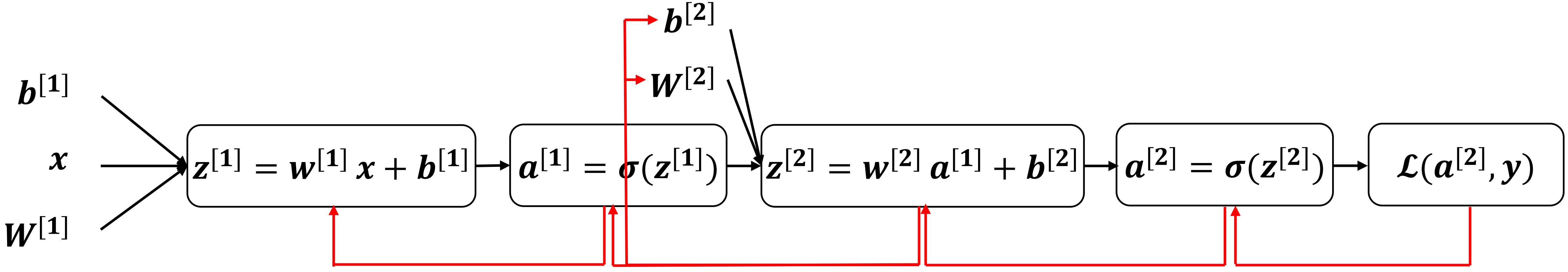
- The derivatives can be computed by moving from right to left


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial a^{[1]}} = \frac{\partial \mathcal{L}}{\partial a^{[2]}} \times \frac{\partial a^{[2]}}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial a^{[1]}} = (a^{[2]} - y)w^{[2]T}$$

Backward Propagation

- The derivatives can be computed by moving from right to left

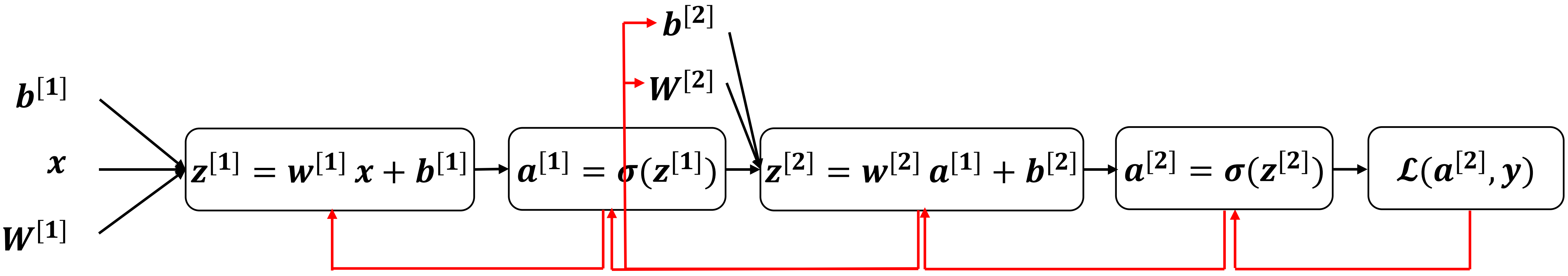


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$\dfrac{\partial \mathcal{L}}{\partial z^{[1]}} =$ Partial derivative of $\mathcal{L}$ with respect to $z^{[1]}$

Backward Propagation

- The derivatives can be computed by moving from right to left



$\boldsymbol{b}^{[1]}$

$\boldsymbol{x}$

$\boldsymbol{W}^{[1]}$

$\boldsymbol{z}^{[1]} = \boldsymbol{w}^{[1]}\boldsymbol{x} + \boldsymbol{b}^{[1]}$

$\boldsymbol{a}^{[1]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[1]})$

$\boldsymbol{b}^{[2]}$

$\boldsymbol{W}^{[2]}$

$\boldsymbol{z}^{[2]} = \boldsymbol{w}^{[2]}\boldsymbol{a}^{[1]} + \boldsymbol{b}^{[2]}$

$\boldsymbol{a}^{[2]} = \boldsymbol{\sigma}(\boldsymbol{z}^{[2]})$

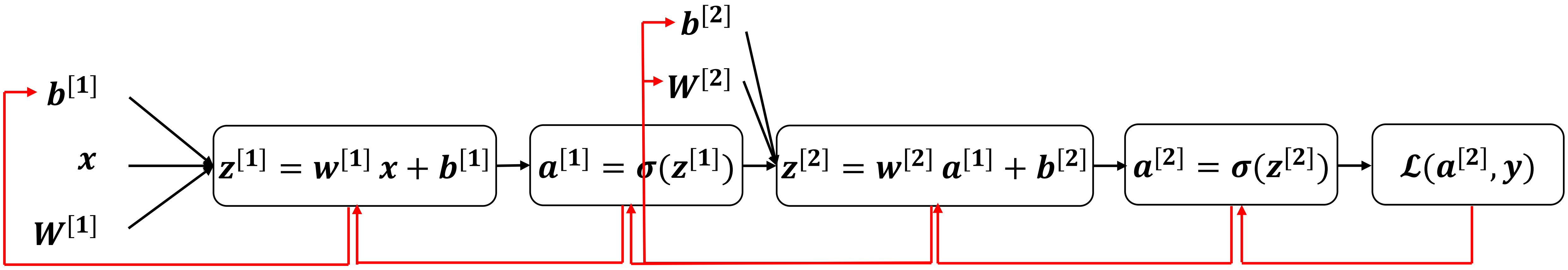
$\mathcal{L}(\boldsymbol{a}^{[2]}, \boldsymbol{y})$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}^{[1]}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{a}^{[2]}} \times \frac{\partial \boldsymbol{a}^{[2]}}{\partial \boldsymbol{z}^{[2]}} \times \frac{\partial \boldsymbol{z}^{[2]}}{\partial \boldsymbol{a}^{[1]}} \times \frac{\partial \boldsymbol{a}^{[1]}}{\partial \boldsymbol{z}^{[1]}} = (\boldsymbol{a}^{[2]} - \boldsymbol{y})\boldsymbol{w}^{[2]T} * \boldsymbol{a}^{[1]}(\mathbf{1} - \boldsymbol{a}^{[1]})$$

Element-wise product

Backward Propagation

- The derivatives can be computed by moving from right to left

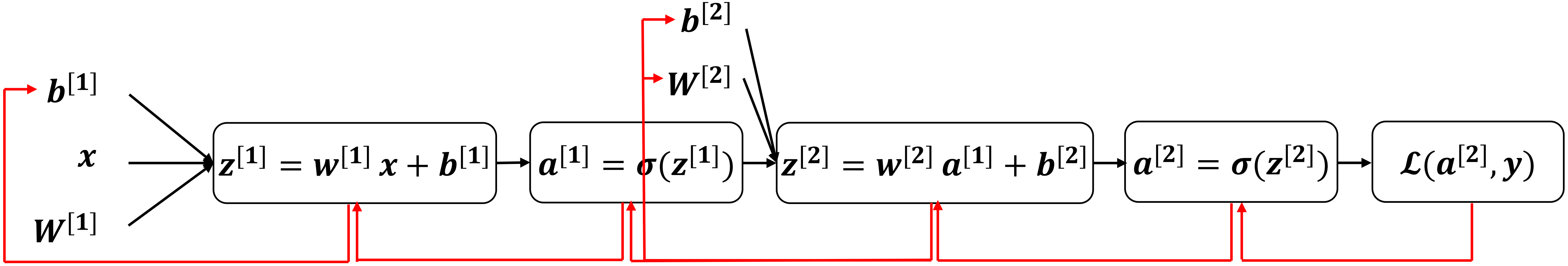


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial b^{[1]}} = \frac{\partial \mathcal{L}}{\partial a^{[2]}} \times \frac{\partial a^{[2]}}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial a^{[1]}} \times \frac{\partial a^{[1]}}{\partial z^{[1]}} \times \frac{\partial z^{[1]}}{\partial b^{[1]}}$$

Backward Propagation

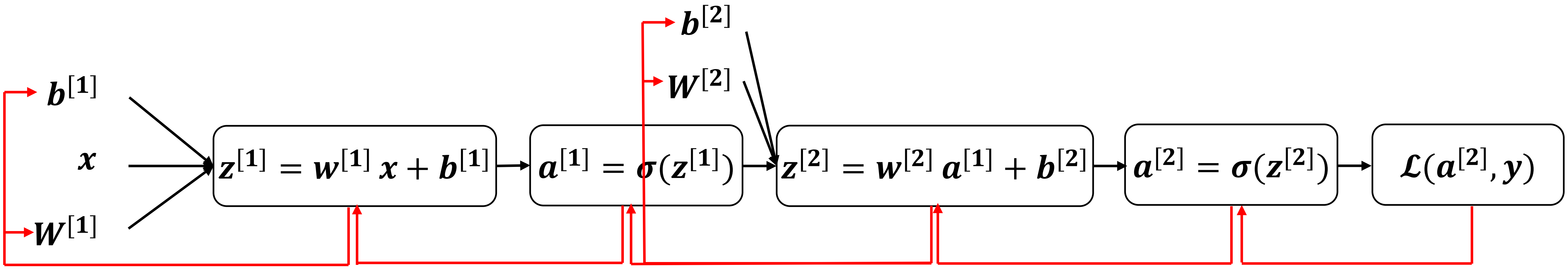
- The derivatives can be computed by moving from right to left


$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$b^{[2]}$
$W^{[2]}$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial b^{[1]}} = (a^{[2]} - y)w^{[2]T} * a^{[1]}(1 - a^{[1]})$$

