Section 6. 경사 하강 심화 이론 (Advanced Topics in Gradient Descent)

ACADENTIAL

목차

ACADENTIAL

Recap from Section 5.
Gradient Descent의 기본 개념

ACADENTIAL

Recap

Section 5. Gradient Descent의 기본 개념

Gradient Descent

역할: **손실 함수의 값이 최소화**하도록 **모델의 weight을 최적화하는 것**

원리:

- **경사는 손실함수가 증가하는 방향을 향한다.**
- **따라서 경사하강은 경사의 음의 방향**으로 모델의 weight을 update해주는 것이다!

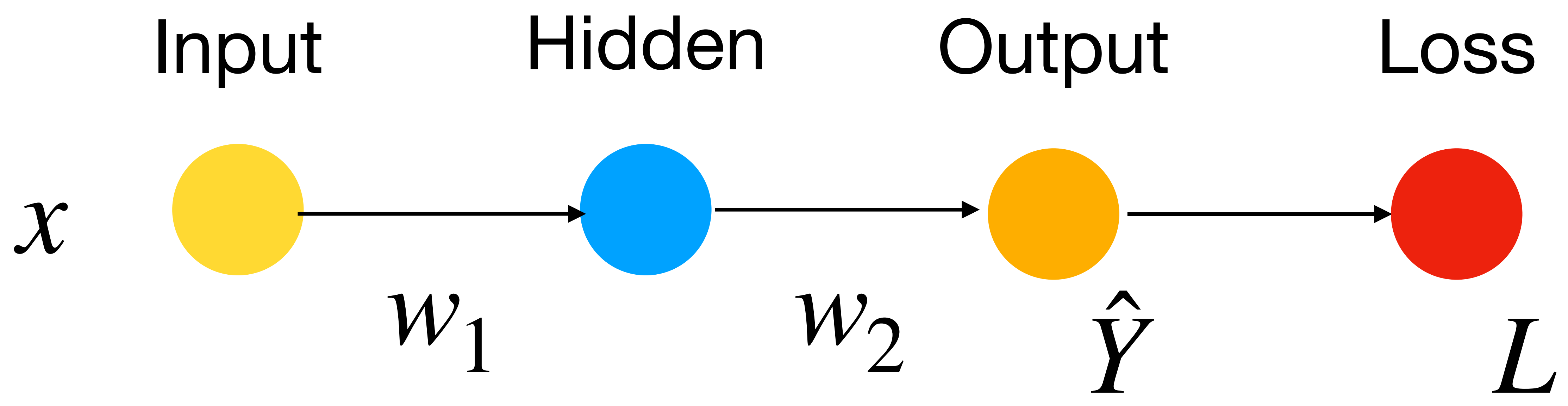
$$w \rightarrow w - \lambda \cdot \frac{dL}{dw}$$

Recap

Section 5. Gradient Descent의 기본 개념

- 하지만 이것은 variable이 하나인 (single variate input)의 간단한 예시였다.

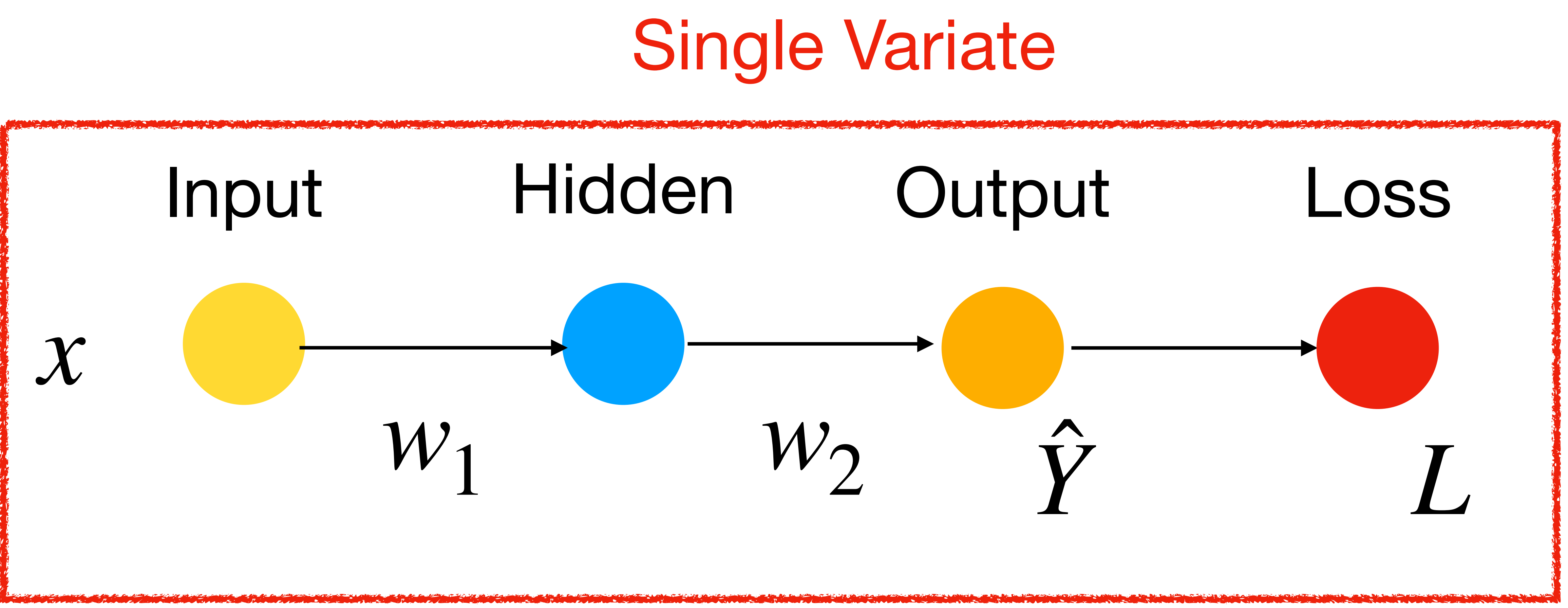
$$w \rightarrow w - \lambda \cdot \frac{dL}{dw}$$

