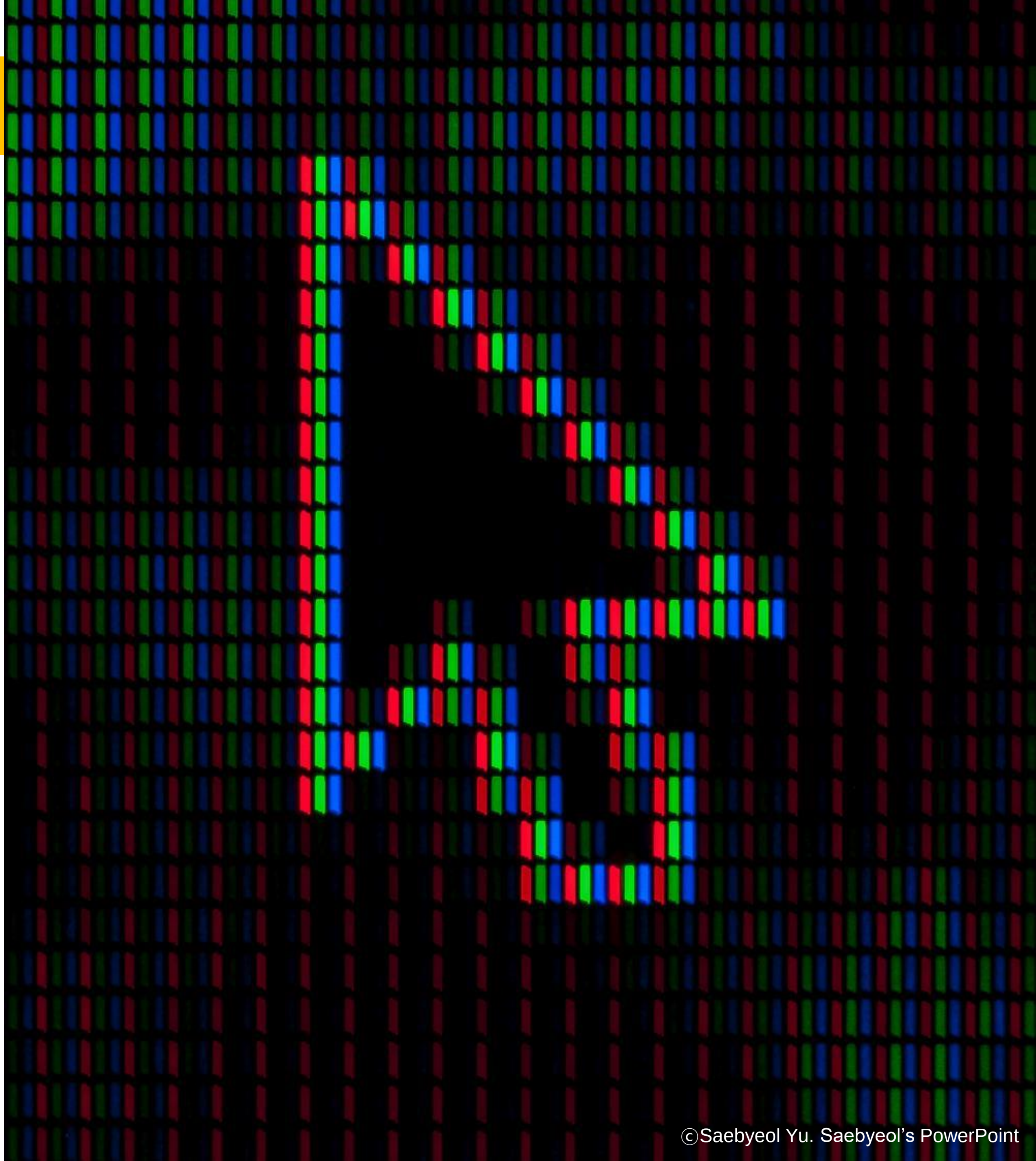
BACK PROPAGATION

9기 정주영

목차 a table of contents



Part 1 오차역전파법의 의미와 용도

Part 1

Back Propagation Definition

- 오차역전파법(Back-propagation)이란?
- 다층 신경망 학습에 사용되는 통계적 기법으로 출력층에서 발생한 결과를 입력층 방향으로 전송하면서 오차의 수정을 통해 가중치를 재설정하는 과정을 말한다.

전제

신경망에는 적응 가능한 가중치와 편향이 있고, 이 가중치와 편향을 훈련 데이터에 적응하도록 조정하는 과정을 '학습'이라 합니다. 신경망 학습은 다음과 같이 4단계로 수행합니다.

1단계 – 미니배치

훈련 데이터 중 일부를 무작위로 가져옵니다. 이렇게 선별한 데이터를 미니배치라 하며, 그 미니배치의 손실 함수 값을 줄이는 것이 목표입니다.

2단계 – 기울기 산출

미니배치의 손실 함수 값을 줄이기 위해 각 가중치 매개변수의 기울기를 구합니다. 기울기는 손실 함수의 값을 가장 작게 하는 방향을 제시합니다.

3단계 – 매개변수 갱신

가중치 매개변수를 기울기 방향으로 아주 조금 갱신합니다.

4단계 – 반복

1~3단계를 반복합니다.

수치미분

오차역전파법

Part 1

Back Propagation Reason to use

	장점	단점
수치미분법	만들기 쉬움	느림
오차역전파법	만들기 복잡함	빠름

```
grad_numerical = network.numerical_gradient(x_batch, t_batch)
[18]  ✓ 11.1s

grad_backprop = network.gradient(x_batch, t_batch)
[19]  ✓ 0.2s
```

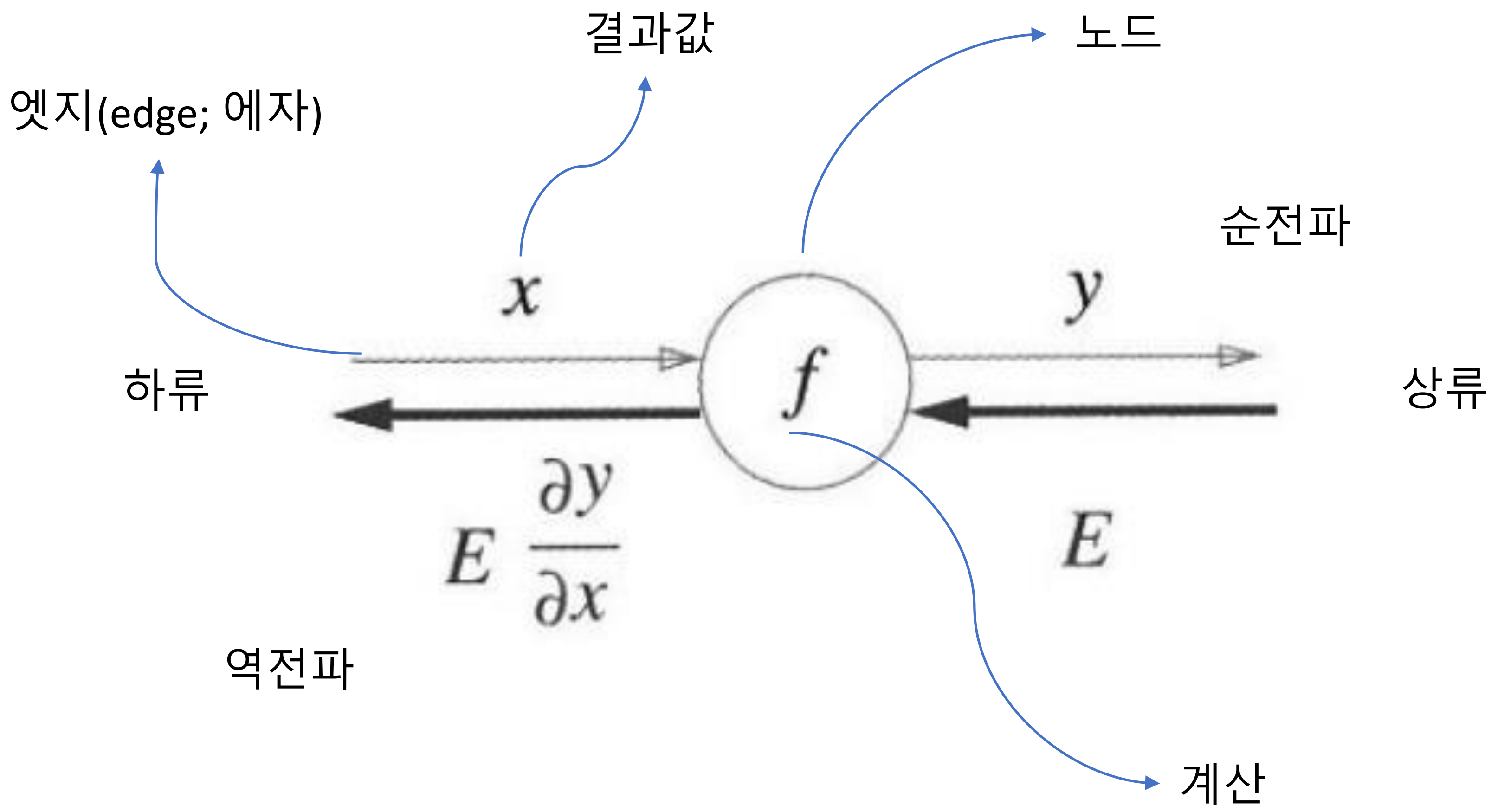
Efficient!