Backward Propagation

- The derivatives can be computed by moving from right to left



$b^{[1]}$

x

$W^{[1]}$

$z^{[1]} = w^{[1]}x + b^{[1]}$

$a^{[1]} = \sigma(z^{[1]})$

$b^{[2]}$

$W^{[2]}$

$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$

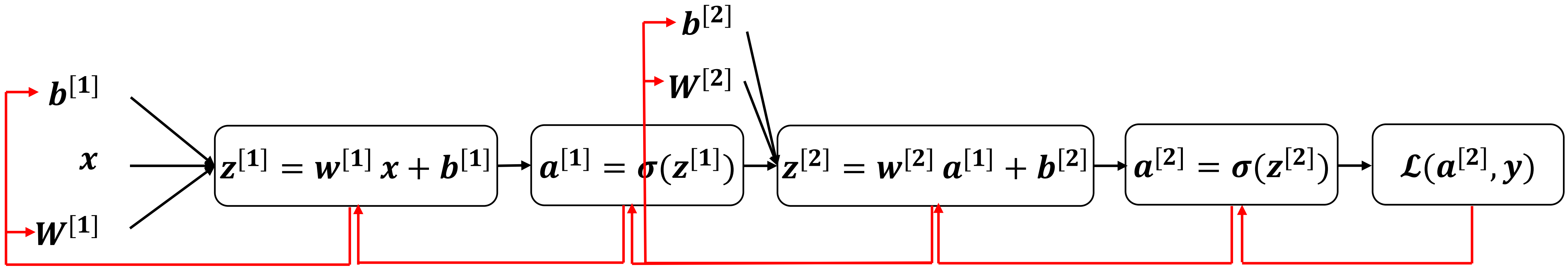
$a^{[2]} = \sigma(z^{[2]})$

$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial W^{[1]}} = \frac{\partial \mathcal{L}}{\partial a^{[2]}} \times \frac{\partial a^{[2]}}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial a^{[1]}} \times \frac{\partial a^{[1]}}{\partial z^{[1]}} \times \frac{\partial z^{[1]}}{\partial W^{[1]}}$$

Backward Propagation

- The derivatives can be computed by moving from right to left



$b^{[2]}$
$W^{[2]}$
$b^{[1]}$
x
$W^{[1]}$
$z^{[1]} = w^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\mathcal{L}(a^{[2]}, y)$

$$\frac{\partial \mathcal{L}}{\partial W^{[1]}} = \left((a^{[2]} - y)w^{[2]T} * a^{[1]}(1 - a^{[1]})\right) x^T$$

Backward Propagation: Summary

- Here is the summary of the gradients in our given neural network:

$$\boldsymbol{dz^{[2]}} = \frac{\partial \mathcal{L}}{\partial z^{[2]}} = a^{[2]} - y$$

$$\boldsymbol{db^{[2]}} = \frac{\partial \mathcal{L}}{\partial b^{[2]}} = a^{[2]} - y$$

$$\boldsymbol{dW^{[2]}} = \frac{\partial \mathcal{L}}{\partial W^{[2]}} = (a^{[2]} - y)a^{[1]T}$$

$$\boldsymbol{dz^{[1]}} = \frac{\partial \mathcal{L}}{\partial z^{[1]}} = dz^{[2]} w^{[2]T} * a^{[1]}(1 - a^{[1]})$$

$$\boldsymbol{db^{[1]}} = \frac{\partial \mathcal{L}}{\partial b^{[1]}} = dz^{[2]} w^{[2]T} * a^{[1]}(1 - a^{[1]})$$

$$\boldsymbol{dW^{[1]}} = \frac{\partial \mathcal{L}}{\partial W^{[1]}} = \left(dz^{[2]} w^{[2]T} * a^{[1]}(1 - a^{[1]})\right) x^{T}$$

How to train a Neural Network: Summary

1. Define the neural network structure
2. Initialize the model's parameters
3. Loop:
    - Implement forward propagation
    - Compute loss
    - Implement backward propagation to get the gradients
    - Update parameters (gradient descent)

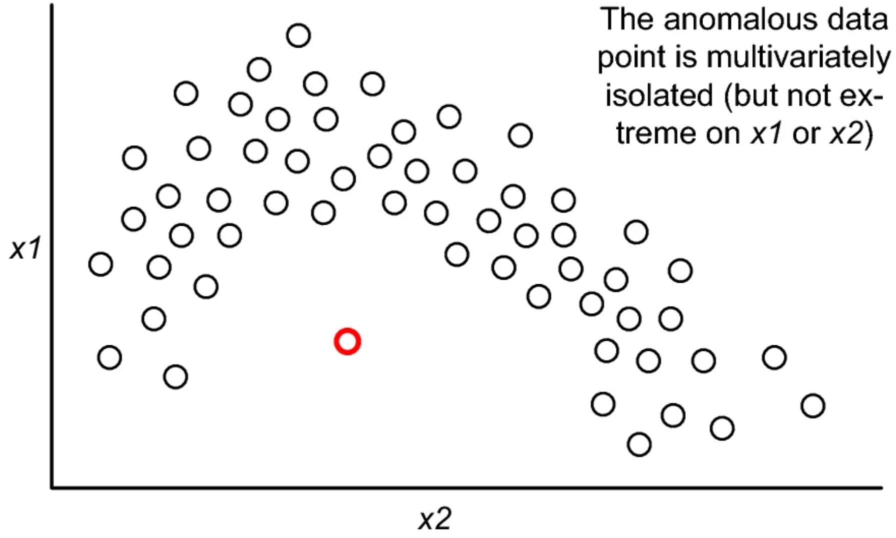
We will delve into more detail and discuss implementation during the lab practice.

What are anomalies?

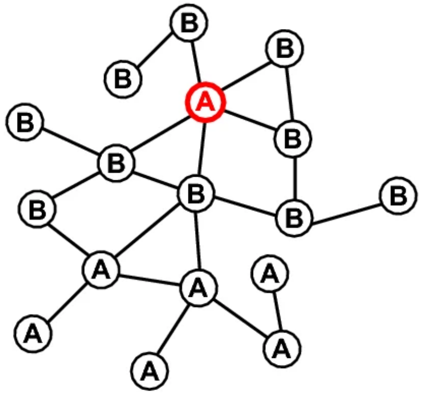
- Anomaly is a data object that deviates significantly from the rest of the objects
- Also referred to as outliers, exceptions, peculiarities, surprise, etc.
  - use “outlier” and “anomaly” interchangeably
- Anomalies translate to significant (often critical) real life entities
  - early signs of system failure
    - if the power company can find anomalies in the electrical power grid and remedy them, potentially avoid expensive damage
  - fraud in the financial industry
    - involves analyzing patterns of typical transactions and identifying deviations from those patterns in a large pool of legitimate transactions

Identification of anomalies

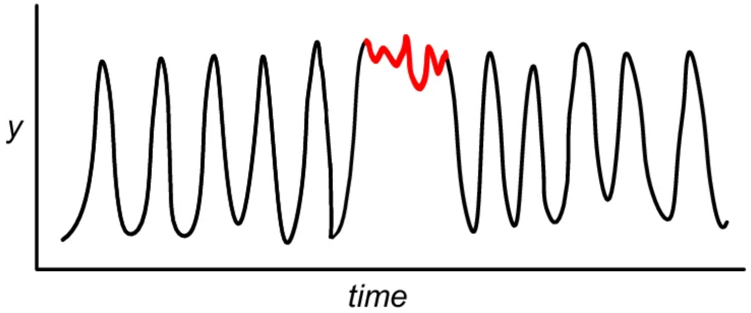
- Red occurrences illustrate a variety of anomalies



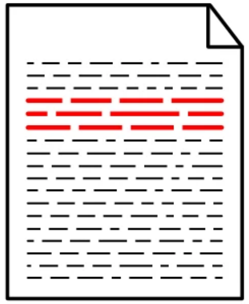
The anomalous data point is multivariately isolated (but not extreme on *x1* or *x2*)



The anomalous vertex has a different class label than its adjacent vertices.



The anomalous time interval deviates from the cyclical pattern.



The anomalous text section is comprised of unusually long words.

Image Source: On the nature and types of anomalies: a review of deviations in data, Ralph Foorthuis

Types of anomalies

- Global anomalies (Point anomalies)
  - simplest and the most common type
- Contextual anomalies (Conditional outliers)
  - requires background information to determine contextual attributes and contexts
- Collective anomalies
  - requires background information to model the relationship among objects to find groups of anomalies

Global anomalies

- A data object is a **global anomaly** if it deviates significantly from the rest of the data set
- Most anomaly detection methods are aimed at finding global anomalies
- Applications
  - Intrusion detection in computer networks
    - A large number of packages is broadcast in a short time
  - Suspicious transactions in trading systems
    - Transactions that do not follow the regulations

points in region R significantly deviate from the rest of the data set

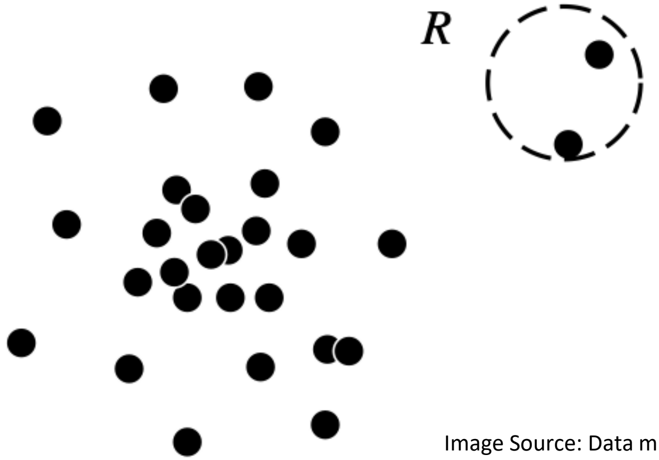


Image Source: Data mining concepts and techniques.