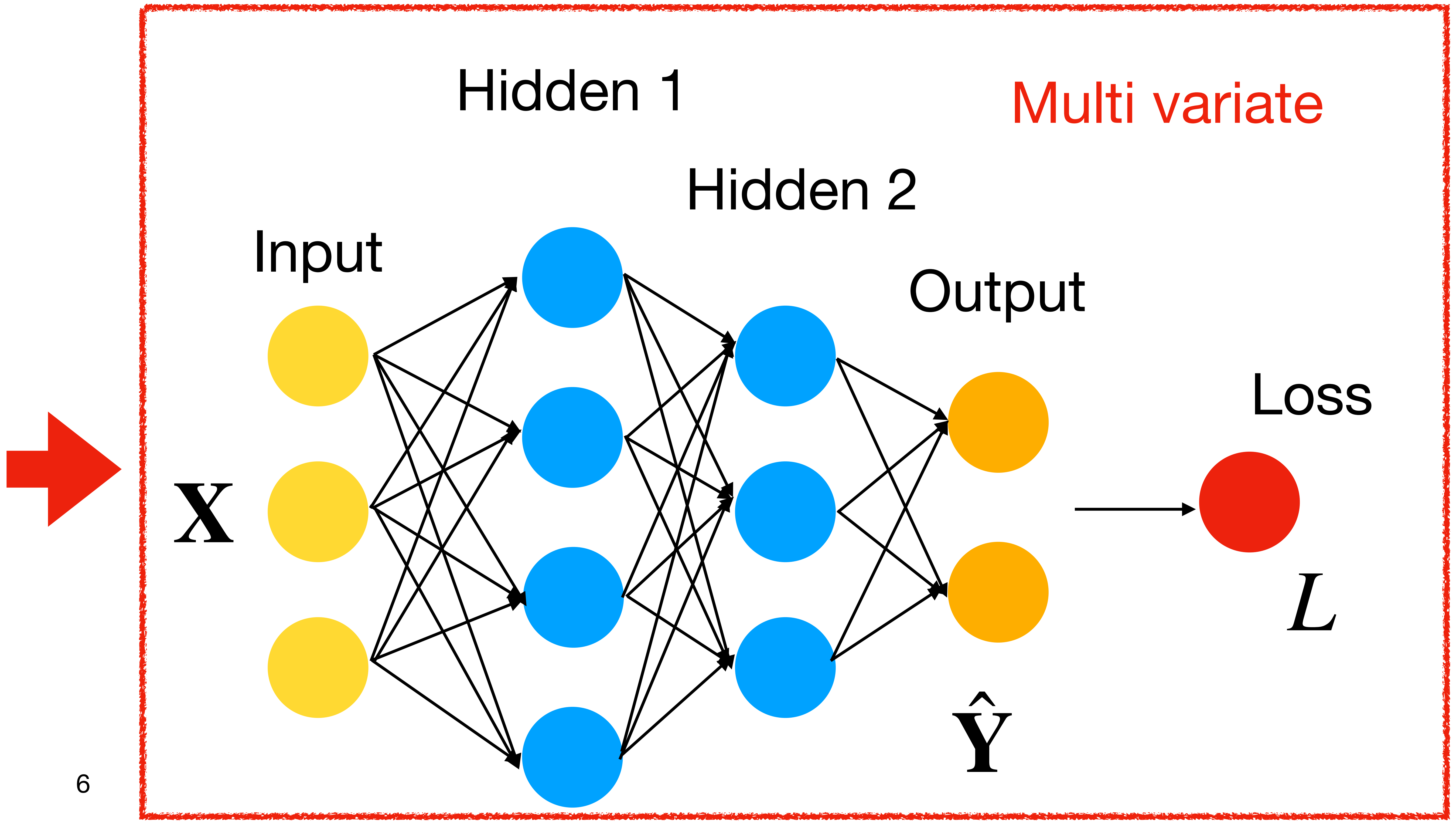


Recap

Extending from single variate to multi-variate

- 하지만 Input feature가 여러 개 (multi-variate)하거나 Hidden layer가 여러개의 neuron들로 구성되어 있으면 어떻게 할 것인가?