Part 2 계산그래프로 설명하는 오차 역전파법

Part 2

Back Propagation

How to calculate via graph

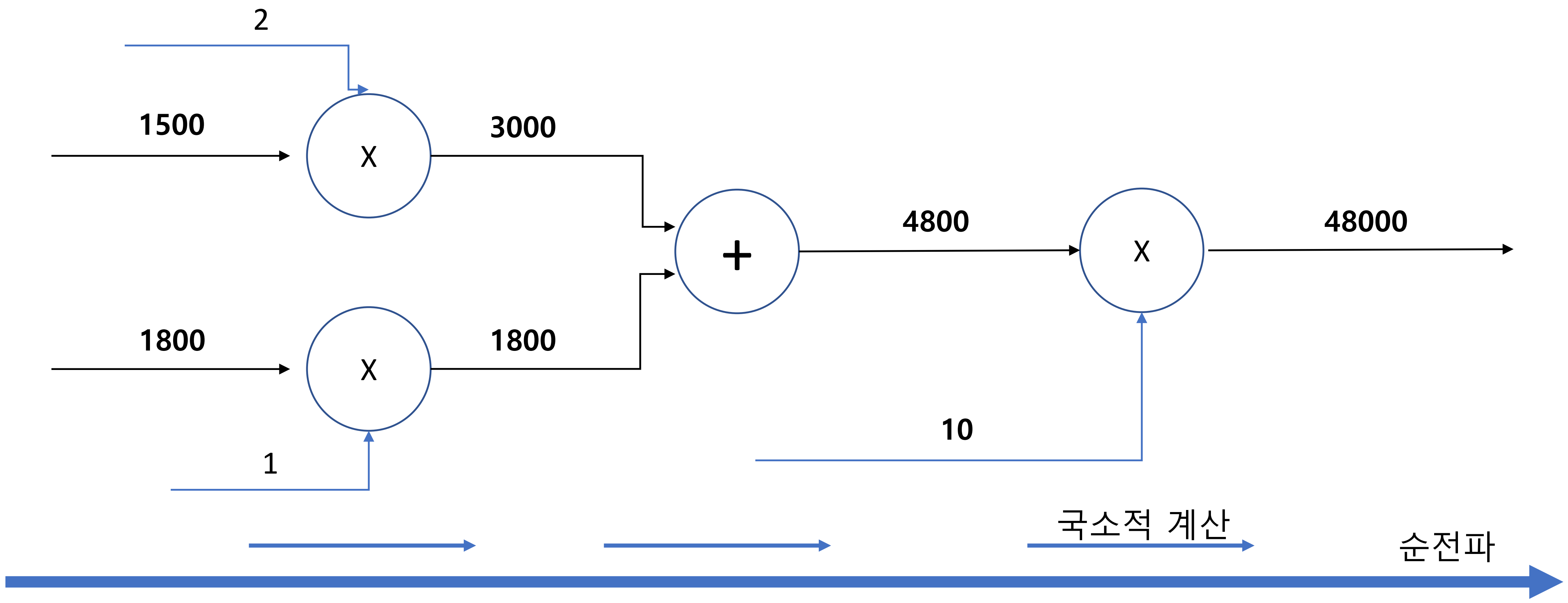


Part 2

Back Propagation

How to calculate via graph – Quiz!

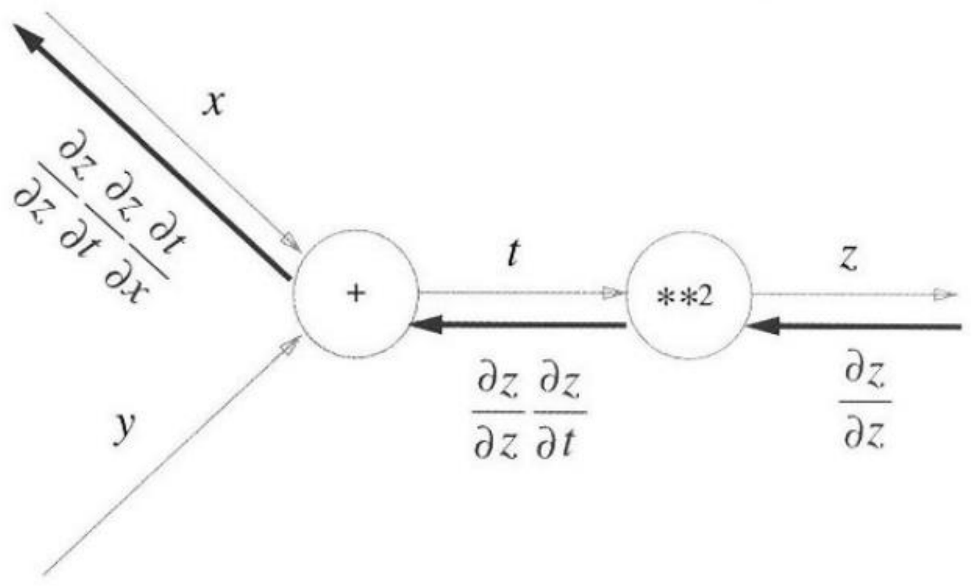
문제 : DSL 학회원 1명은 MT에서 물 2병과 주스 1병을 소비합니다.
물은 1병에 1500원, 주스는 1에 1800원이라고 합니다. MT를 가는 학회원이 총 10명이라면 음료 구매비는 총 얼마일까요?

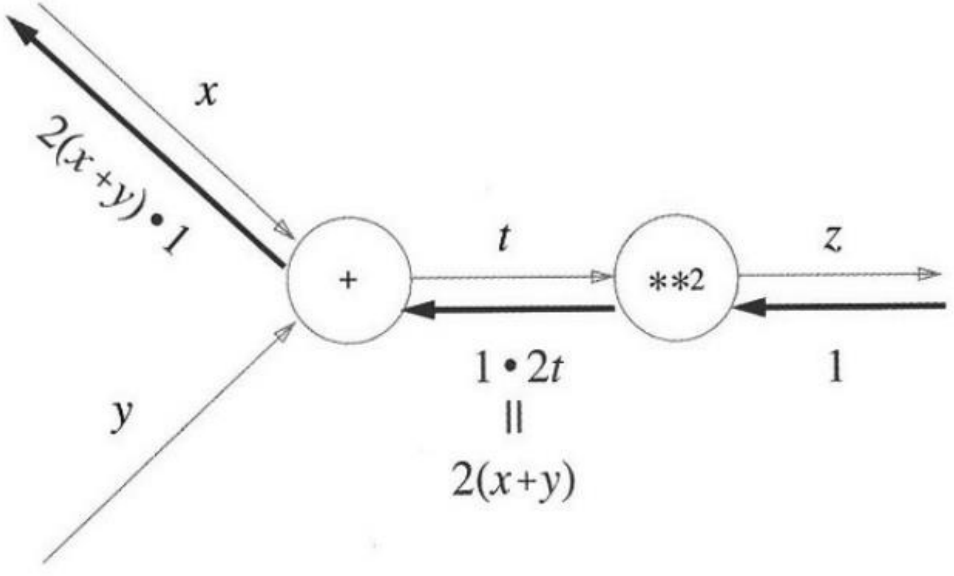


Part 2

Back Propagation

How to calculate via graph





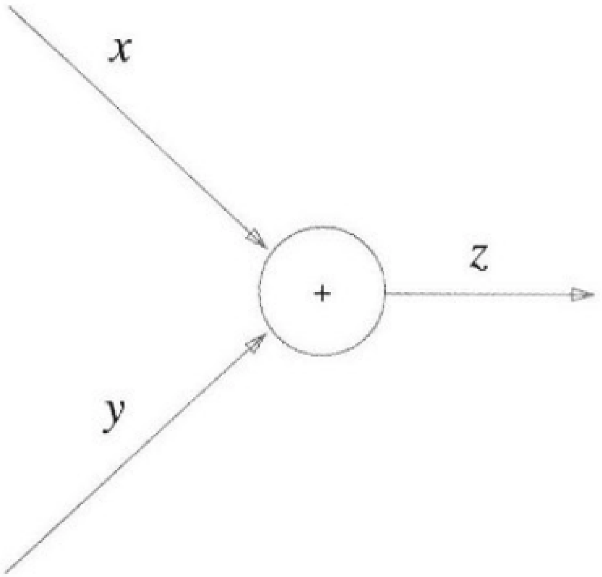
$$z = t^2$$
$$t = x + y$$

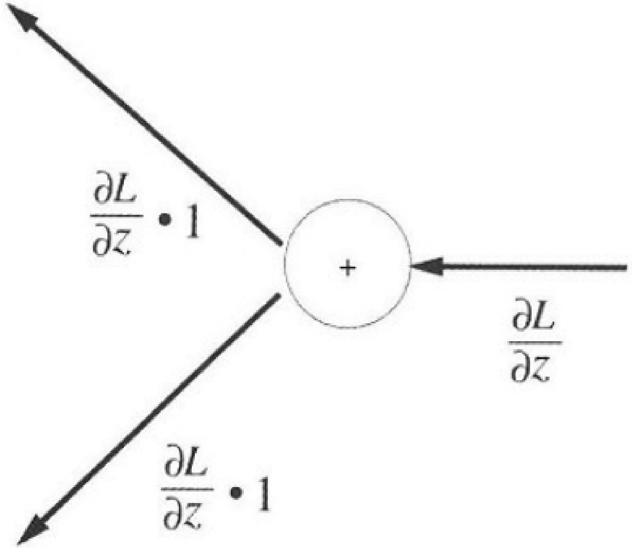
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial t}\frac{\partial t}{\partial x}$$