Contextual anomalies

- A data object is a **contextual anomaly** if it deviates significantly with respect to a specific context of the object. (Conditional outliers)
- “*The temperature today is 2◦C. Is it abnormal (i.e., an anomaly)*?”
- The context has to be specified as part of the problem definition
  - Contextual attributes: define context. e.g. location and time of year
  - Behavioral attributes: define characteristics. e.g. temperature and humidity
- Global anomaly detection can be regarded as a special case of contextual anomaly detection
  - where the set of contextual attributes is empty
- Application: credit card fraud detection

Collective anomalies

- A subset of data objects forms a **collective anomaly** if the objects as a whole deviate significantly from the entire data set
  - the individual data objects may not be anomalies
- *"1 order is delayed vs 100 orders are delayed on a single day"*
- Applications
  - Multiple denial-of-service packages as a whole
  - Many transactions of the same stock in a short period
- Consider the behavior of individual objects AND behavior of groups of objects

The black objects form a collective outlier

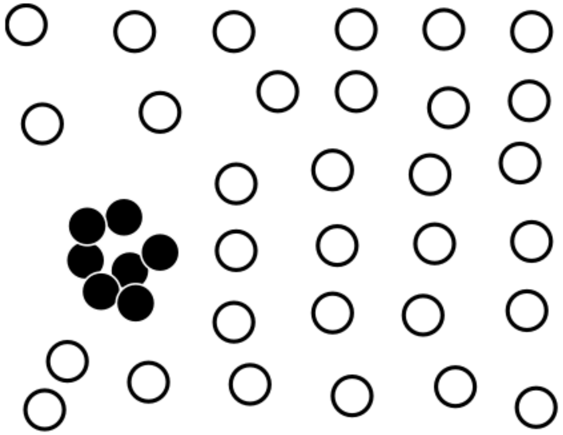


Image Source: Data mining concepts and techniques.

Anomaly detection

- In a general context:
  - Any method for finding events that don’t conform to an expectation
- In the context of network and host security
  - anomaly detection refers to identifying unexpected intruders or breaches
- Historically, the field of statistics tried to find and remove outliers as a way to improve analyses
- There are now many fields where the anomalies are the objects of greatest interest
- The rare events may be the ones with the greatest impact, and often in a negative way

Key applications of anomaly detection

- Fraud detection
- Industrial damage detection
- Cyber Intrusion detection
- …

Fraud Detection

- Fraud detection refers to detection of criminal activities occurring in commercial organizations
- Malicious users might be
  - actual customers of the organization, someone posing as a customer (also known as identity theft), employees of the company
- Types of fraud
  - Credit card fraud
  - Insurance claim fraud
  - Insider trading
- Key challenges
  - Fast and accurate real-time detection
  - Misclassification cost is very high

Image Source: https://www.milanfintechsummit.com/europe-card-fraud-digital-payments/

Industrial damage detection

- Detection of different faults and failures in complex industrial systems, structural damages, suspicious events in video surveillance, abnormal energy consumption, etc.
  - Example: aircraft safety
    - Anomalous Aircraft (Engine) / Fleet Usage
    - Anomalies in engine combustion data
    - Total aircraft health and usage management
- Key challenges
  - Data is extremely huge, noisy and unlabeled
  - Detecting anomalous events typically require immediate intervention

Image Source:
https://www.pointhacks.com.au/future-of-a380/

Intrusion detection

- Intrusion Detection
  - Process of monitoring the events occurring in a computer system or network and analyzing them for intrusions
  - using thresholds and heuristics is a simple way to detect intrusions and anomalies
- e.g.: intrusion detection with heuristics
  - suppose we define 10 queries/hour = upper limit of normal use for a certain database
  - Each time the database is queried, invoke a function is_anomaly(user) with the user’s ID as an argument
  - If the user queries the database for an 11th time within an hour, the function will indicate that access as an anomaly

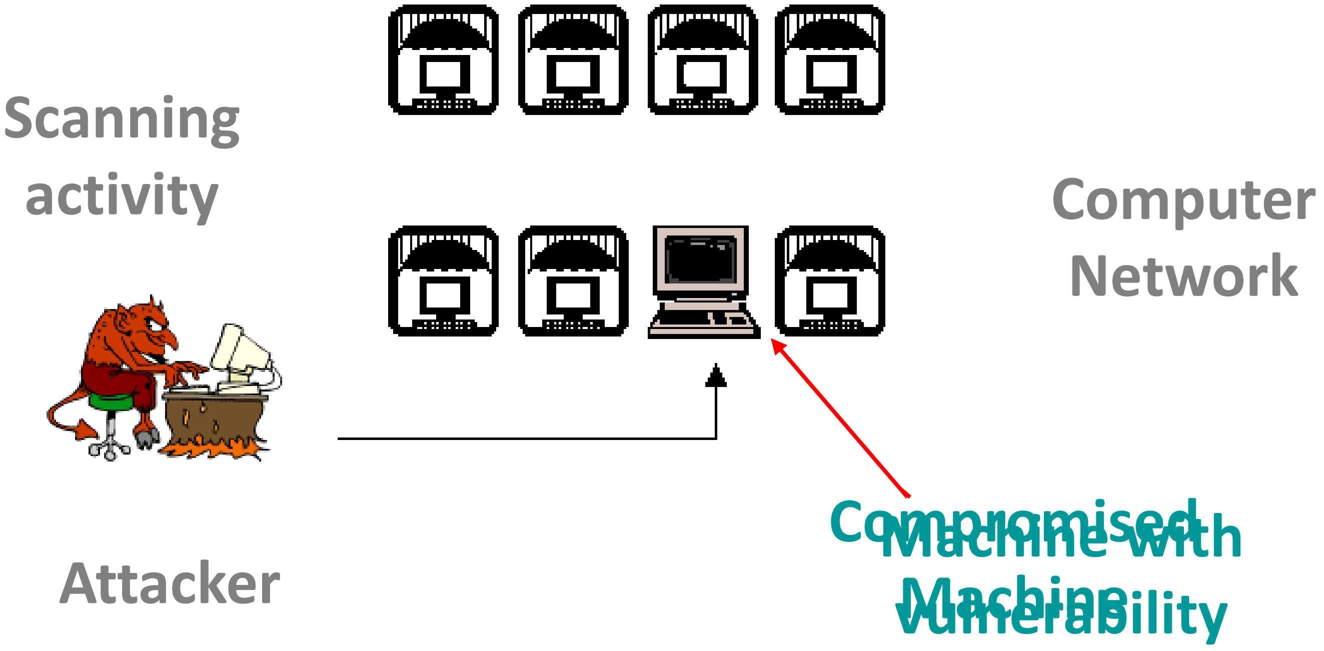
Intrusion Detection

- Although threshold-based anomaly detection logic is easy to implement, some questions arise:
  - How do we set the threshold?
  - Could some users require a higher threshold than others?
  - Could there be times when users legitimately need to access the database more often?
  - How frequently do we need to update the threshold?
  - Could an attacker exfiltrate data by taking over many user accounts, thus requiring a smaller number of accesses per account?
- Instead, using machine learning can help us to avoid having to come up with answers to all of these questions, letting the ***data*** define the solution to the problem

What are intrusions

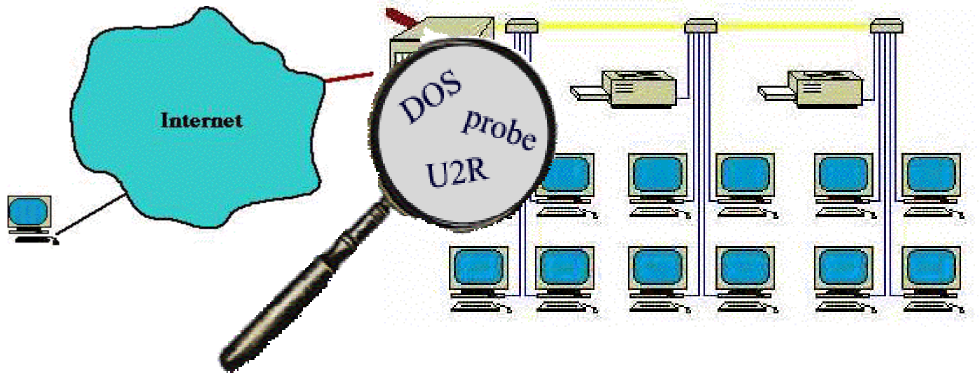
- Intrusions are actions that attempt to bypass security mechanisms of computer systems. They are usually caused by
  - Attackers accessing the system from Internet
  - Insider attackers - authorized users attempting to gain and misuse non-authorized privileges

Typical intrusion scenario



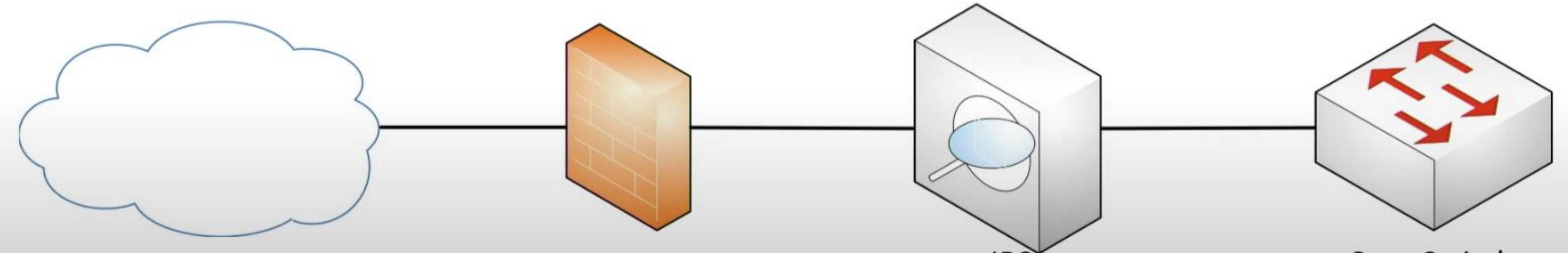
Intrusion Detection System

- Intrusion Detection System
  - monitors a network or systems for malicious activity or policy violations
  - raises alarms when possible intrusion happens
- In terms of location deployed
  - Network Intrusion Detection System (NIDS)
  - Host Intrusion Detection System (HIDS)
- In terms of detection methods
  - signature-based intrusion detection
  - anomaly-based intrusion detection



IDS – Network-based

- **NIDS** - installed on a network and monitor traffic for signs of attacks
- deployed at a strategic point or points within the network
    - e.g. at the network perimeter or at key junctions
    - monitor inbound and outbound traffic to and from all the devices on the network.
    - If NIDS detects a potential intrusion, it alerts the network administrator and take action to prevent the attack from succeeding
    - An example of a NIDS is installing it on the subnet where firewalls are located in order to see if someone is trying to crack the firewall.