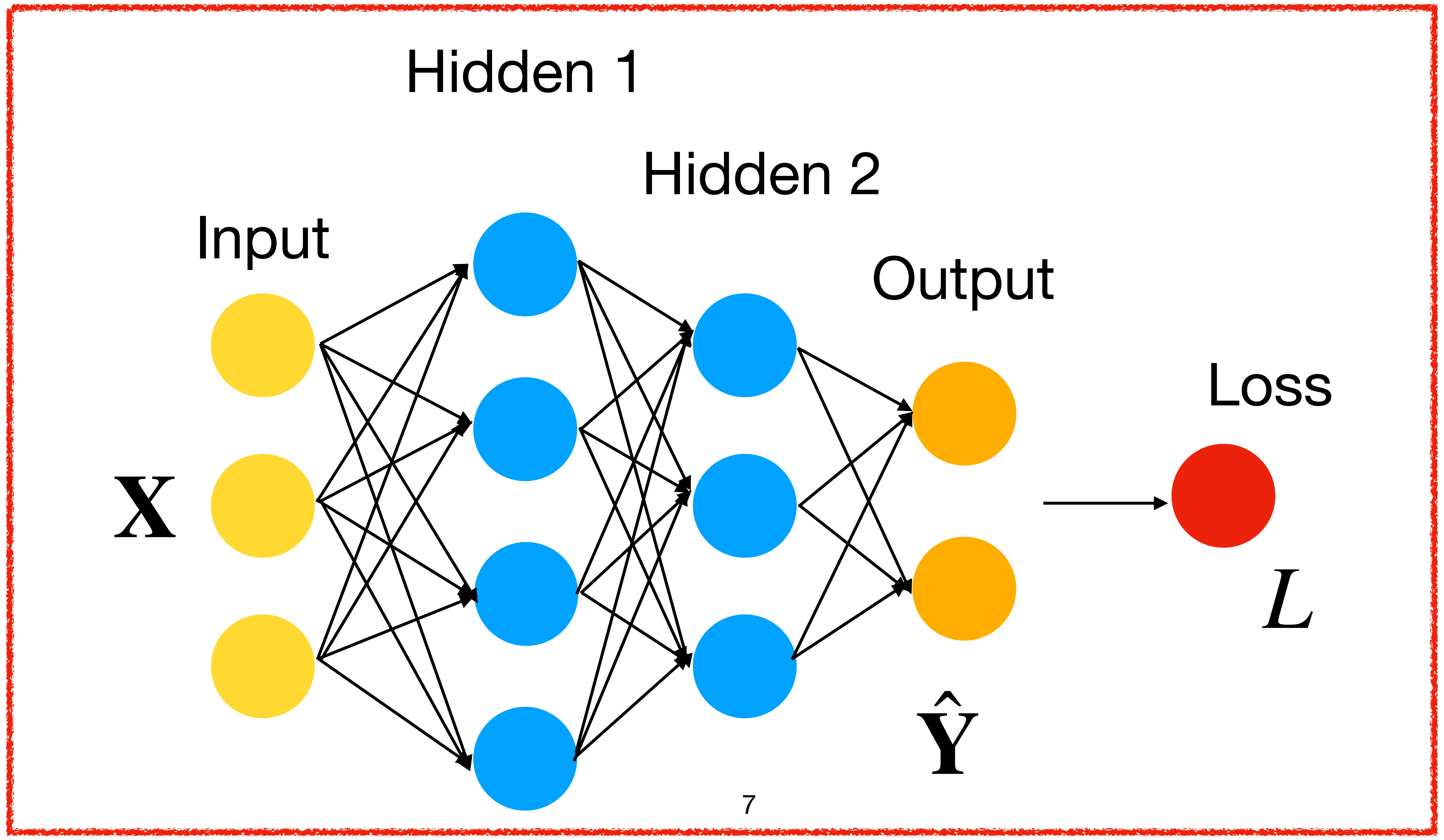




Recap

Extending from single variate to multi-variate

- 이번 시간에는 아래와 같은 Multi-variate의 경우에 대해서 살펴보자!