Part 2

Back Propagation

How to calculate via graph – Node for addition





$$z = x + y$$

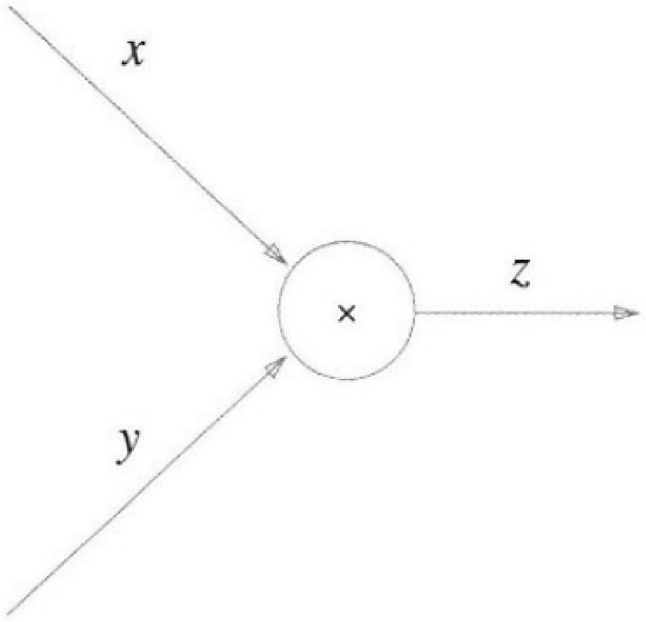
$$\frac{\partial z}{\partial x} = 1 \qquad \frac{\partial z}{\partial y} = 1$$

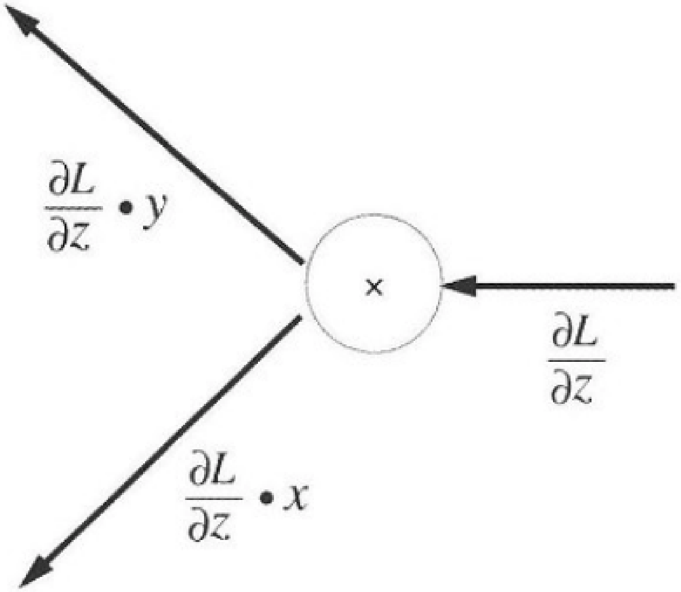
=> 덧셈 노드의 역전파는 입력값을 그대로 흘려넣는다!

Part 2

Back Propagation

How to calculate via graph – Node for multiplication





$$z = x * y$$

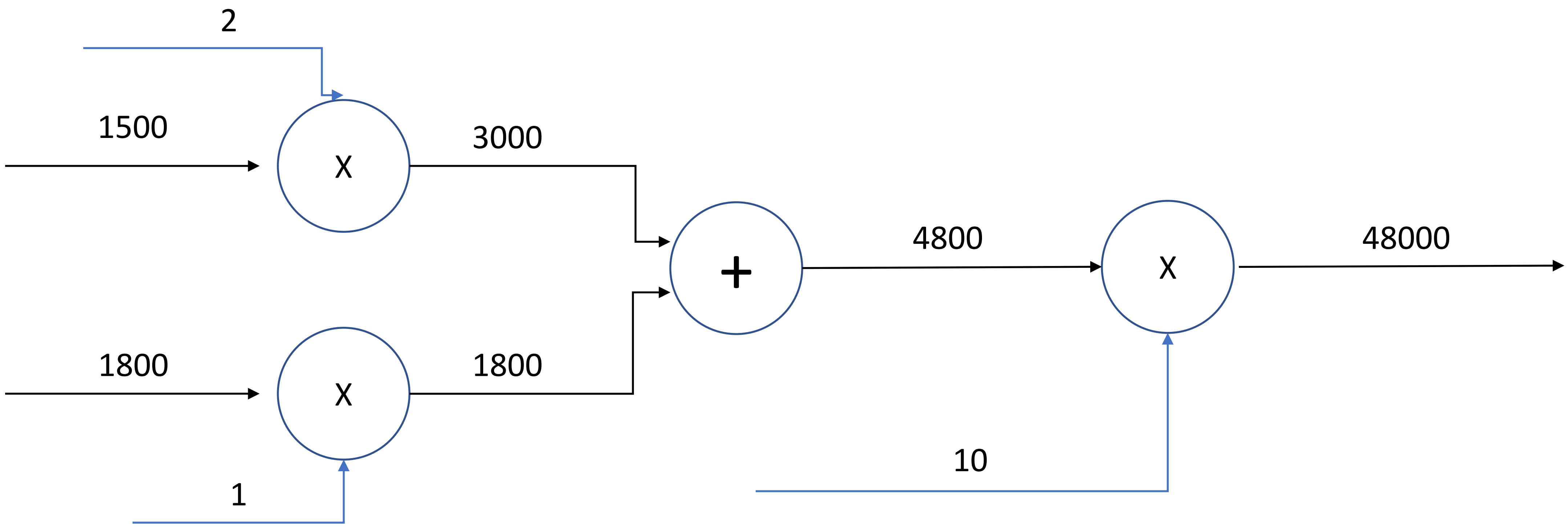
$$\frac{\partial z}{\partial x} = y \qquad \frac{\partial z}{\partial y} = x$$

=> 곱셈노드의 역전파는 상류값에 순전파 때의 입력신호를 '서로 바꾼 값'을 곱하여 하류로 보낸다!