Internet Firewall IDS Core Switch

IDS – Host-based

- **HIDS -** installed on individual hosts, and monitor the host's activity for signs of attacks
- computers or devices in the network with direct access to both the internet and the organization's internal network
  - e.g. servers or workstations
  - are typically used to monitor the host's logs, file system, and system calls, and can alert the administrator if they detect any suspicious activity
  - HIDS are often used to complement NIDS by providing additional visibility and protection for individual hosts on the network
- Both NIDS and HIDS are important tools for detecting and preventing intrusions, and many organizations use a combination of both to provide comprehensive protection for their networks

IDS - signature-based

- **Signature-based** intrusion detection
  - uses a database of known attack patterns or "signatures" to identify potential threats. This type of IDS is effective at detecting known attacks, but it may not be able to identify new or unknown threats
  - Signature-based IDS detects the attacks on the basis of the specific patterns
    - file hashes
    - byte sequences in network traffic
    - malicious instruction sequence that is used by the malware
- Signature-based IDS can easily detect the attacks whose pattern already exists in the system but it is quite difficult to detect the new malware attacks as their pattern is not known

IDS - anomaly-based

- **Anomaly-based** intrusion detection
  - identify unusual or suspicious behavior in network traffic. Anomaly-based detection is capable of alerting on unknown suspicious behavior
  - Anomaly-based detection involves first training the system with a normalized baseline and then comparing activity against that baseline
  - Once event appear out of the ordinary an alert is triggered
    - Alerts can be triggered by anything that does not align with the normalized baseline, e.g. a user logging in during non-business hours, a flood of new IP addresses attempting to connect to the network, or new devices being added to a network without permission
  - Machine learning-based method has a better-generalized property in comparison to signature-based IDS as these models can be trained according to the applications and hardware configurations

ML for intrusion detection: motivation

- Traditional intrusion detection system IDS tools (e.g. SNORT) are based on signatures of known attacks
  - Snort applies rules to monitored traffic and issues alerts when it detects certain kinds of suspicious activity on the network.
- Limitations
  - Signature database has to be manually revised for each new type of discovered intrusion
  - Substantial latency in deployment of newly created signatures across the computer system
  - They cannot detect emerging cyber threats

ML for intrusion detection

- Machine Learning and AI-driven techniques can alleviate these limitations – potential to recognize unforeseen attacks.
- Anomaly detection is based on profiles that represent normal behaviour of users, hosts, or networks, and detecting attacks as significant deviations from this profile
  - Major challenges: lack of sufficient amount of high-quality anomalous data
  - Major limitation: possible high false alarm rate, since detected deviations do not necessarily represent actual attacks
    - previously unseen (yet legitimate) system behaviors may also be recognized as anomalies
  - Major approaches: statistical methods, clustering, neural networks, support vector machines, …

Abnormal Behavior

- Some elements to consider in order to detect anomalous traffic
  - The number of connections to and from a specific host
  - Unusual remote communication ports or unexpected traffic patterns
  - Unusual traffic peaks occurring at particular times of the day (for example, traffic carrying on during the night)
  - Communication bandwidth heavily occupied by particular hosts within the network
- Abnormal behavior in network traffic
  - can be a sign of a security threat, such as a cyber attack or malware infection
  - can also indicate other problems, such as network congestion or faulty equipment

Feature engineering for anomaly detection

- As with any other task in machine learning, selecting good features for anomaly detection is important
  - For example, to detect when a system process has an abnormally high CPU utilization
  - all we need is the CPU utilization metric, which you can extract from most basic system monitoring modules
  - However, many use cases will require us to generate features on which to apply anomaly detection algorithms
- e.g.
  - host intrusion detection
  - network intrusion detection

Features engineering for HIDS

- Malware is the most dominant threat vector for hosts (e.g. servers, desktops) in many environments
- We can detect the malware by collecting system-level activity signals and looking for indicators of compromise (IoCs) in the data
  - Running processes
  - Active/new user accounts
  - Network connections
  - System scheduler changes
  - Startup operations
  - Temporary file directories
  - Browser extensions …

Features engineering for NIDS

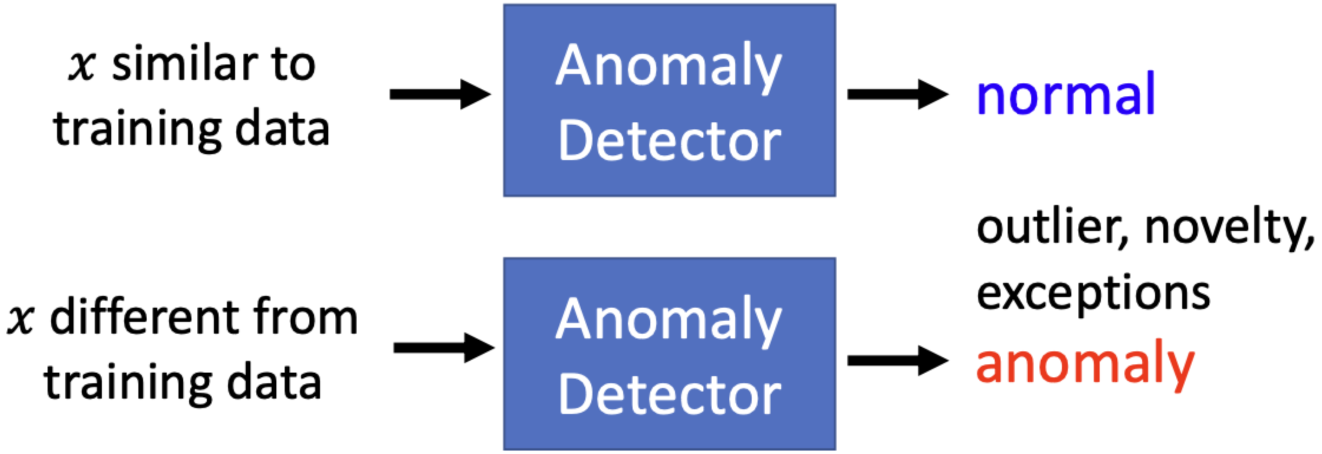
- In networking, a packet is a small segment of a larger message
  - Data sent over the Internet is divided into packets
  - These packets are then recombined by the computer or device that receives them
- There is a difference between
  - extracting network traffic metadata (info about packet’s contents, origin, and destination)
  - inspecting network traffic content (the actual data)
- Deep packet inspection (DPI)
  - the process of examining the data encapsulated in network packets
  - DPI is capable of collecting signals that can help detect spam, malware, intrusions, and subtle anomalies
  - Real-time streaming DPI is a challenging problem in computer science
    - because of the computational requirements necessary to decrypt, disassemble, and analyze packets going through a network

Outline

- What are anomalies
- Types of anomalies
- Applications of anomaly detection
- Machine learning for intrusion detection
- Anomaly detection techniques
  - Problem definition
  - Categories
    - With labels – train classifiers
    - Without labels – statistical methods

Anomaly detection - problem definition

- Given a set of training data $\{x^1, x^2, \dots, x^N\}$
- Find a model detecting whether input x is similar to training data



x similar to training data
Anomaly Detector
normal
outlier, novelty, exceptions
x different from training data
Anomaly Detector
anomaly

- Different approaches use different ways to determine the similarity.

Categories

- Given training data $\{x^1, x^2, \ldots, x^N\}$, two categories:

- 1. With labels $\{y^1, y^2, \ldots, y^N\}$
  - Supervised methods: train a classifier, e.g. logistic regression, nn, …
  - The classifier can output “unknown”
  - (Note that none of the training data is labelled as “unknown”)

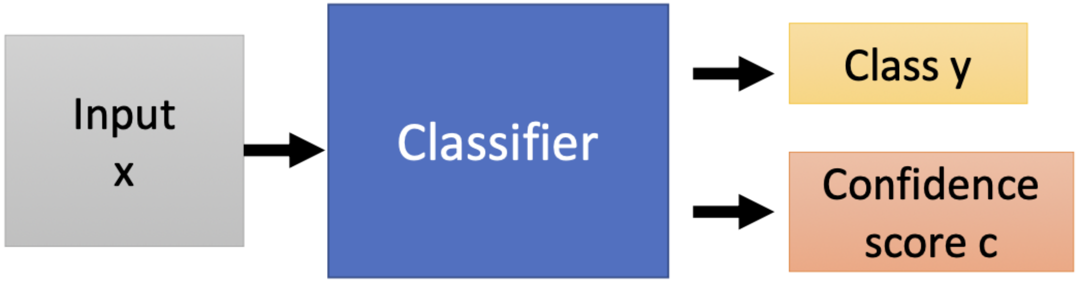
- 2. Without labels
  - Statistical approaches

Supervised methods

- Sample of data for analysis is given with domain expert–provided labels
- Anomaly detection is modeled as a classification problem
- Task: learn a classifier that can recognize anomalies
- Challenge
  - The two classes (normal vs anomalies) are imbalanced,
  - i.e. the number of normal samples likely far exceeds the number of outlier samples.
- Anomaly detection using a one-class model
  - A classifier is built to describe only the normal class.
  - Any samples that do not belong to the normal class are regarded as anomalies.