Multi variate

ACADENTIAL

6-1. Multi-variate Multi neuron Neural Network

ACADENTIAL

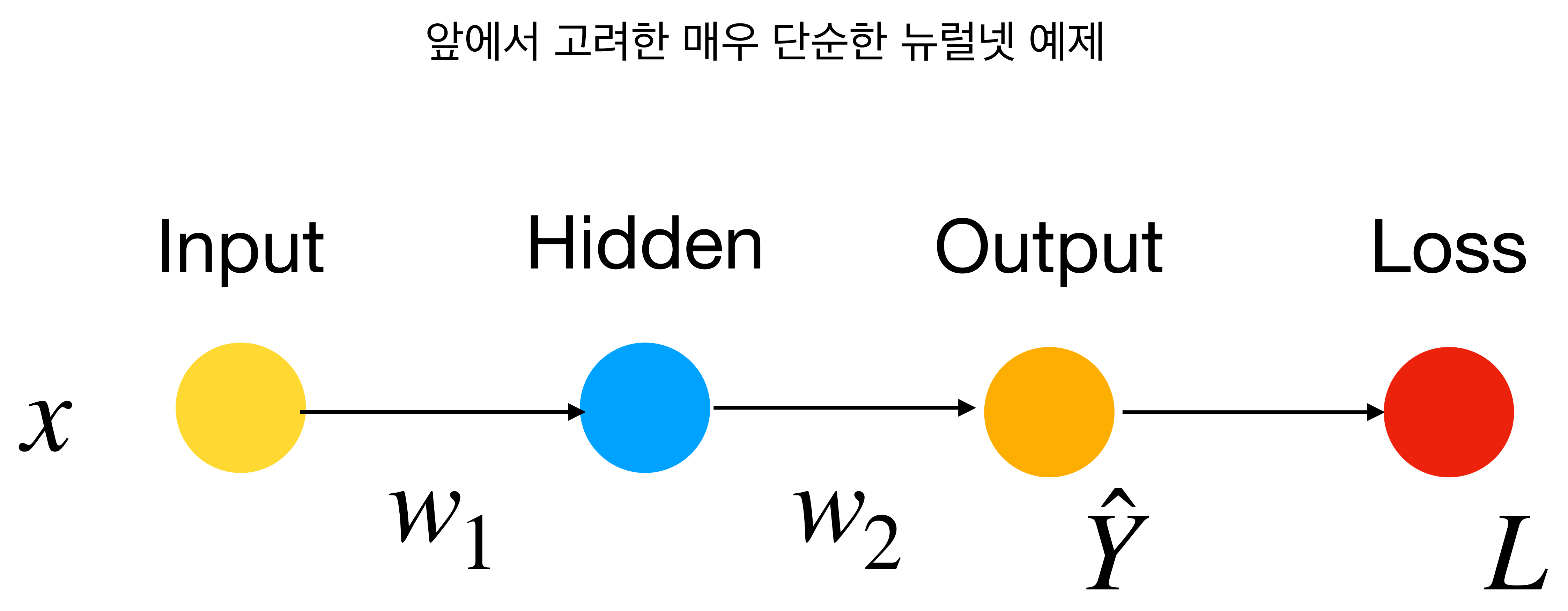
Objective

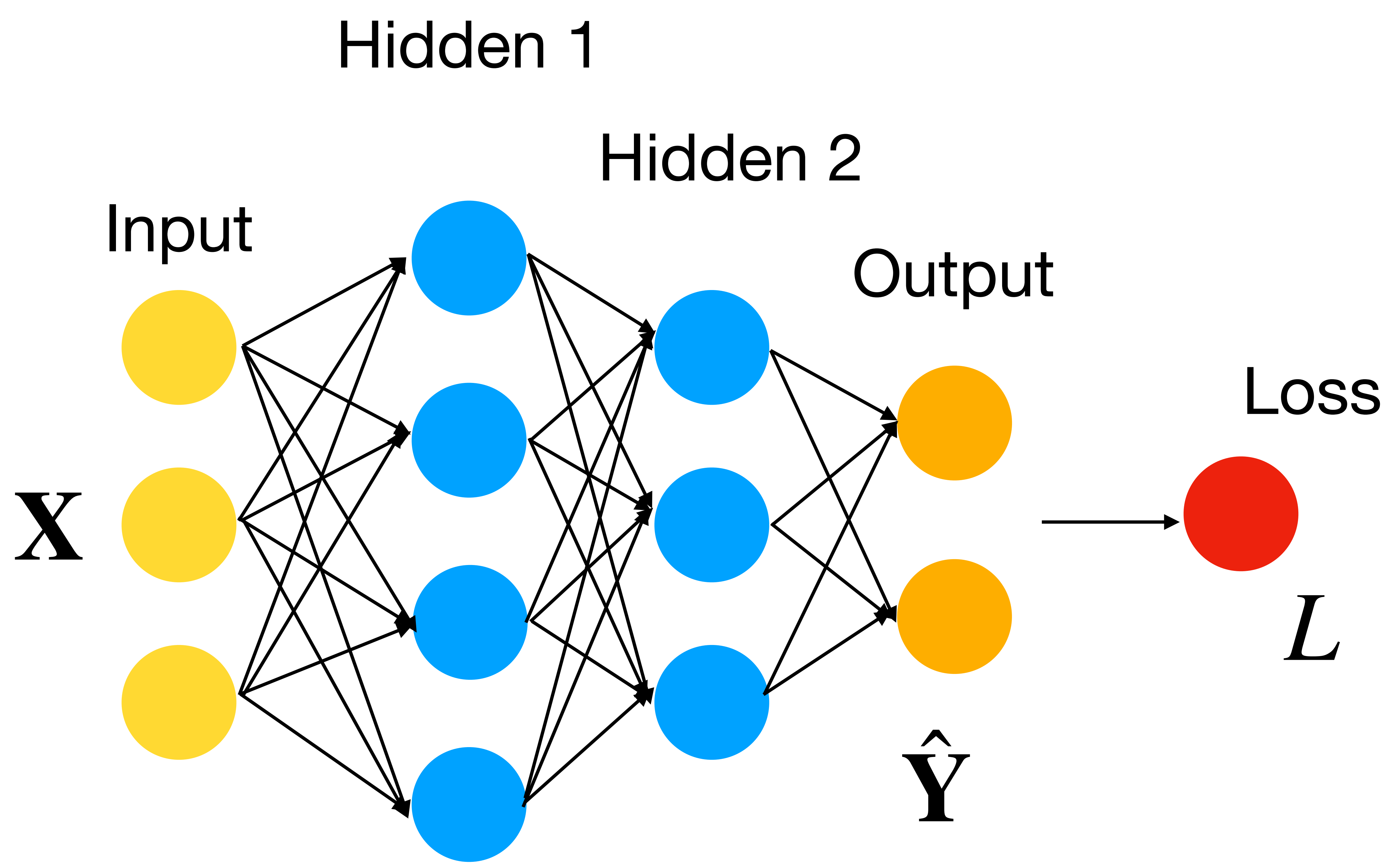
학습 목표

- Scalar input, single neuron으로 구성된 Neural Network을 Multi variate, Multi neuron으로 확장하기
- Neural Network의 forward pass을 행렬의 곱으로 표현하기

Multi-variate Multi-neuron NN

- 여러 개의 뉴론들 $W \in \mathbb{R}^{M \times D}$로 구성된 **Layer들이 여러 개 쌓여**있고,
- **입력값이 다차원의 vector** $\mathbf{x} \in \mathbb{R}^D$