Part 2

Back Propagation

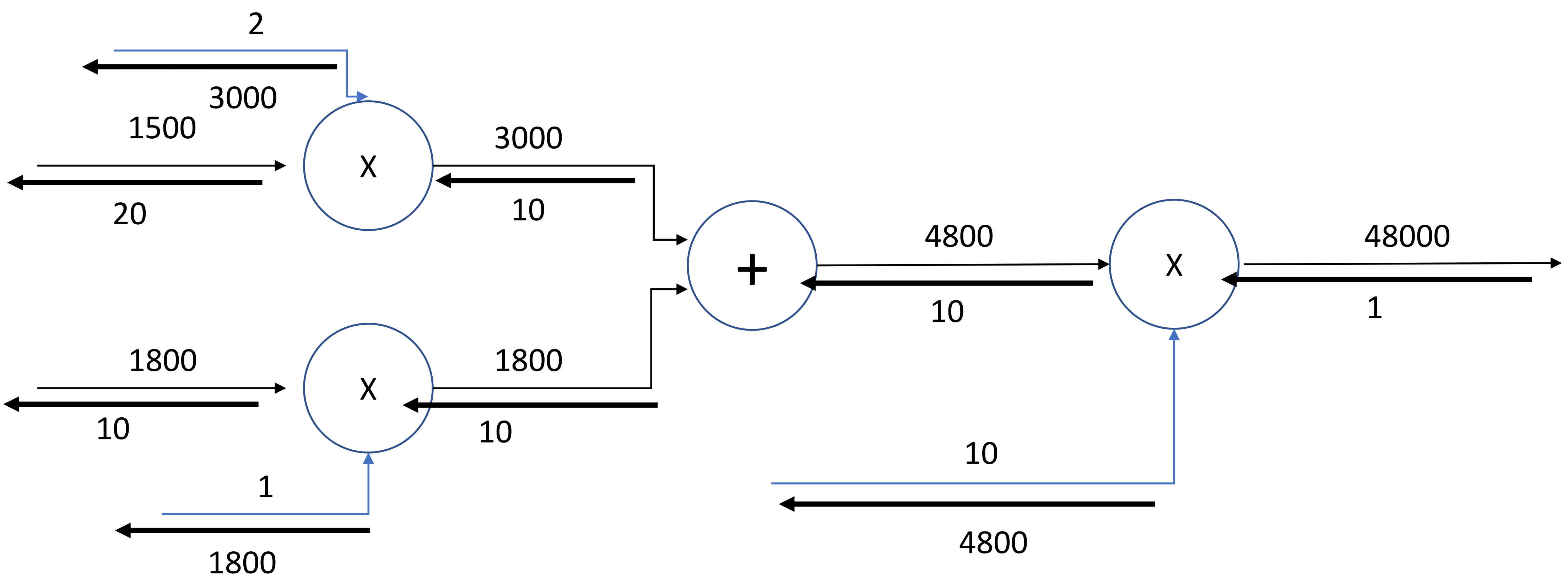
How to calculate via graph



Part 2

Back Propagation

How to calculate via graph



Part 2

Back Propagation

How to calculate via graph

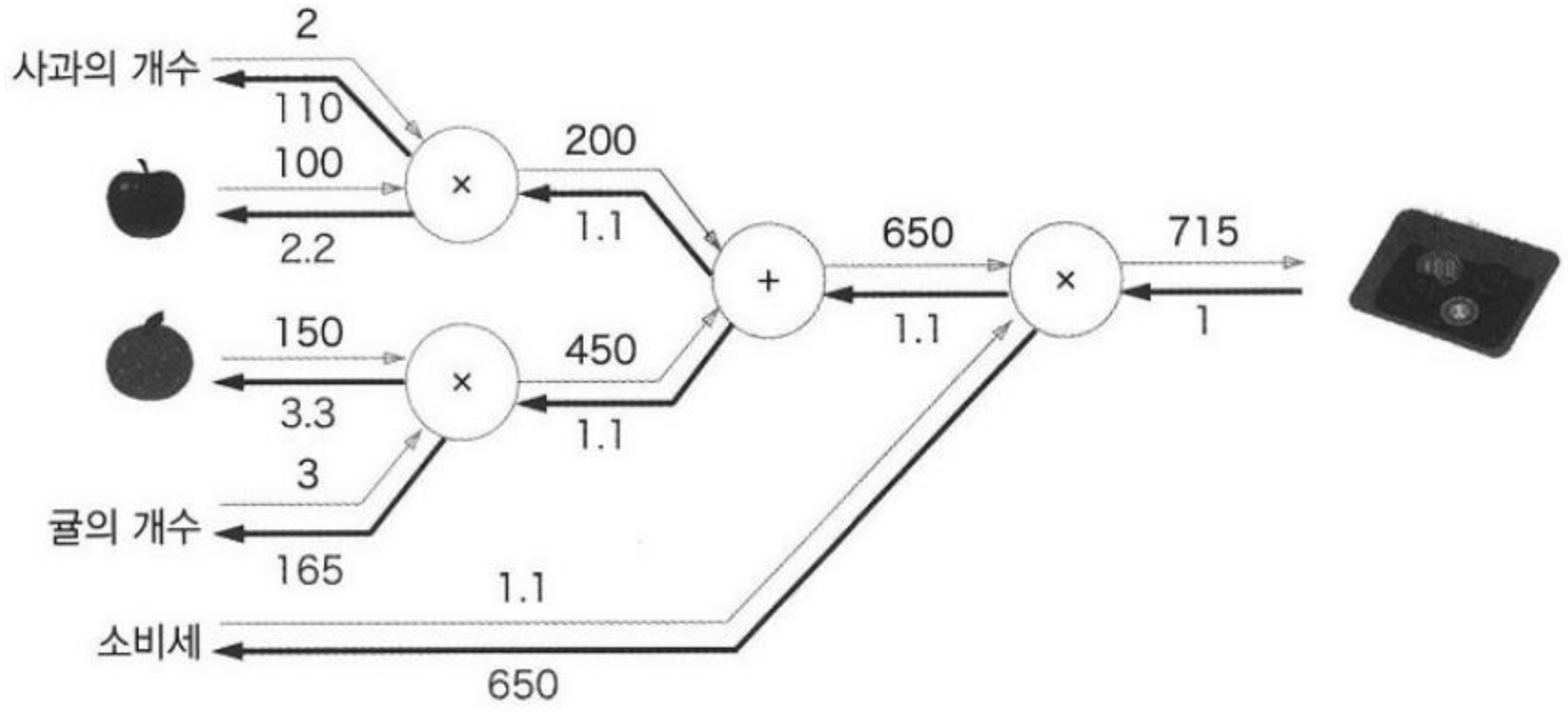
```
class MulLayer:
    def __init__(self):
        self.x = None
        self.y = None

    def forward(self, x, y):
        self.x = x
        self.y = y
        out = x * y

        return out

    def backward(self, dout):
        dx = dout * self.y # x와 y를 바꾼다.
        dy = dout * self.x

        return dx, dy
```

```
class AddLayer:
    def __init__(self):
        pass

    def forward(self, x, y):
        out = x + y
        return out

    def backward(self, dout):
        dx = dout * 1
        dy = dout * 1
        return dx, dy
```

Part 2

Back Propagation

How to calculate via graph

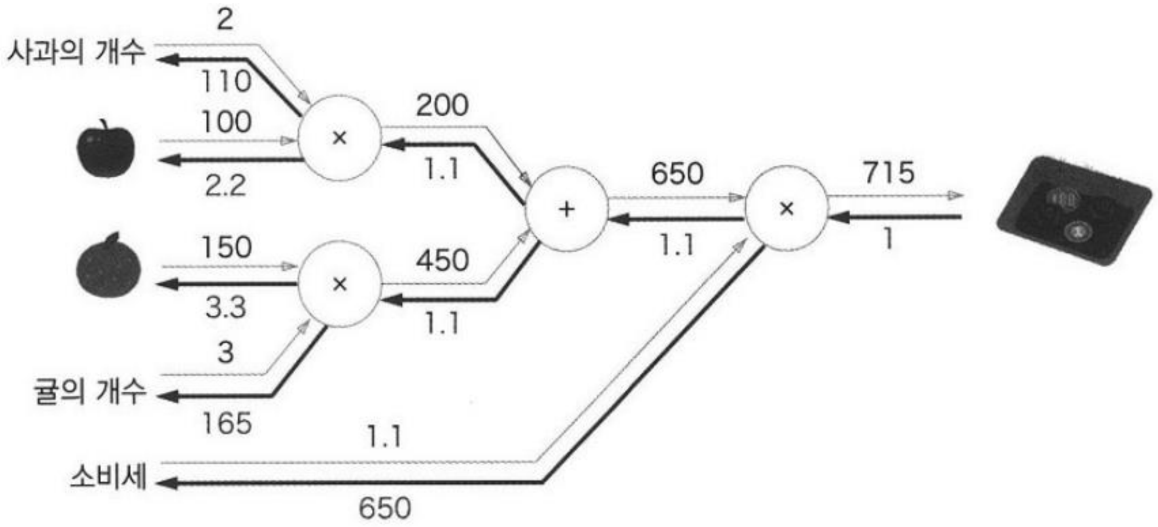


Part 2

Back Propagation

How to calculate via graph

```
apple = 100
apple_num = 2
orange = 150
orange_num = 3
tax = 1.1

# 계층들
mul_apple_layer = MulLayer()
mul_orange_layer = MulLayer()
add_apple_orange_layer = AddLayer()
mul_tax_layer = MulLayer()

# 순전파
apple_price = mul_apple_layer.forward(apple, apple_num) #(1)
orange_price = mul_orange_layer.forward(orange, orange_num) #(2)
all_price = add_apple_orange_layer.forward(apple_price, orange_price) #(3)
price = mul_tax_layer.forward(all_price, tax) #(4)
```

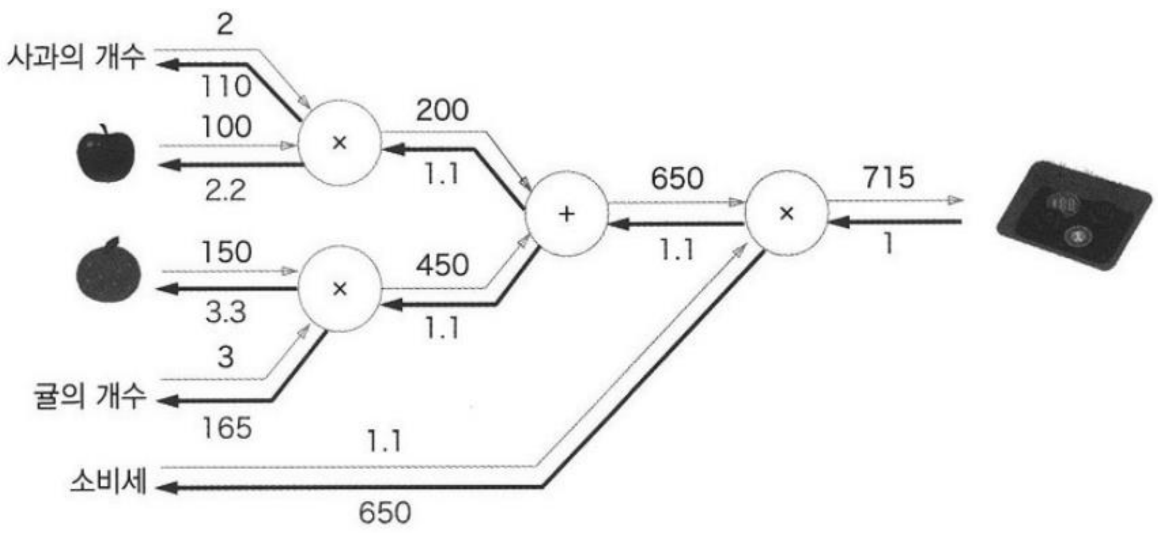


Part 2

Back Propagation

How to calculate via graph

```
apple = 100
apple_num = 2
orange = 150
orange_num = 3
tax = 1.1

# 계층들
mul_apple_layer = MulLayer()
mul_orange_layer = MulLayer()
add_apple_orange_layer = AddLayer()
mul_tax_layer = MulLayer()

# 역전파
dprice = 1
dall_price, dtax = mul_tax_layer.backward(dprice) #(4)
dapple_price, dorange_price = add_apple_orange_layer.backward(dall_price) #(3)
dorange, dorange_num = mul_orange_layer.backward(dorange_price) #(2)
dapple, dapple_num = mul_apple_layer.backward(dapple_price) #(1)

print(price) # 715
print(dapple_num, dapple, dorange, dorange_num, dtax) # 110 2.2 3.3 165 650
```