Supervised methods

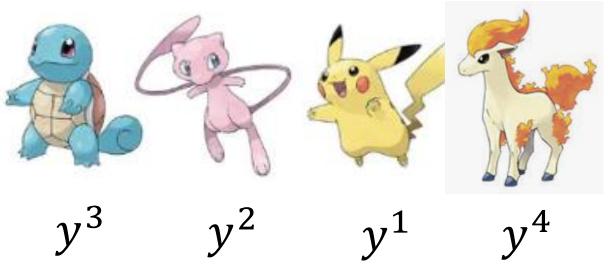
- Suppose you have trained a classifier using the labelled data
- How to use the classifier
  - Given input **x**
  - In addition to the prediction, output a confidence score **c**
  - Compare it with threshold λ



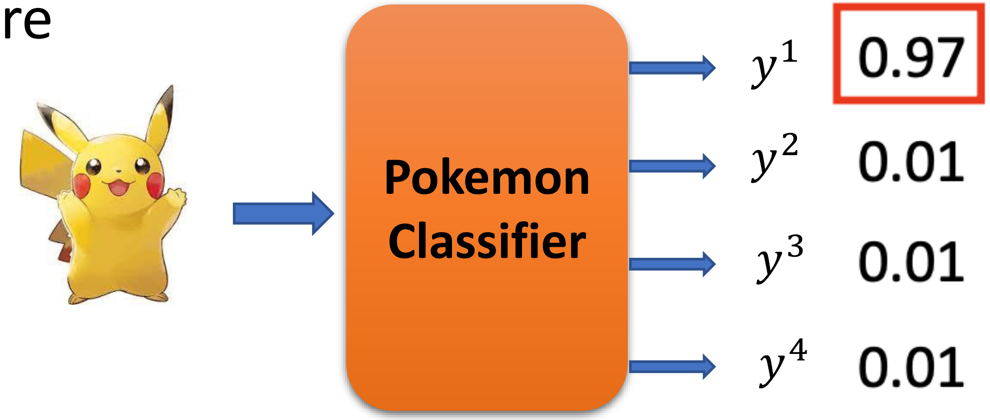
- Anomaly detection:

$$\mathrm{x} = \begin{cases} normal, & c(x) > \lambda \\ anomaly, & c(x) \leq \lambda \end{cases}$$

Example of confidence score



Confidence c: can be the maximum among the **softmax** score



Pokemon Classifier

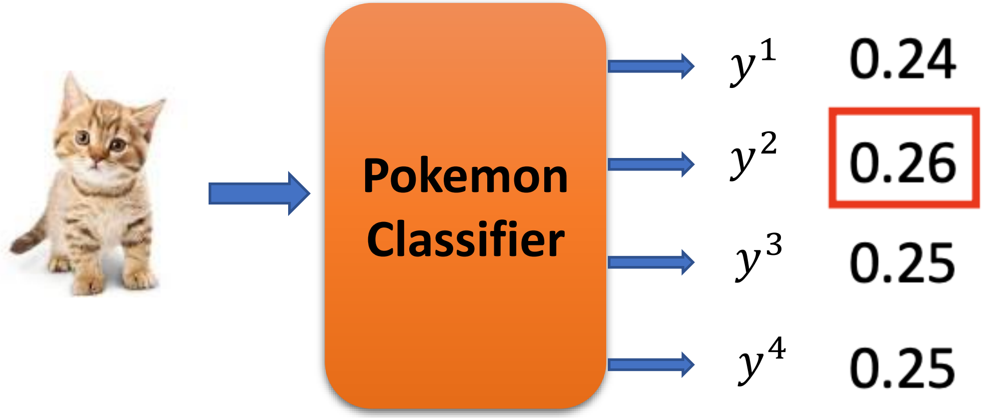
y^1 $\boxed{0.97}$

y^2 0.01

y^3 0.01

y^4 0.01

Very confident
Normal



Pokemon Classifier

y^1 0.24

y^2 $\boxed{0.26}$

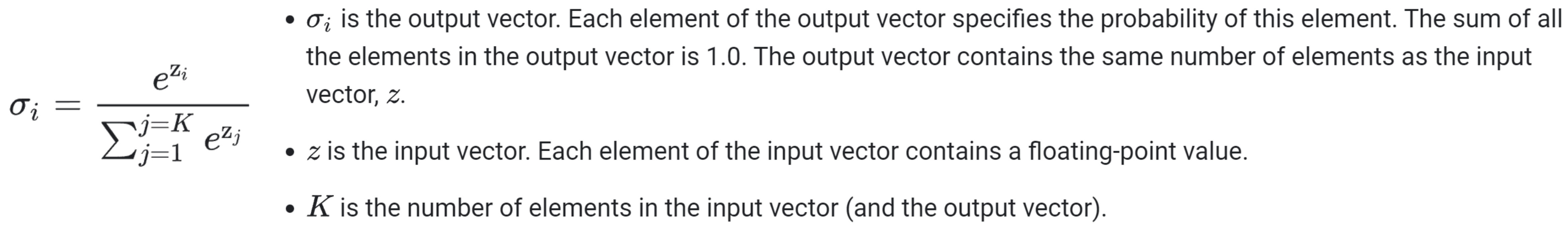
y^3 0.25

y^4 0.25

Not confident
Anomaly

Softmax

- Motivation: sigmoid produces a decimal between 0 and 1.0.
  - E.g, email classifier, output of 0.8 suggests: 80% = spam, 20% = ham.
  - the sum of the probabilities = 1.0
- Softmax function: **multiclass** classification
  - extends the idea into a multi-class case.
  - assigns decimal probabilities to each class in a multi-class problem.
  - The decimal probabilities must add up to 1.0.



$$\sigma_i = \frac{e^{\mathrm{z}_i}}{\sum_{j=1}^{j=K} e^{\mathrm{z}_j}}$$

- σ_i is the output vector. Each element of the output vector specifies the probability of this element. The sum of all the elements in the output vector is 1.0. The output vector contains the same number of elements as the input vector, z.
- z is the input vector. Each element of the input vector contains a floating-point value.
- K is the number of elements in the input vector (and the output vector).

Softmax - example

- Suppose the Input vector z is: [1.2, 2.5, 1.8]
- computes the denominator as: $\text{denominator} = e^{1.2} + e^{2.5} + e^{1.8} = 21.552$
- The softmax probability of each element is therefore:

$$\sigma_1 = \frac{e^{1.2}}{21.552} = 0.154$$

$$\sigma_i = \frac{e^{\mathbf{z}_i}}{\sum_{j=1}^{j=K} e^{\mathbf{z}_j}}$$

$$\sigma_2 = \frac{e^{2.5}}{21.552} = 0.565$$

$$\sigma_1 = \frac{e^{1.8}}{21.552} = 0.281$$

- The output vector is: $\sigma = [0.154, 0.565, 0.281]$

Softmax Function vs Sigmoid Function

- As mentioned before, the softmax function and the sigmoid function are similar.
  - The softmax operates on a vector while the sigmoid takes a scalar
- In fact, the sigmoid function is a special case of the softmax function for a classifier with only two input classes.
- Can you justify the statement above?
  - Hint: assume the input vector is [x, 0]

calculate the first output element with the usual softmax formula:

$$\sigma(\vec{z})_1 = \frac{e^{z_1}}{e^{z_1} + e^{z_2}} = \frac{e^{x}}{e^{x} + e^{0}} = \frac{e^{x}}{e^{x} + 1}$$