Multi-variate Multi-neuron NN

Multivariate Case

- 예를 들어 **ResNet-50의 경우 23 million (2300만개의 weight parameter)**들로 구성됨.
- 통상적으로 input도 scalar 값이 아니라 다차원의 vector이다. (**multivariate**)
- 어떻게 하면:

$$\boxed{x, w \in \mathbb{R}^1} \rightarrow \boxed{\mathbf{x} \in \mathbb{R}^D, W \in \mathbb{R}^{M \times D}}$$

scalar input
single neuron

Multi-variate
Multi-neuron

ACADENTIAL

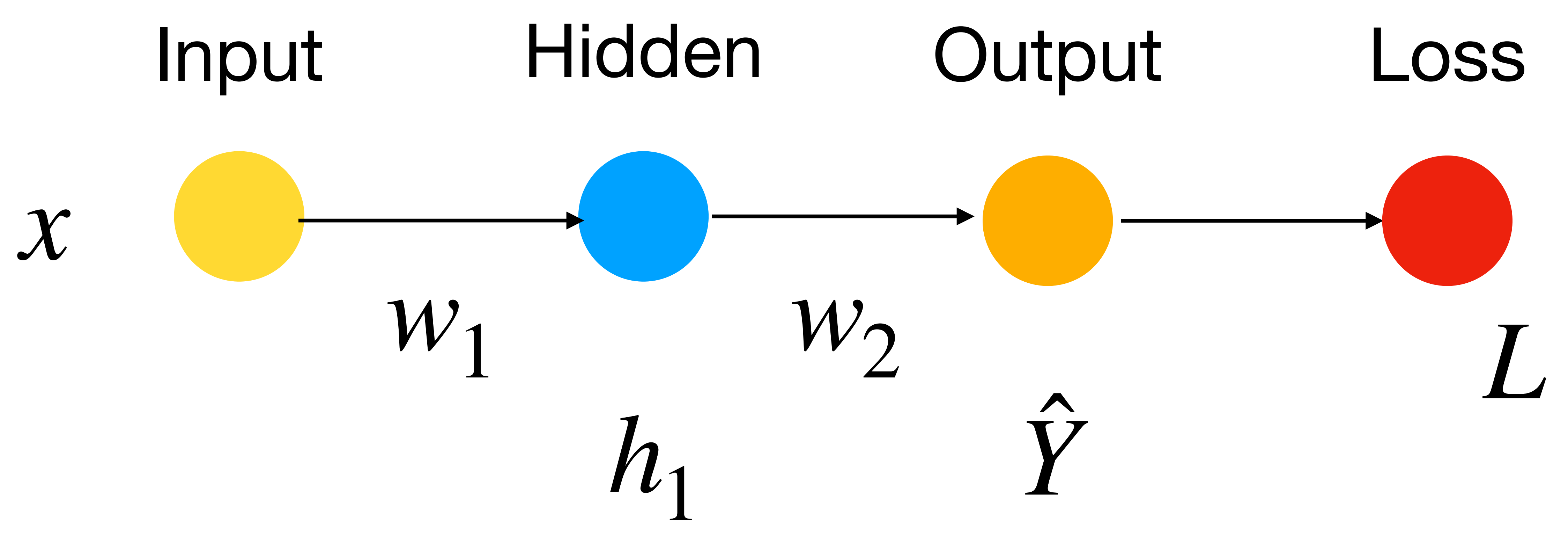
Single Variable, Single Neuron Neural Network Forward Pass