Part 3 레이어를 계산 그래프로 표기하기

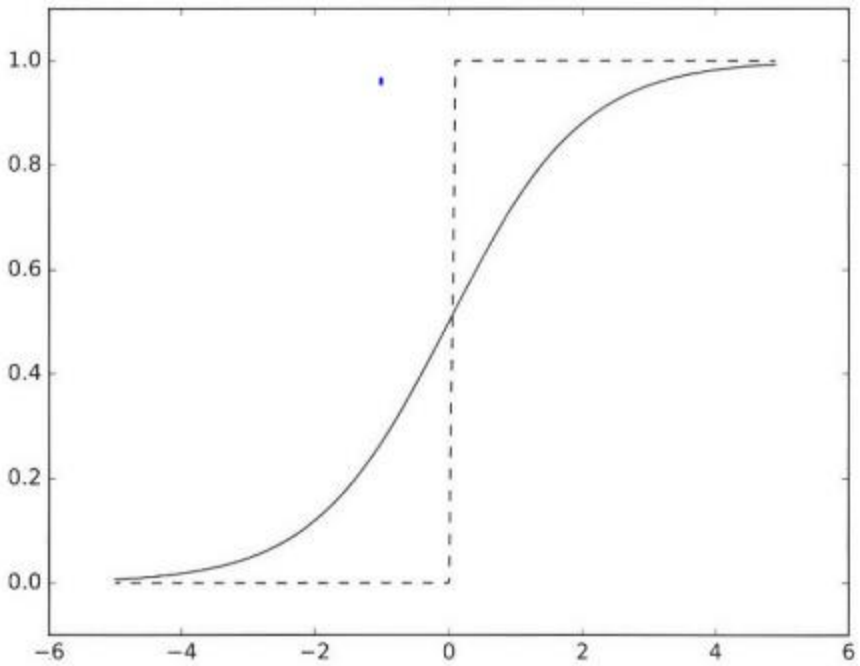
Part 3

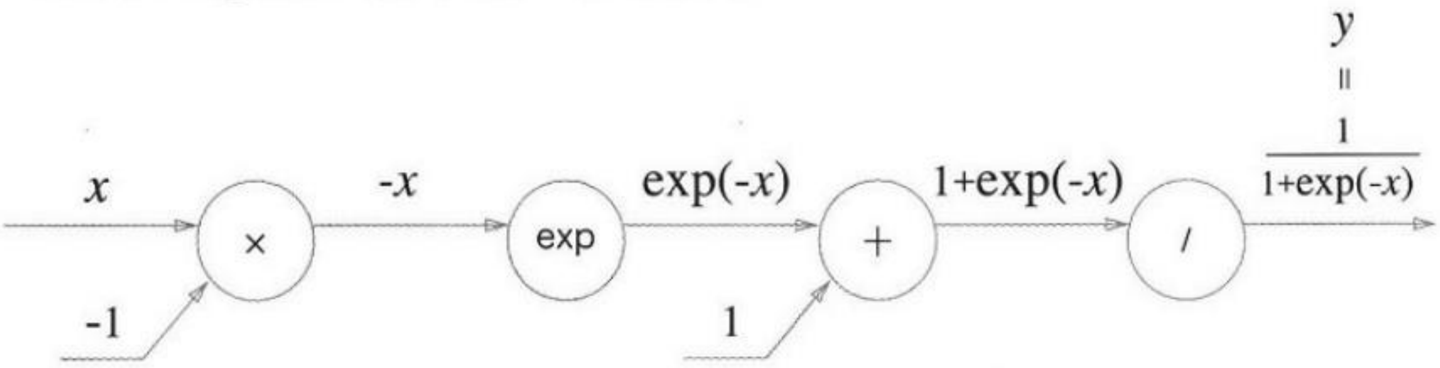
Back Propagation

How to represent layer via calculation graph

시그모이드 함수

$$y = \frac{1}{1 + \exp(-x)}$$



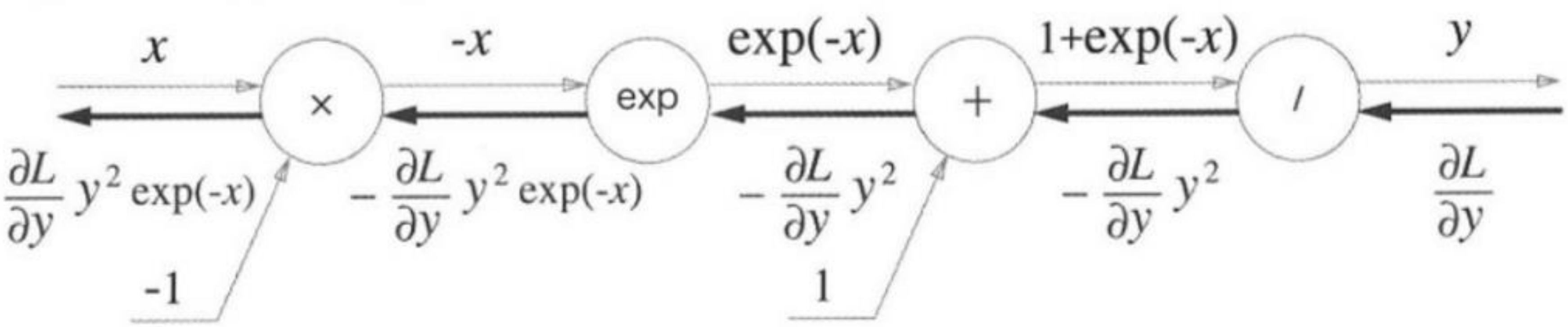


Part 3

Back Propagation

How to represent layer via calculation graph

$y = \exp(x)$ $y = 1/x$



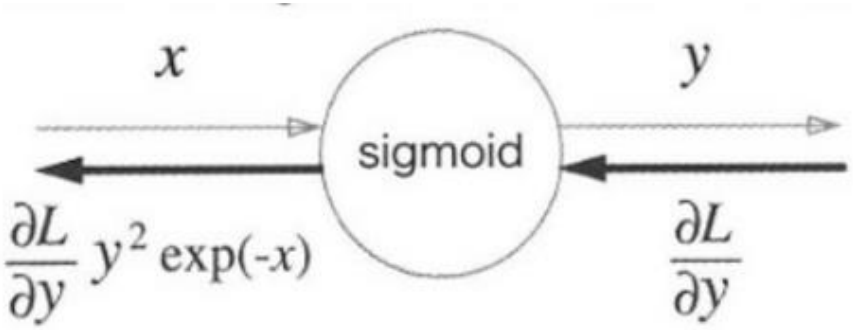
$$\frac{\partial y}{\partial x} = \exp(x)$$

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

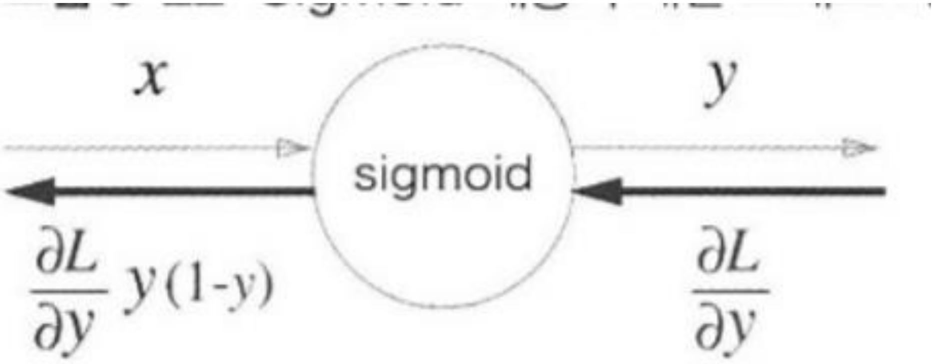
Part 3

Back Propagation

How to represent layer via calculation graph



$$\frac{\partial L}{\partial y} y^2 \exp(-x) = \frac{\partial L}{\partial y} \frac{1}{(1+\exp(-x))^2} \exp(-x)$$
$$= \frac{\partial L}{\partial y} \frac{1}{1+\exp(-x)} \frac{\exp(-x)}{1+\exp(-x)}$$
$$= \frac{\partial L}{\partial y} y(1-y)$$

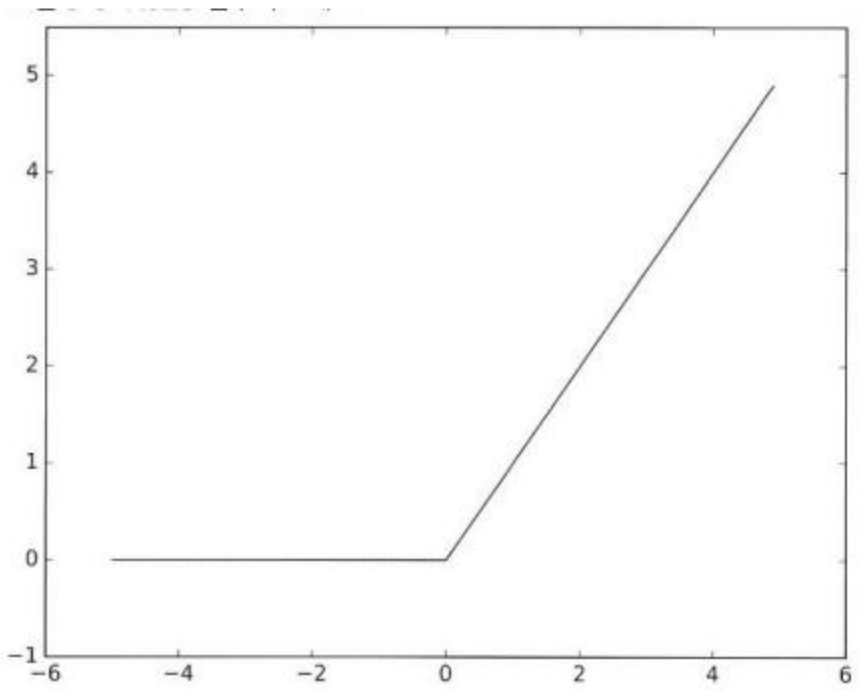


Back Propagation

How to represent layer via calculation graph

ReLu 함수

$$y=\begin{cases} x & (x>0) \\ 0 & (x\le 0) \end{cases}$$



Part 3

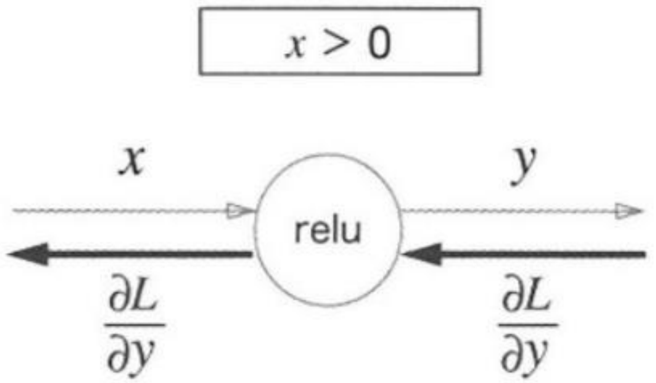
Back Propagation

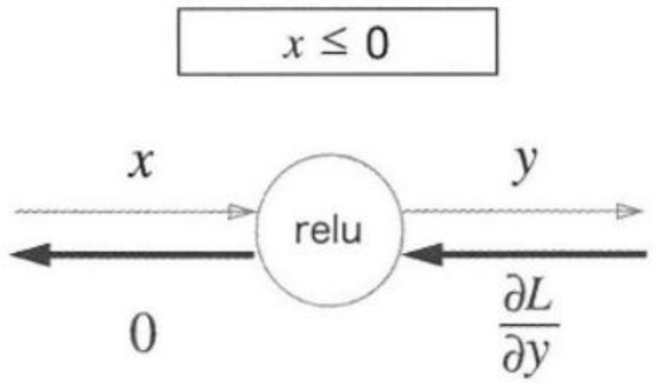
How to represent layer via calculation graph