Simplifies to

$$\sigma(\vec{z})_1 = \frac{1}{1 + e^{-x}}$$

Example framework

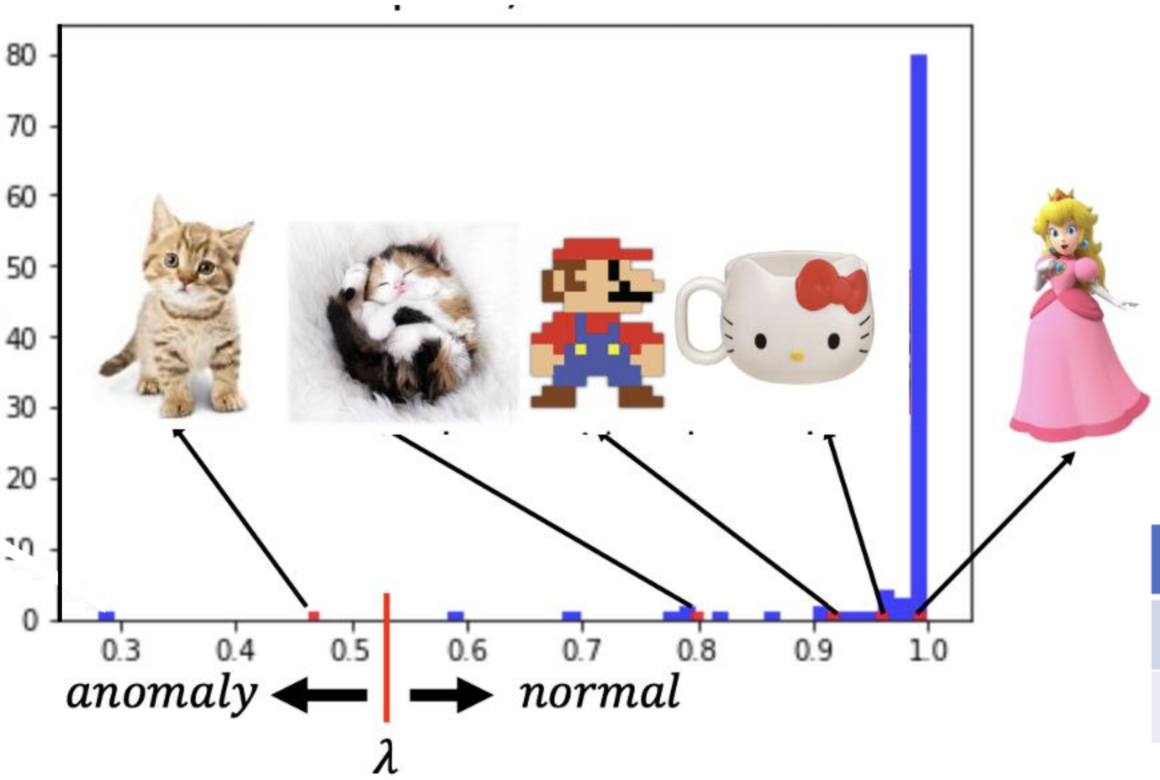
- Images x of characters from Pokemon
  - Each x is labelled by its characters y
  - train a classifier, obtain the confidence score c from the classifier

$$\mathbf{x} = \begin{cases} normal, & c(x) > \lambda \\ anomaly, & c(x) \leq \lambda \end{cases}$$

- Evaluate the performance of anomaly detector
  - determine λ and other hyperparameters, e.g. if using NN: size of the hidden layers, learning rate, # iterations, # layers
- Given new data x, from pokemon or not

Evaluation

100 pokemons, (normal: blue)
5 anomalies, (anomaly: red)



$$\mathbf{x} = \begin{cases} normal, & c(x) > \lambda \\ anomaly, & c(x) \le \lambda \end{cases}$$

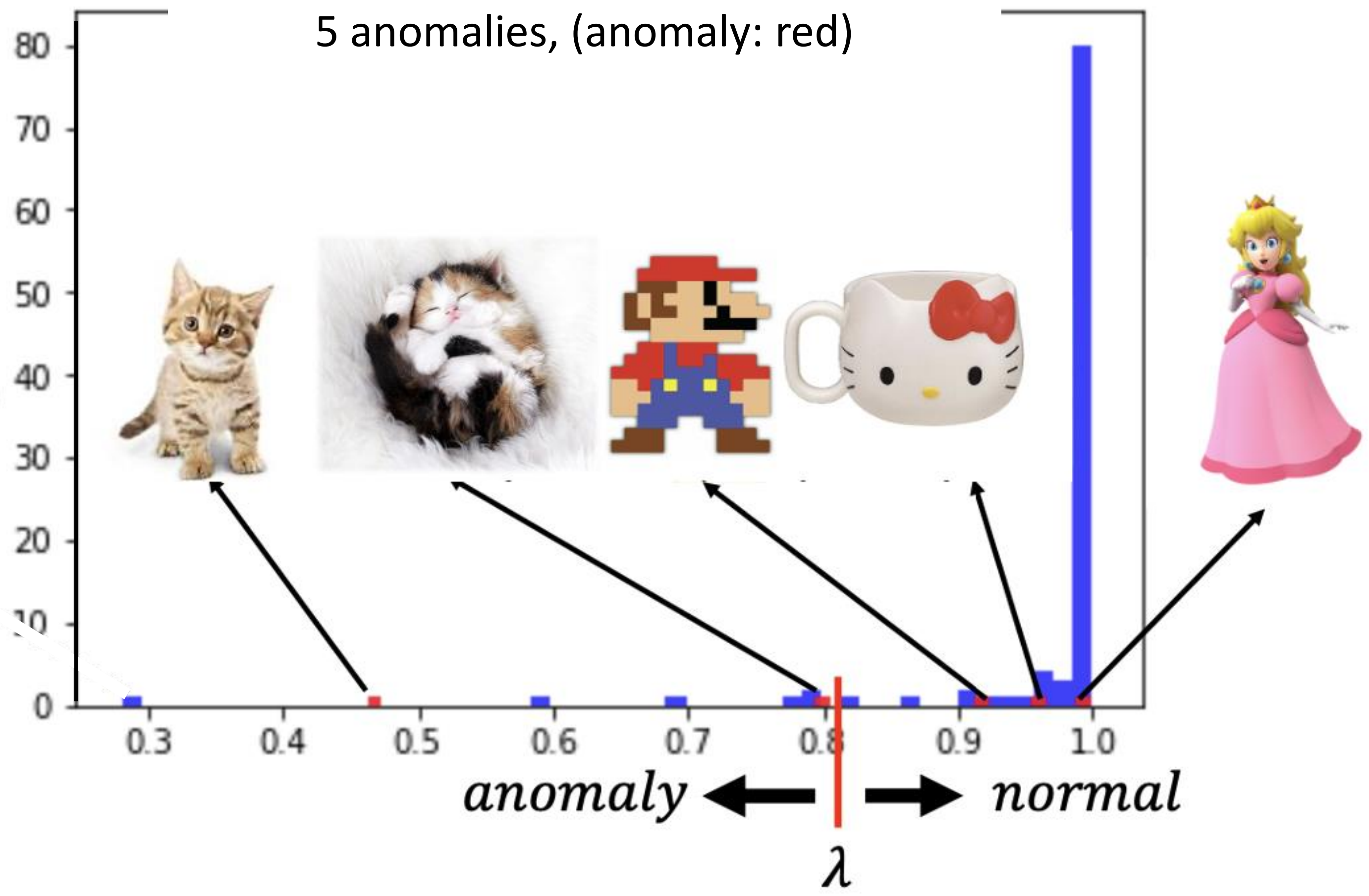
λ depends on specific application

	Anomaly	Normal
Detected	1	1
Not Det	4	99

False alarm

missing

100 pokemons, (normal: blue)
5 anomalies, (anomaly: red)



$$\mathbf{x} = \begin{cases} normal, & c(x) > \lambda \\ anomaly, & c(x) \le \lambda \end{cases}$$

	Anomaly	Normal
Detected	1	1
Not Det	4	99

vs

	Anomaly	Normal
Detected	2	6
Not Det	3	94

Evaluation

Define cost table to measure performance

	Anomaly	Normal
Detected	1	1
Not Det	4	99

Cost = 104 ✓

Cost = 401

	Anomaly	Normal
Detected	2	6
Not Det	3	94

Cost = 603

Cost = 306 ✓

Cost	Anomaly	Normal
Detected	0	100
Not Det	1	0

Cost Table A

Cost	Anomaly	Normal
Detected	0	1
Not Det	100	0

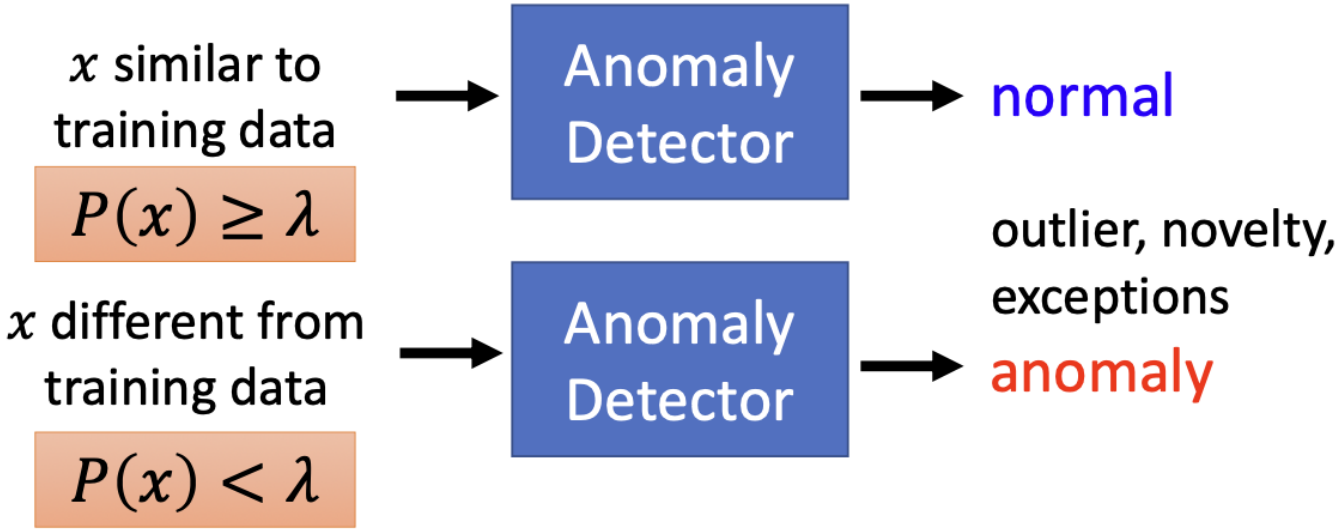
Cost Table B

Statistical Methods

- No labels available
- Assumption: normal objects are “clustered”
  - Normal objects follow a pattern far more frequently than anomalies
- General steps
  - Build a profile of normal behavior
    - summary statistics for overall population
    - find ”clusters”
  - Use the normal profile to detect anomalies
    - anomalies are observations whose characteristics differ significantly from the normal profile