ACADENTIAL

Multi-variate Multi-neuron NN

Single variate, Single Neuron의 경우

앞에서 고려한 매우 단순한 뉴럴넷 예제

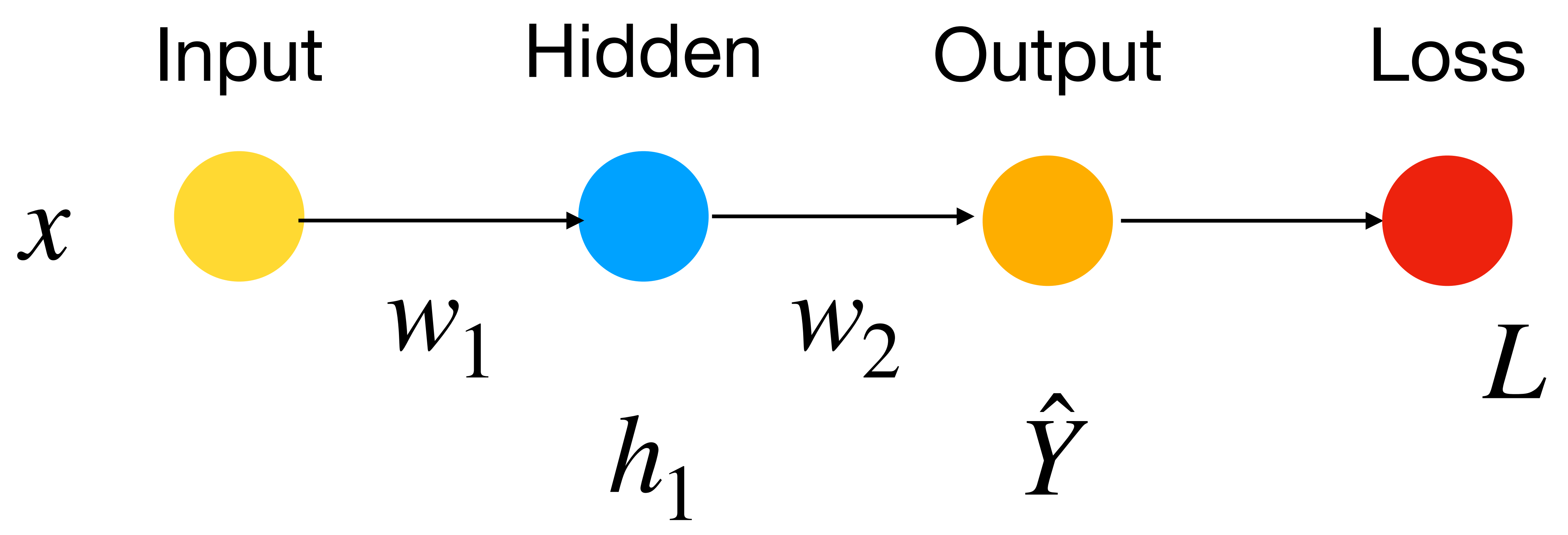


일단 편의상 Activation Function은 생략

Multi-variate Multi-neuron NN

Single variate, Single Neuron의 경우

앞에서 고려한 매우 단순한 뉴럴넷 예제



- Input → Hidden

$$h_1 = w_1 x$$

- Hidden → Output

$$\hat{Y} = w_2 h_1$$

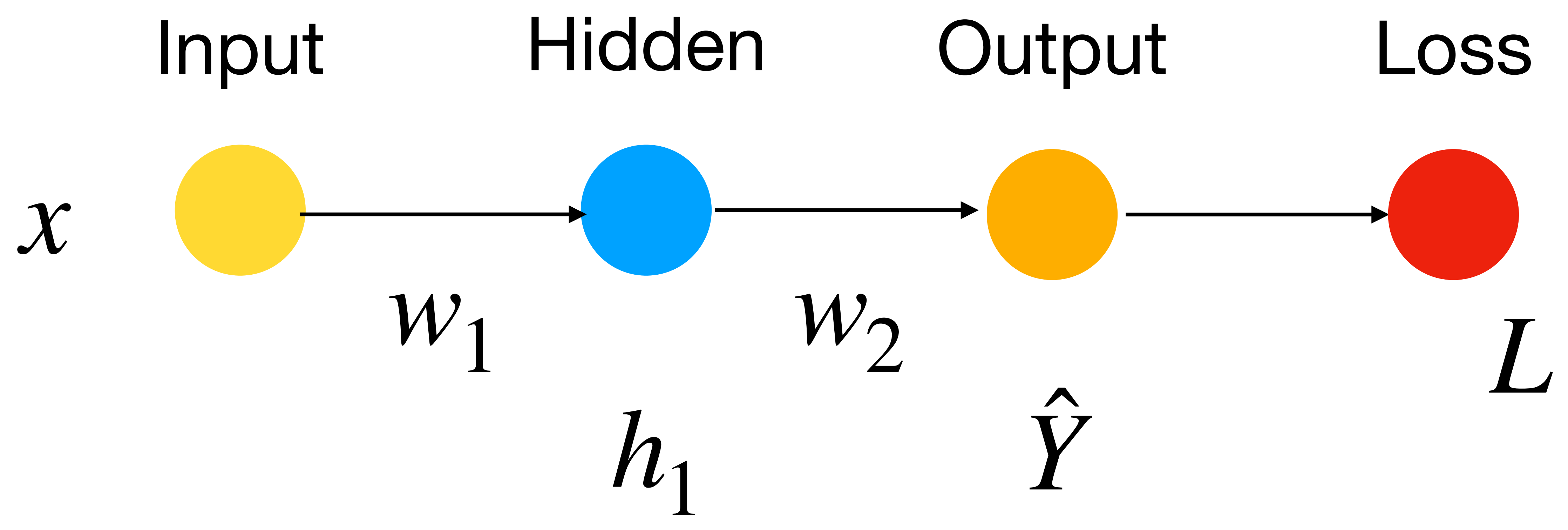
- Output → Loss

$$L = L(Y, \hat{Y})$$

Multi-variate Multi-neuron NN

Single variate, Single Neuron의 경우

앞에서 고려한 매우 단순한 뉴럴넷 예제



- Input → Hidden

$$h_1 = w_1 x$$

- Hidden → Output

$$\hat{Y} = w_2 h_1$$

- Output → Loss

$$L = L(Y, \hat{Y})$$

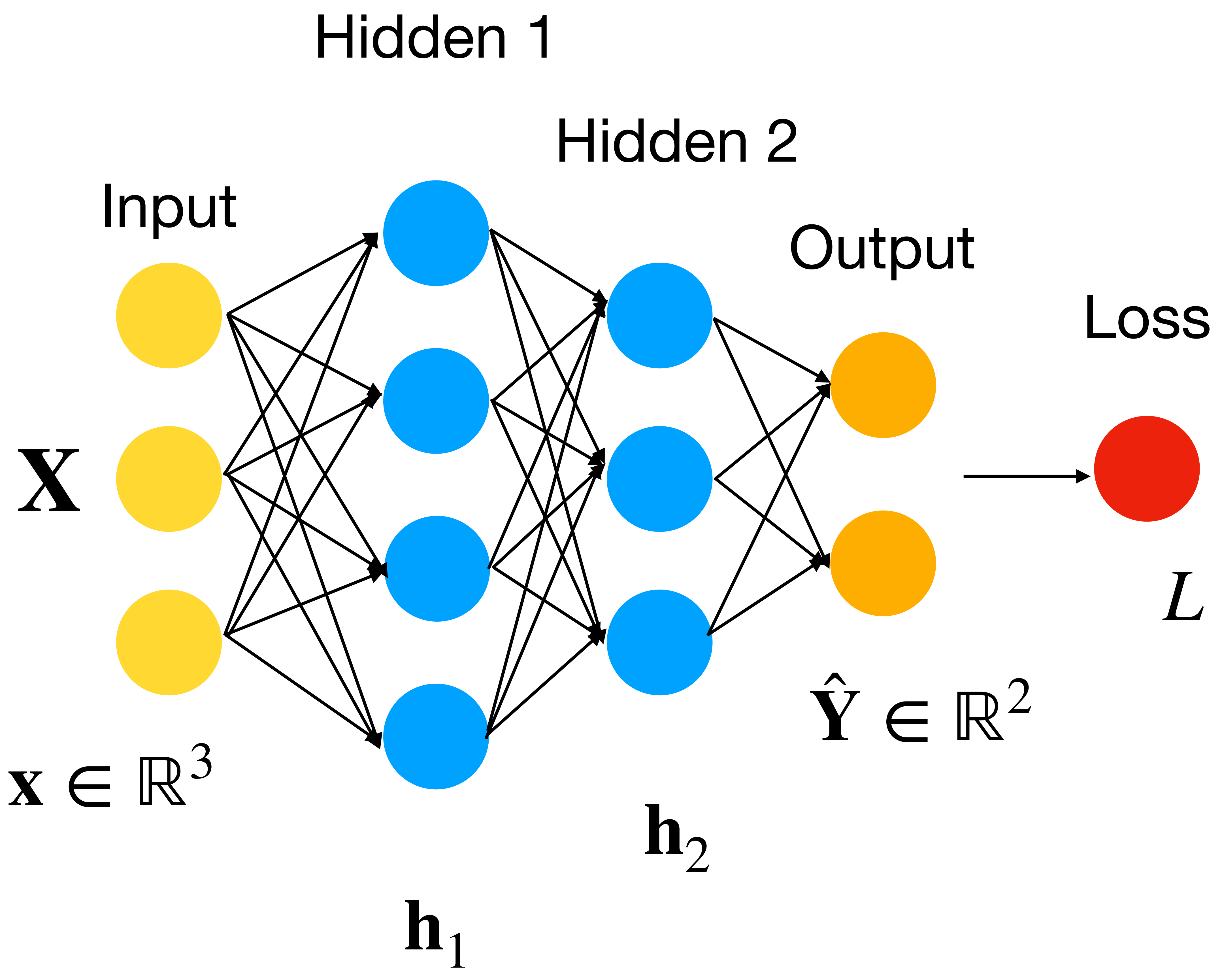
단순히 scalar의 곱이다!