ReLu 함수

$$y=\begin{cases} x & (x>0) \\ 0 & (x\le 0) \end{cases}$$

$$\frac{\partial y}{\partial x}=\begin{cases} 1 & (x>0) \\ 0 & (x\le 0) \end{cases}$$





Part 3

Back Propagation

How to represent layer via calculation graph

```
class Relu:
    def __init__(self):
        self.mask = None

    def forward(self, x):
        self.mask = (x <= 0)
        out = x.copy()
        out[self.mask] = 0

        return out

    def backward(self, dout):
        dout[self.mask] = 0
        dx = dout

        return dx
```

Mask :True/False로 구성된 Numpy 배열

입력값 ≤ 0 -> True

입력값 > 0 -> False

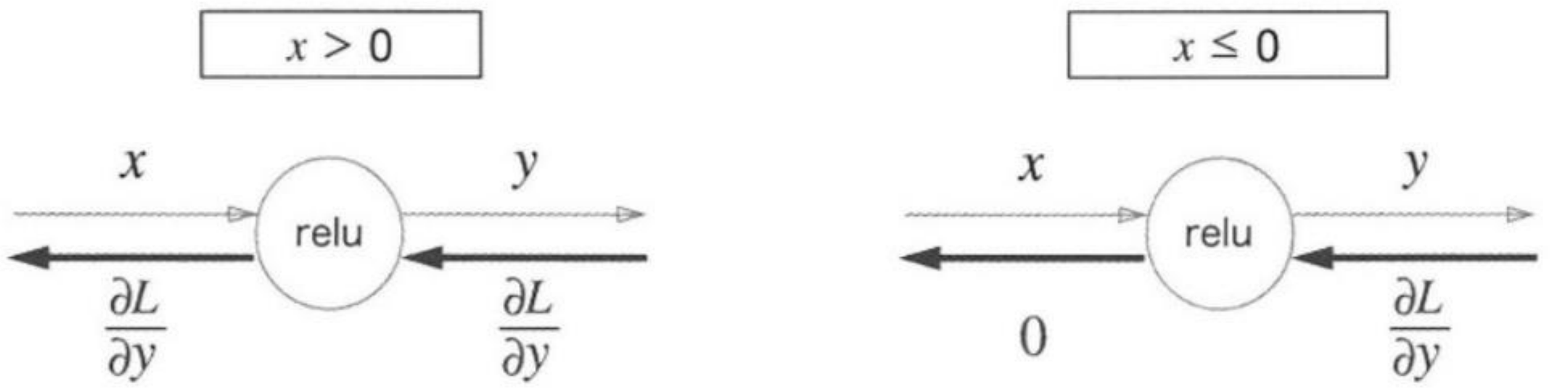
$$\begin{pmatrix} 0.5 & -0.1 \\ 0.4 & -0.2 \end{pmatrix} \Rightarrow \begin{pmatrix} False & True \\ False & True \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} 0.5 & 0 \\ 0.4 & 0 \end{pmatrix}$$

Part 3

Back Propagation

How to represent layer via calculation graph



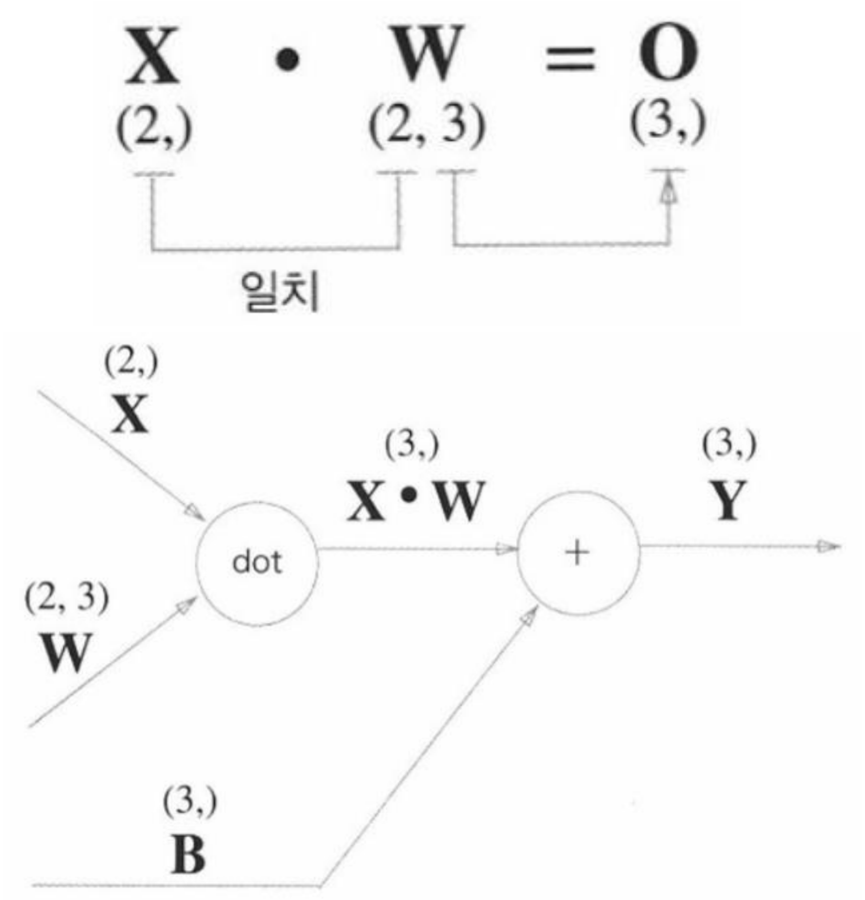
순전파 과정에서 출력이 제대로 이루어진다면 역전파시에도 그대로 출력된다.
그러나 값이 제대로 출력될 수 없다면 역전파 시 하류로 신호를 보내지 않는다.

Part 3

Back Propagation

How to represent layer via calculation graph

Affine 계층



행렬의 대응하는 원소 수가 일치하여야 한다.

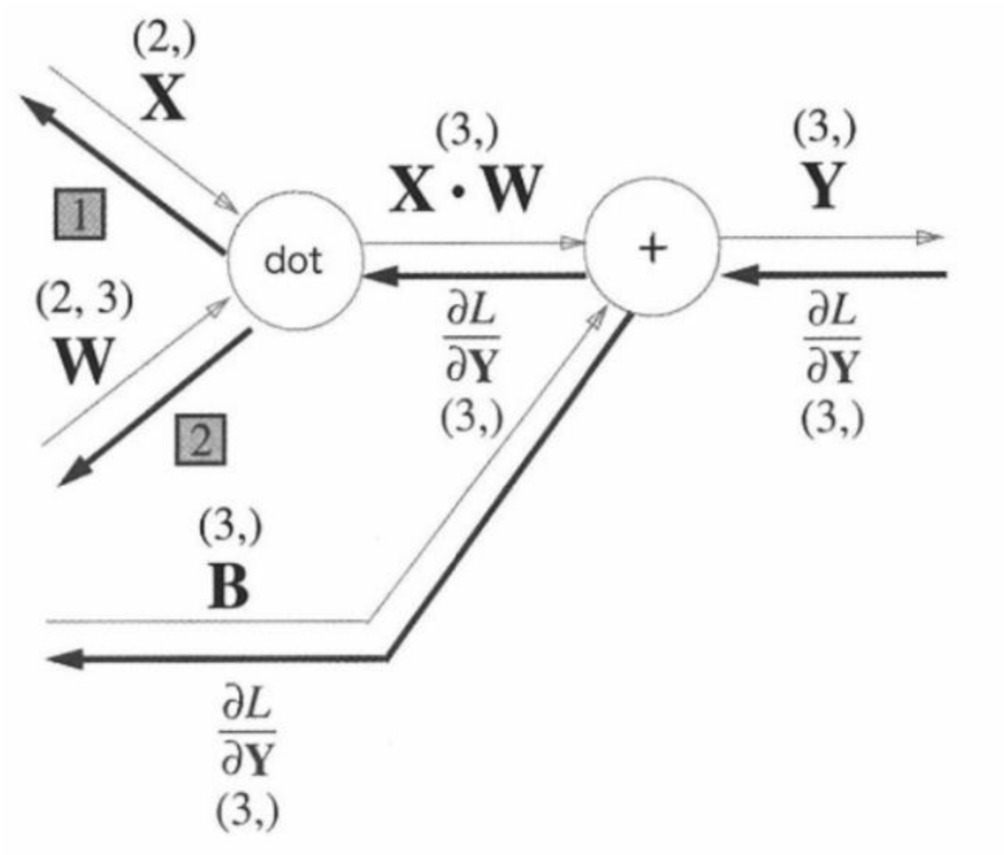
Y = X*W + B

'dot' 노드는 행렬의 대응하는 원소 수가
일치하도록 곱을 조립하는 기능 수행

Part 3

Back Propagation

How to represent layer via calculation graph



$$\frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{Y}} \cdot \mathbf{W}^{\mathrm{T}}$$