Problem definition

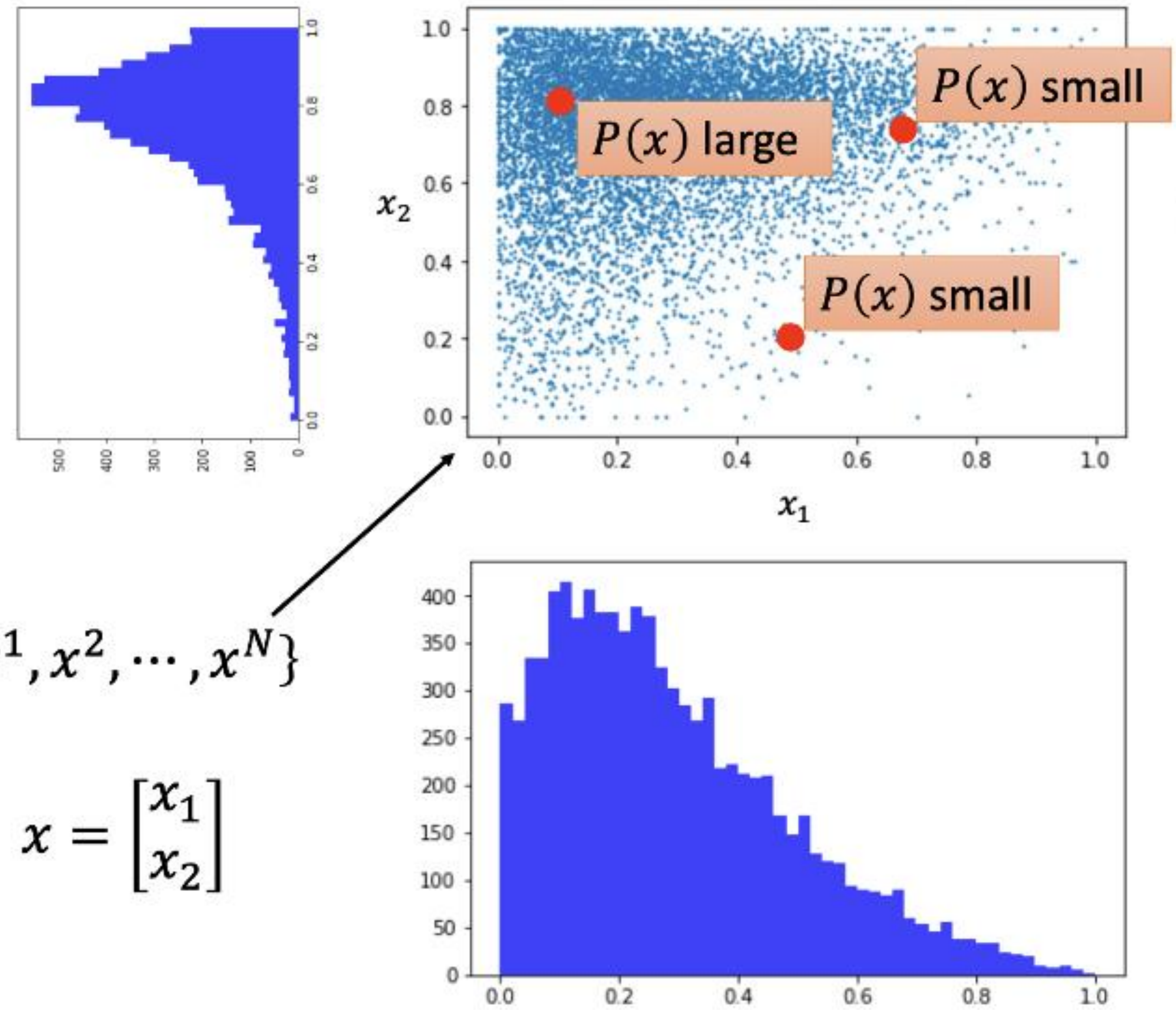
- Given training data $\{x^1, x^2, \ldots, x^N\}$, no labels available
- Find a model detecting whether input x is similar to training data


x similar to training data
$P(x) \geq \lambda$
Anomaly Detector
normal
x different from training data
$P(x) < \lambda$
Anomaly Detector
outlier, novelty, exceptions
anomaly

Example features

- Monitoring computers in a data center
  - x^i = features of machine i, such as:
    - x_1 = memory use, x_2 = network traffic, x_3 = number of disk access/sec …
- Fraud detection
  - x^i = features of user i's activities, e.g. purchase time, payment type …
- Industrial damage detection
  - x^i = features of aircraft i's engine, e.g. heat generated, vibration intensity…
- **Goal:**
  - model $P(x)$ from training data
  - Identify anomalies by checking whether $P(x) < \lambda$.

Illustration



$$\{x^1, x^2, \cdots, x^N\}$$

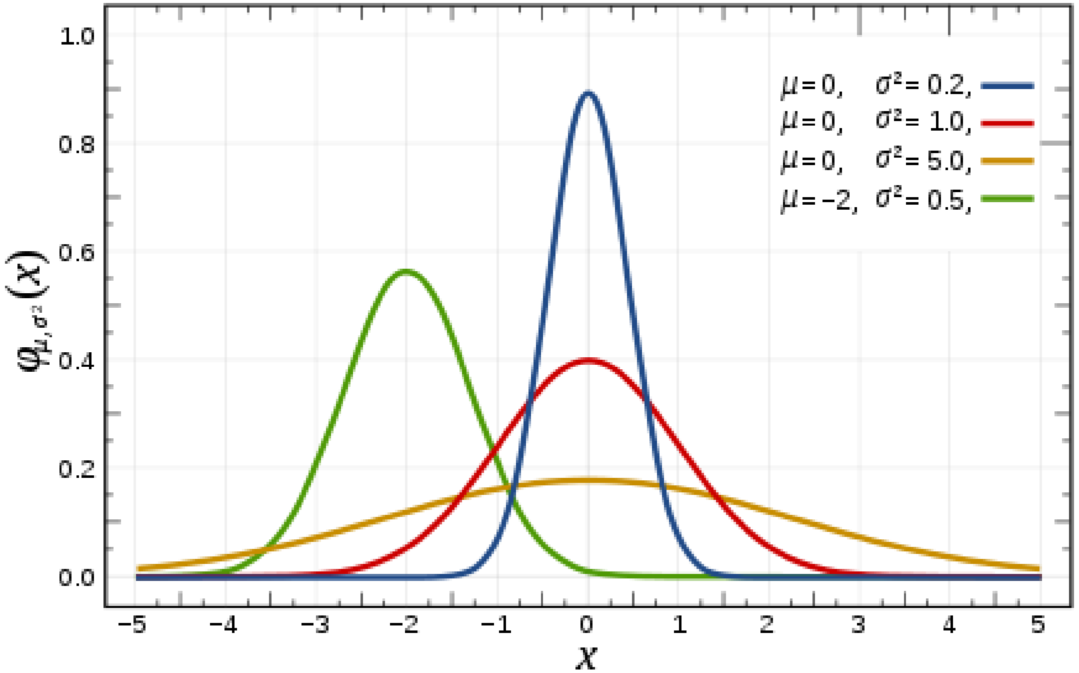
$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Gaussian distribution

- Gaussian (normal) distribution is a type of continuous probability distribution.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

- Given $x \in \mathbb{R}$, $x \sim N(\mu, \sigma^2)$: x is distributed Gaussian with mean μ, variance σ^2.



Parameter estimation

- Given data $\{x^1, x^2, \ldots, x^N\}$, $x^i \in \mathbb{R}$
- Assume $x^i \sim N(\mu, \sigma^2)$, how to learn the approximate values of parameters μ and σ?
  - Use the maximum likelihood method to estimate μ and σ:
  - set the mean of the Gaussian to be the mean of the data,
  - set the standard deviation of the Gaussian to be the standard deviation of the data.

$$\hat{\mu} = \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x^i$$

$$\hat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N} \left(x^i - \bar{x}\right)^2$$

Anomaly detection with Gaussian distribution

- Given training data $\{x^1, x^2, \dots, x^N\}$, $x^i \in \mathbb{R}^n$, n = # features
- Fit parameters $\mu_1, \mu_2, \dots, \mu_n;\ \sigma_1^2, \sigma_2^2, \dots, \sigma_n^2$
  - $\mu_j = \frac{1}{N}\sum_{i=1}^{N} \boxed{x_j^i}$ → *j-th* feature of data x^i
  - $\sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}(x_j^i - \mu_j)^2$
- Given new data $x \in \mathbb{R}^n$, compute $P(x)$:

$$P(x) = \prod_{j=1}^{n} p(x_j|\mu_j, \sigma_j^2) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_j} \exp(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2})$$

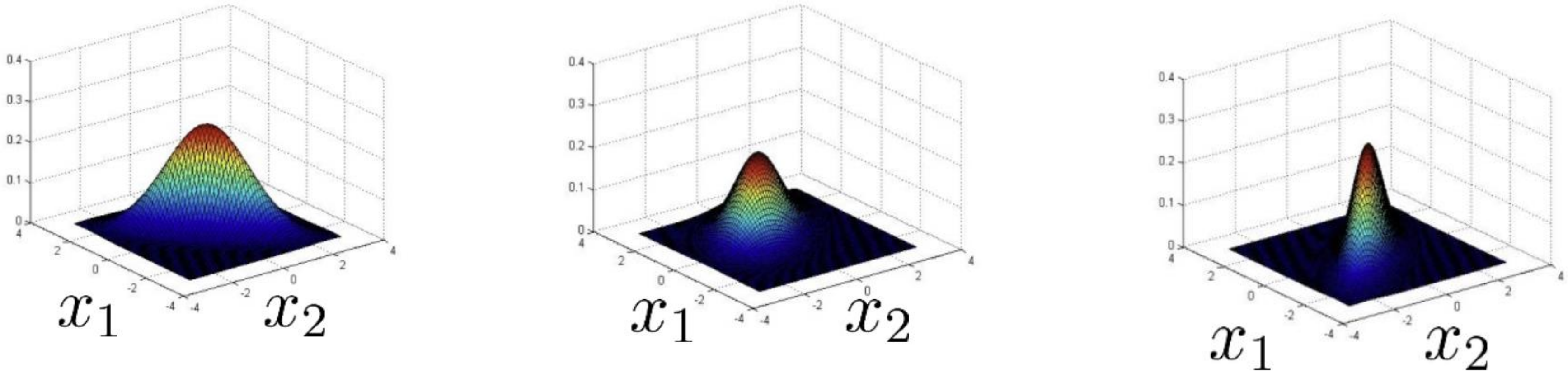
$$x = \begin{cases} normal, & if\ P(x) \geq \lambda \\ anomaly, & if P(x) < \lambda \end{cases}$$