Multivariate Multi-Neuron Neural Network Forward Pass

ACADENTIAL

Multi-variate Multi-neuron NN

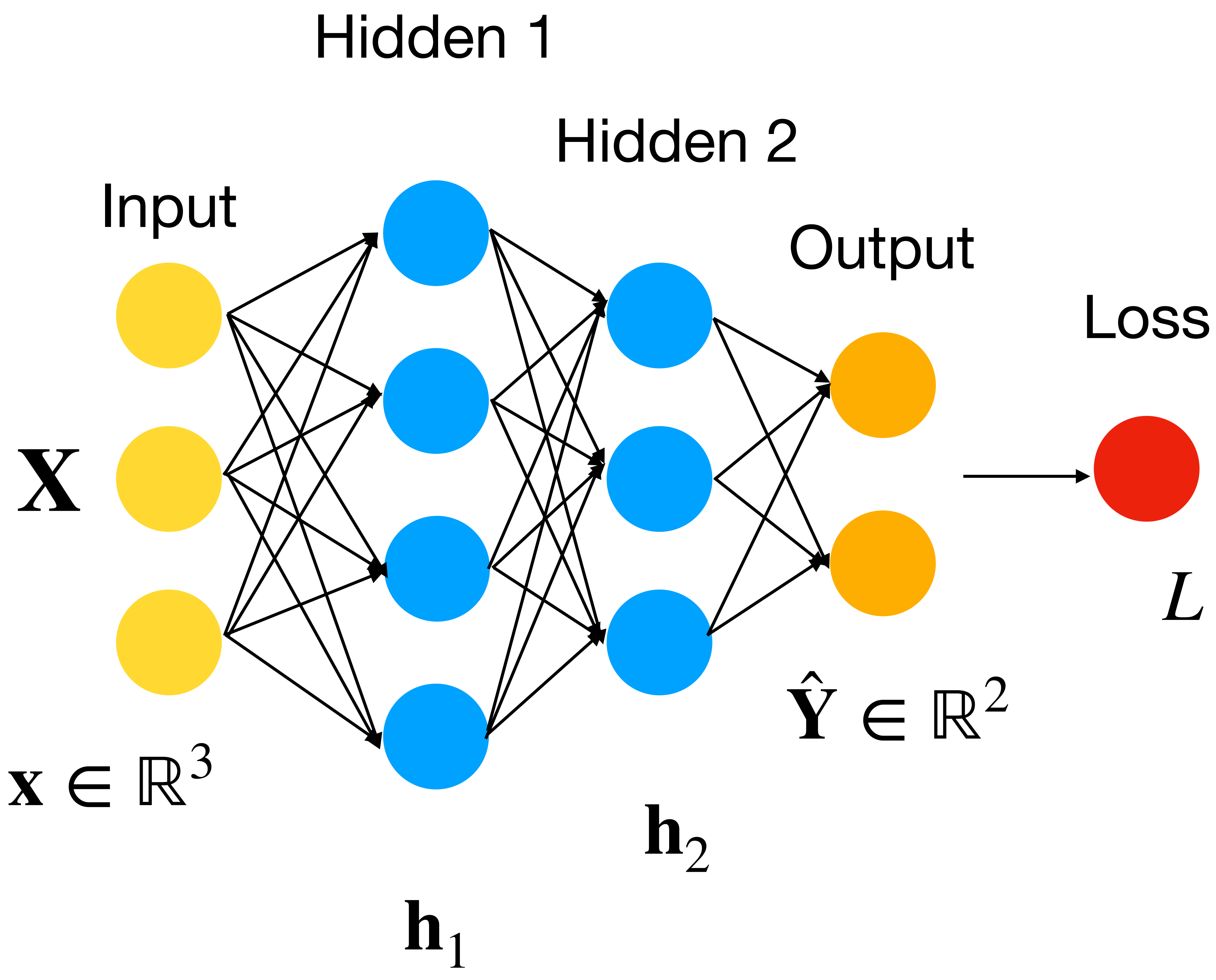
Multi variate, Multi Neuron의 경우



Multi-variate, Multi-neuron에 대해서는 forward pass을 어떻게 계산할까?