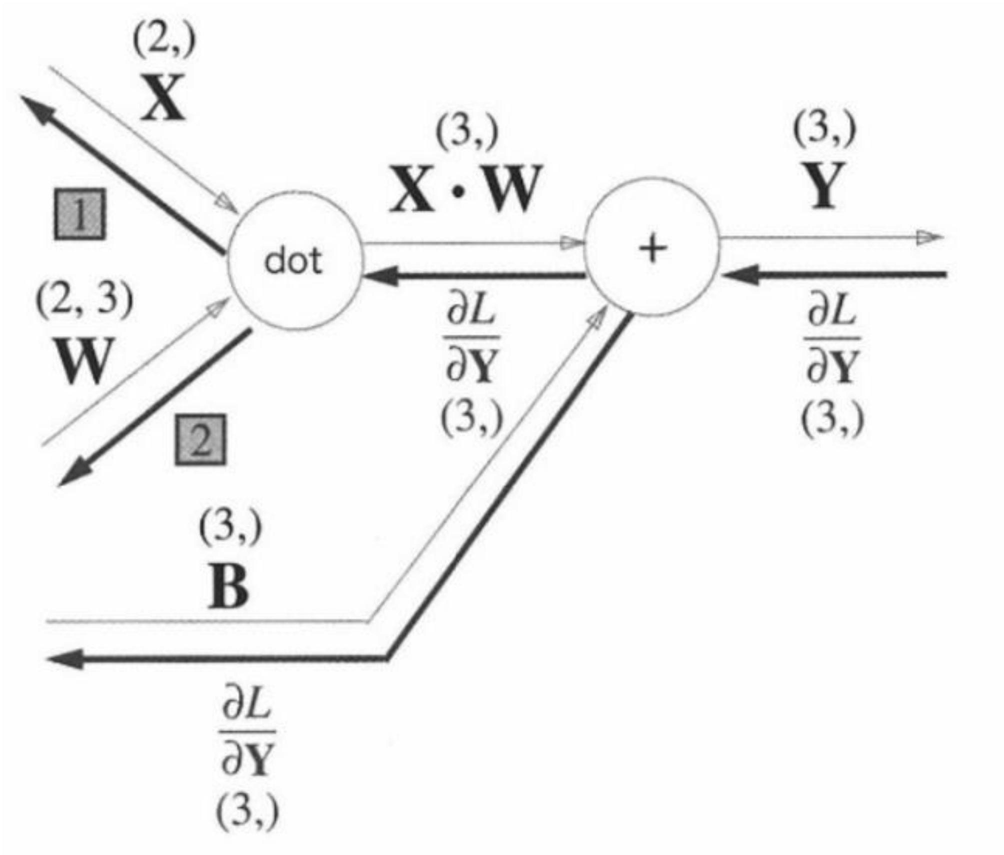
$$\frac{\partial L}{\partial \mathbf{W}} = \mathbf{X}^{\mathrm{T}} \cdot \frac{\partial L}{\partial \mathbf{Y}}$$

$$\mathbf{W} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{pmatrix} \qquad \mathbf{W}^{\mathrm{T}} = \begin{pmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \\ w_{13} & w_{23} \end{pmatrix}$$

Part 3

Back Propagation

How to represent layer via calculation graph



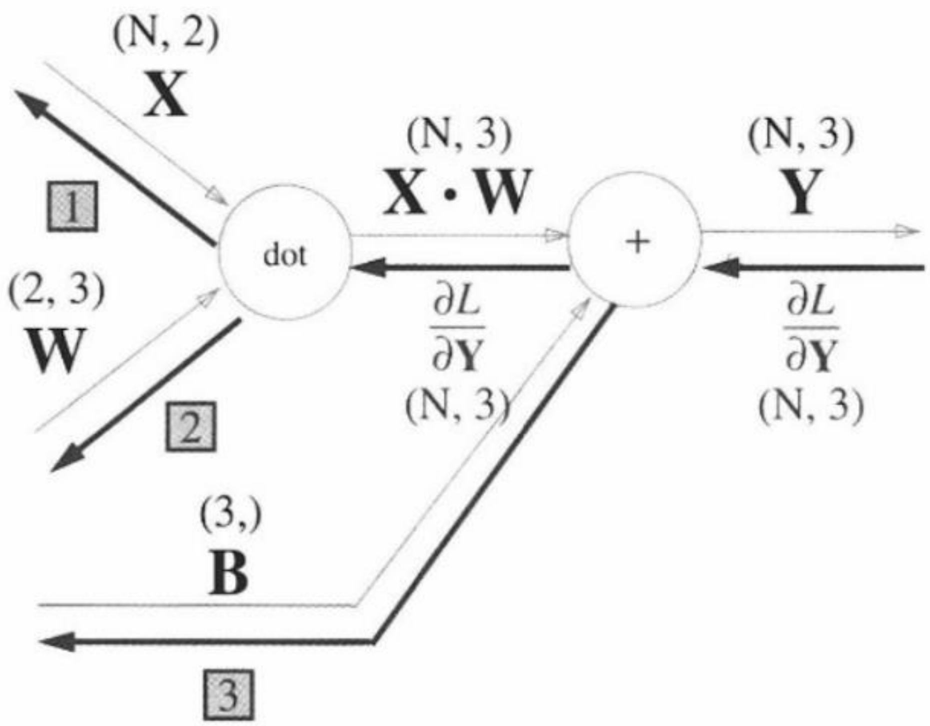
1 $$\underset{(2,)}{\frac{\partial L}{\partial \mathbf{X}}} = \underset{(3,)}{\frac{\partial L}{\partial \mathbf{Y}}} \cdot \underset{(3,\ 2)}{\mathbf{W}^{\mathrm{T}}}$$

2 $$\underset{(2,\ 3)}{\frac{\partial L}{\partial \mathbf{W}}} = \underset{(2,\ 1)}{\mathbf{X}^{\mathrm{T}}} \cdot \underset{(1,\ 3)}{\frac{\partial L}{\partial \mathbf{Y}}}$$

Back Propagation

How to represent layer via calculation graph

배치용 Affine 계층



1 $$\underset{(N,\,2)}{\frac{\partial L}{\partial \mathbf{X}}} = \underset{(N,\,3)}{\frac{\partial L}{\partial \mathbf{Y}}} \cdot \underset{(3,\,2)}{\mathbf{W}^{\mathrm{T}}}$$

2 $$\underset{(2,\,3)}{\frac{\partial L}{\partial \mathbf{W}}} = \underset{(2,\,N)}{\mathbf{X}^{\mathrm{T}}} \cdot \underset{(N,\,3)}{\frac{\partial L}{\partial \mathbf{Y}}}$$

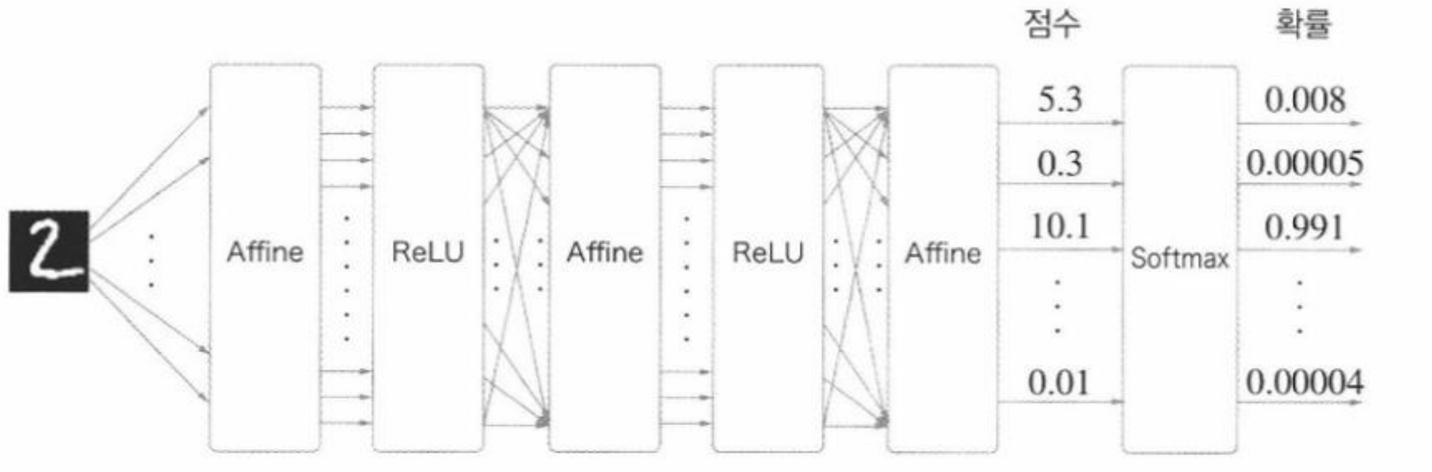
3 $$\underset{(3)}{\frac{\partial L}{\partial \mathbf{B}}} = \underset{(N,\,3)}{\frac{\partial L}{\partial \mathbf{Y}}}$$ 의 각 열의 합