Multivariate Gaussian distribution

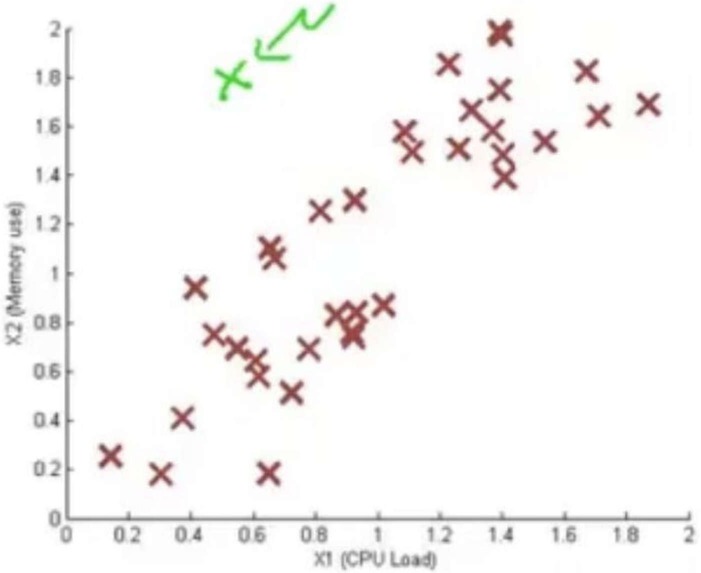
- Given data $\{x^1, x^2, \ldots, x^N\}$, $x^i \in \mathbb{R}^n$, n = # features
- Model $P(x)$ all in one go. (not model p(x) separately)

$$\boldsymbol{P(x)} = \frac{1}{(2\pi)^{n/2}} \frac{1}{|\Sigma|^{1/2}} exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}$$

- Input: vector x, output: P(x) (scaler)
- Parameters: $\mu \in \mathbb{R}^n$ (mean), $\Sigma \in \mathbb{R}^{n\times n}$(covariance matrix)



Anomaly detection with multivariate Gaussian distribution



1. Fit model $P(x)$ by computing

- $\mu = \frac{1}{N}\sum_{i=1}^{N} x^i$
- $\Sigma = \frac{1}{N}\sum_{i=1}^{N}(x^i - \mu)\,(x^i - \mu)^T$

2. Given new data x, compute

$$P(x) = \frac{1}{(2\pi)^{n/2}}\frac{1}{|\Sigma|^{1/2}} exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}$$

$$x = \begin{cases} normal, & if\ P(x) \geq \lambda \\ anomaly, & if P(x) < \lambda \end{cases}$$

n: dimension of x (# of features)
N: # of training examples

$|M|$: The determinant of a square matrix M, which is a number that can be calculated from M

Relationship to the original model

- Original model $P(x) = \prod_{j=1}^{n} p(x_j|\mu_j, \sigma_j^2) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_j} \exp(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2})$
  - Manually create features to capture anomalies where x_1 and x_2 take unusual combinations of values. e.g. $x_3 = \frac{x_1}{x_2} = \frac{CPU\ load}{memory}$
  - Computationally cheaper, scales better to large # features
  - OK even if # training examples is small

- Multivariate Gaussian model $P(x) = \frac{1}{(2\pi)^{n/2}} \frac{1}{|\Sigma|^{1/2}} exp\left\{-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right\}$
  - Automatically captures correlations between features
  - Computationally expensive
  - Must have N > n or else Σ is non-invertible.

Statistical Anomaly Detection

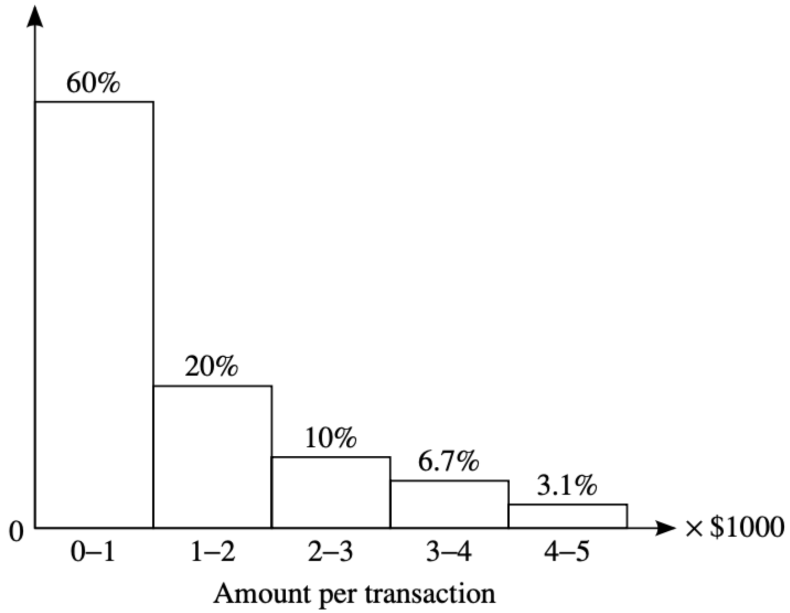
- Anomalies are objects that are fit poorly by a statistical model
  - Statistical methods assume normal data objects are generated by a statistical model, and data not following the model are anomalies

- Learn a statistical model describing the distribution of the data
  - <u>Parametric</u>: assumes that the normal data objects are generated by a parametric distribution with parameters, e.g. Gaussian distribution
  - **<u>Nonparametric</u>: tries to determine the model from the input data, e.g. histogram**

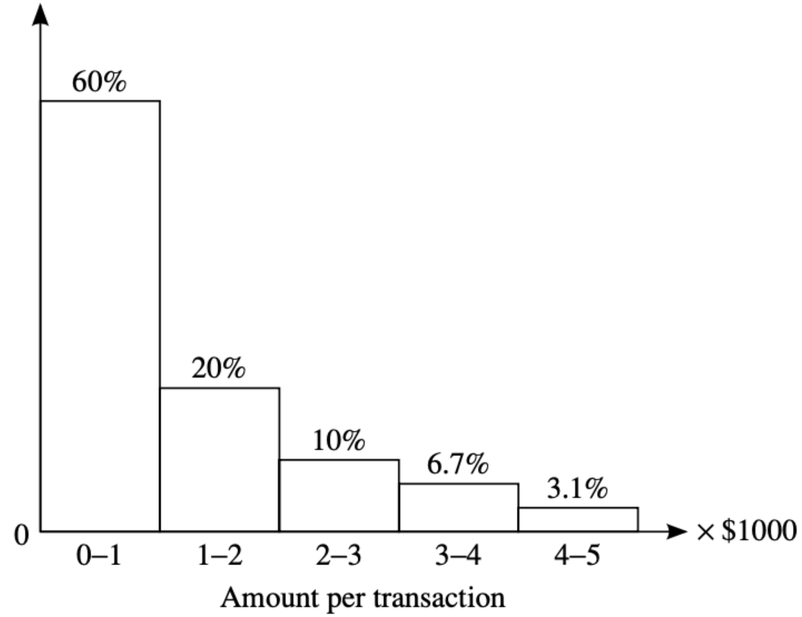
Nonparametric Methods

- The model of "normal data" is learned from the input data, rather than assuming one a priori
- Nonparametric methods often make fewer assumptions about the data, and thus can be applicable in more scenarios
- Anomaly detection using a histogram
  - A store records the purchase amount for every customer transaction.
  - e.g. 60% of the transaction amounts are between $0.00 and $1000

Histogram of purchase amounts in transactions.



Nonparametric Methods



- Idea: use the histogram as a nonparametric statistical model to capture anomalies
  - A transaction in the amount of $7500 can be regarded as an anomaly
    - 1 − (60% + 20% + 10% + 6.7% + 3.1%) = 0.2% of transactions have an amount higher than $5000
  - A transaction amount of $385 can be treated as normal
    - It falls into the bin (or bucket) holding 60% of the transactions
- Steps
  - Histogram construction: construct a histogram using input data
  - Anomaly detection: if the object falls in one of the histogram's bins, the object is regarded as normal. Otherwise, it is considered an outlier

References

- *Dorothy Denning, “An Intrusion-Detection Model,” IEEE Transactions on Sotware Engineering SE-13:2 (1987): 222–232.*
- *Data mining concepts and techniques, Jiawei Han, Micheline Kamber, Jian Pei*
- *Machine learning, Andrew Ng*
- *“Anomaly Detection: A Tutorial” Arindam Banerjee, Varun Chandola, Vipin Kumar, Jaideep Srivastava, University of Minnesota Aleksandar Lazarevic, United Technology Research Center*
- *Machine Learning and Security: Protecting Systems with Data and Algorithms, Clarence Chio, David Freeman*
- *Softmax function: Thomas Wood. deepai.org*