Part 3

Back Propagation

How to represent layer via calculation graph

Softmax with Loss 계층



$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n} \exp(a_i)}$$

소프트맥스 함수; 다중분류의 시그모이드; 출력

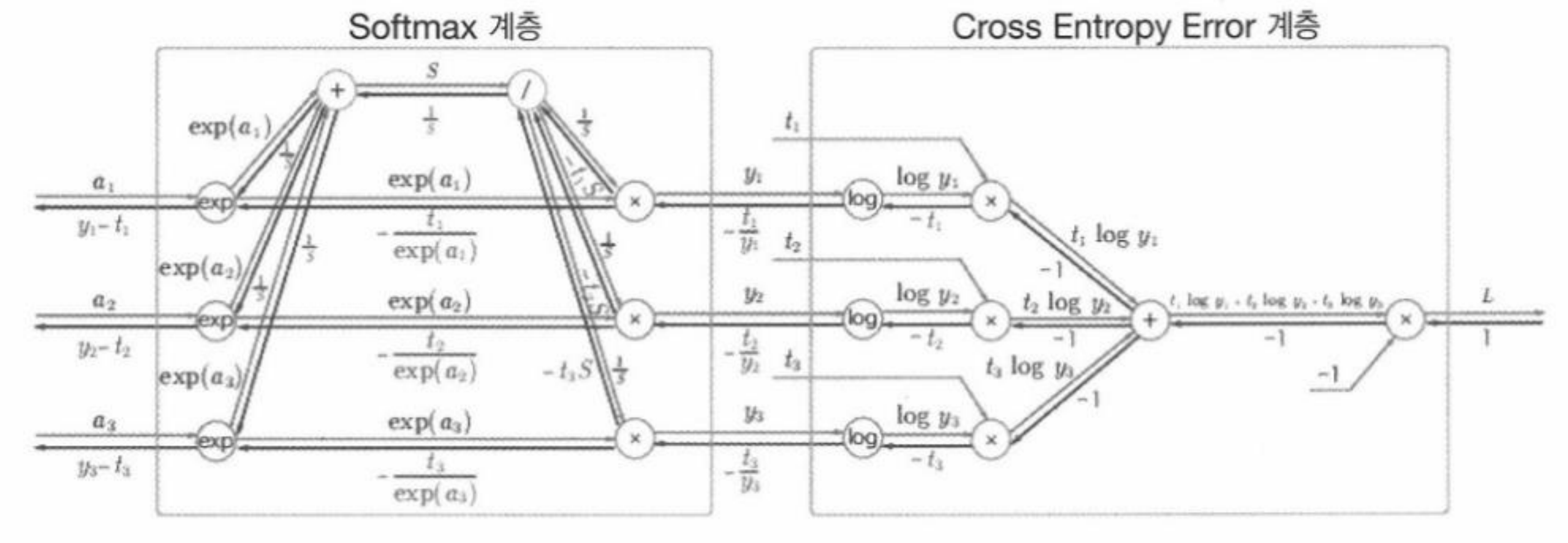
$$E = -\sum_k t_k \log y_k$$

교차 엔트로피 함수; 손실함수; 최적 매개변수 탐색 도구

Part 3

Back Propagation

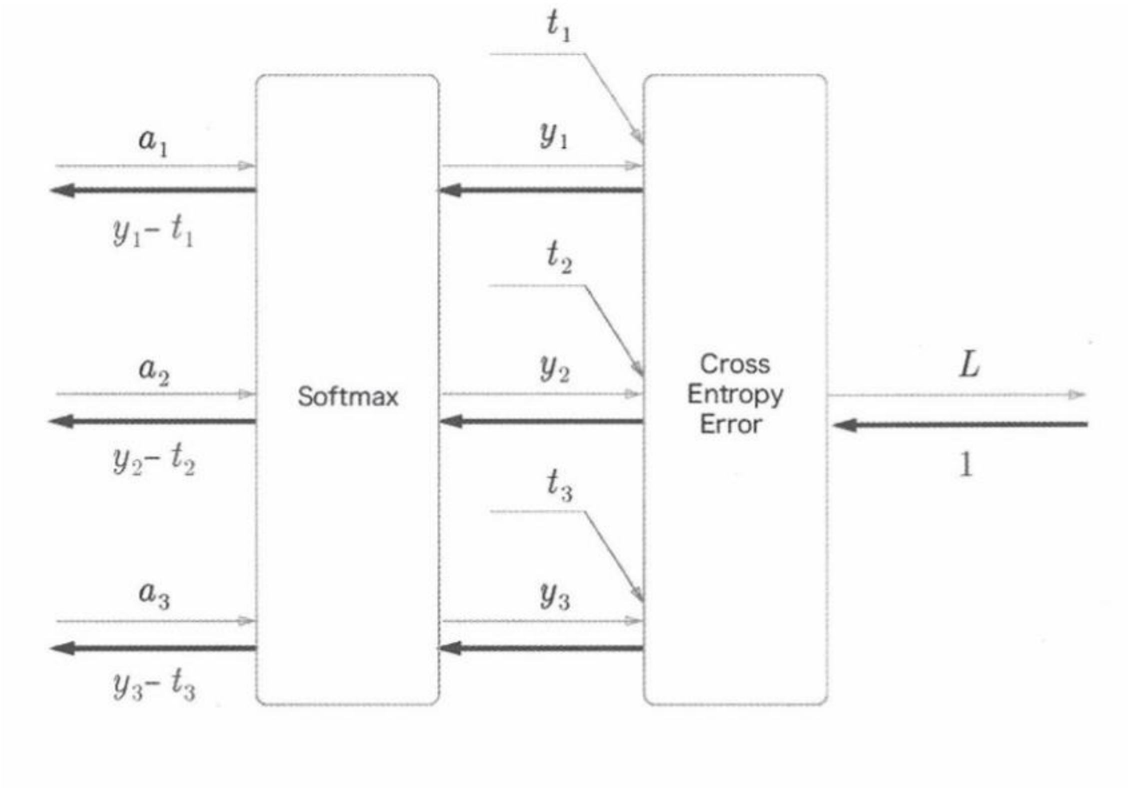
How to represent layer via calculation graph



Part 3

Back Propagation

How to represent layer via calculation graph



역전파 최종 도출값이 매우 간명하게 나타남

$(y_i - t_i)$
⇒ SoftMax 계층의 출력과 정답레이블의 차이

Ex) 정답 레이블 : (0, 1, 0)
⇒ Softmax 출력 : (0.3, 0.2, 0.5)
⇒ 역전파 결과 : (0.3, -0.8, 0.5)
⇔ Softmax 출력 : (0.01, 0.99, 0)
⇒ 역전파 결과 : (0.01, -0.01, 0)

⇒ 역전파 결과(오차)가 낮으므로 학습 정도도 약해질 것!

Part 3

Back Propagation

How to represent layer via calculation graph

- 수치 미분의 용도 => 기울기 확인(gradient check)
- 구현하기 쉬운 수치미분의 결과와 오차역전파법의 결과를 비교하여 오차역전파법이 제대로 구현되었는지 확인하는 과정

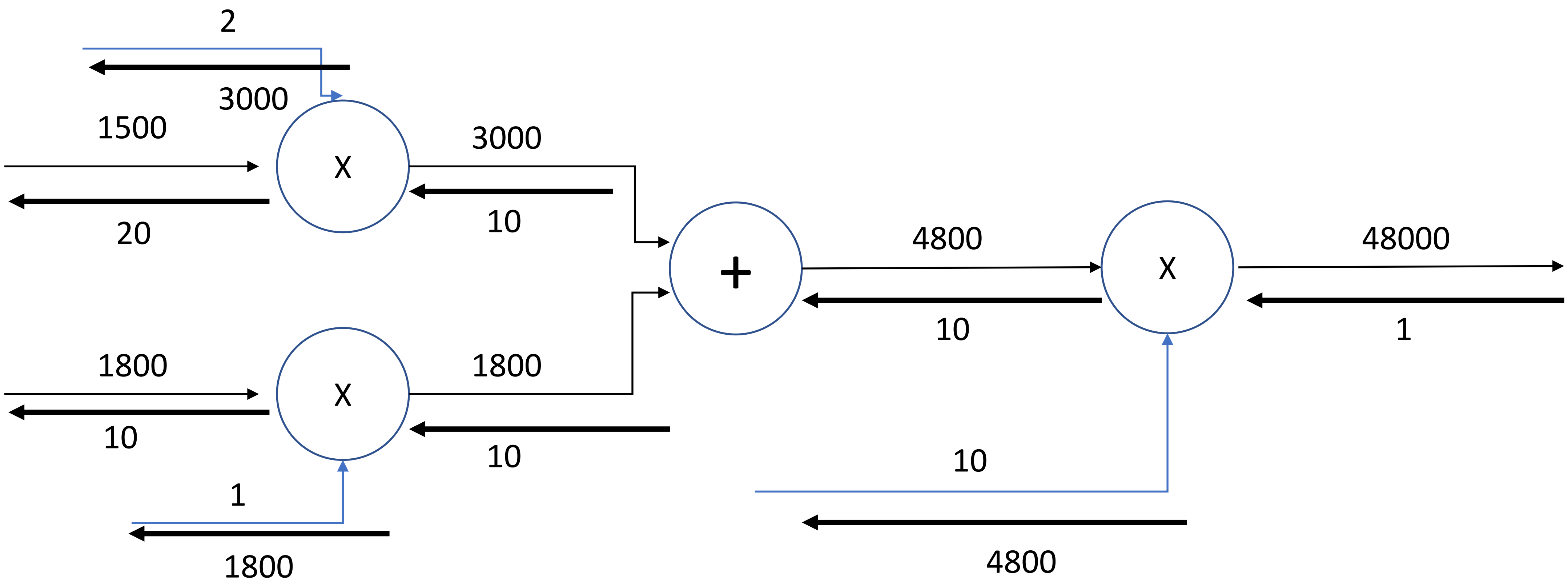
=> 실습!

Part 3

Back Propagation

How to represent layer via calculation graph

과제 : 코드로 아래 계산그래프 역전파 구하기



SUMMARY

- 계산 그래프를 통한 계산과정
- 오차 역전파법을 활용한 효율적인 기울기 계산법
- 2가지 기울기 계산법을 활용한 산출한 기울기 정확도 검증


감사합니다!