Multi-variate Multi-neuron NN

Multi variate, Multi Neuron의 경우



Multi-variate, Multi-neuron에 대해서는 forward pass을 어떻게 계산할까?

결론부터 먼저 말하면:

행렬의 곱 (Matrix Multiplication)

으로 표현할 수 있다!

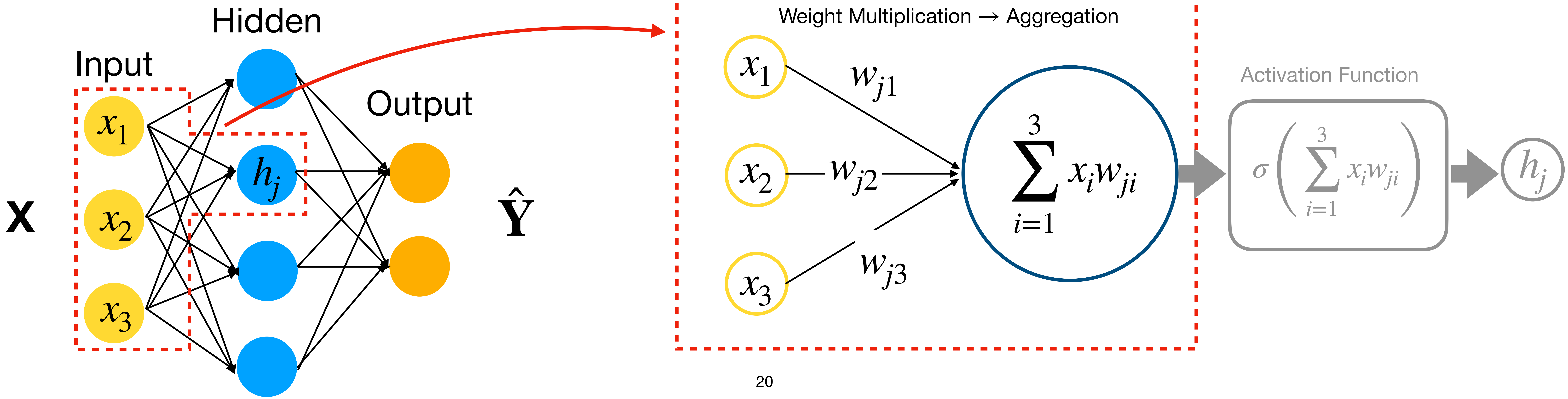
왜 그렇게 되는지 살펴보자!

Recap from Section 2

ACADENTIAL

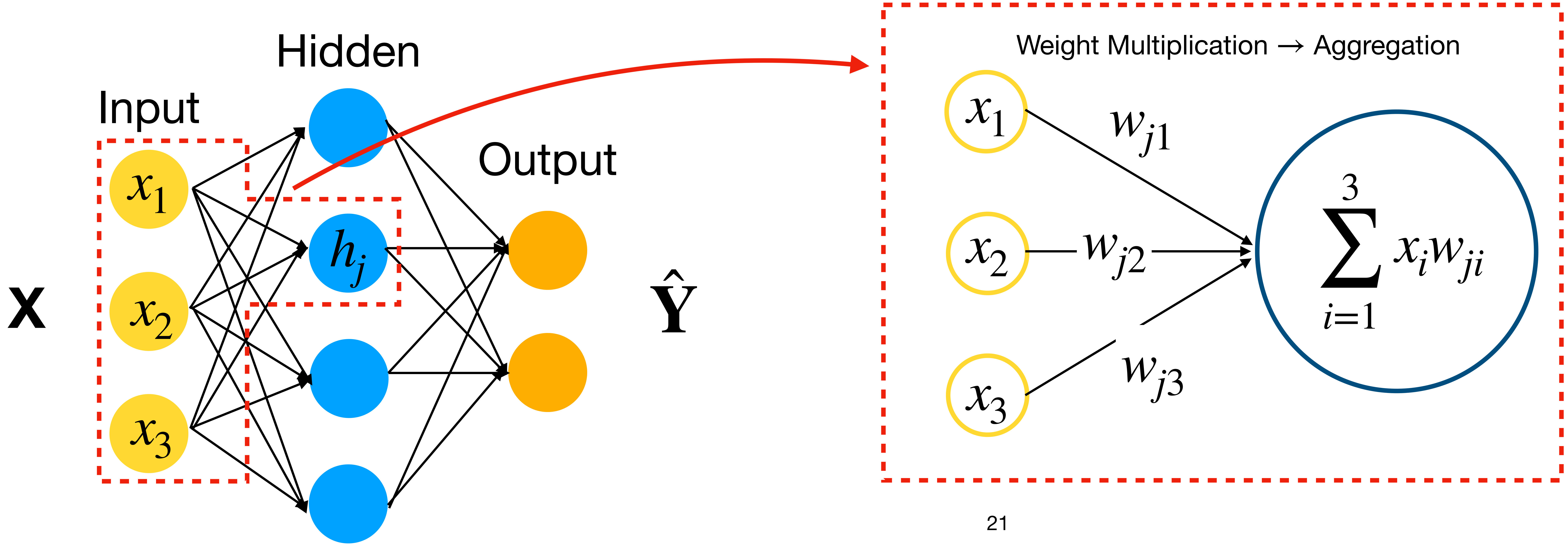
Recap from Section 2

- (일단 편의상 Activation Function에 대해서는 생략)
- $\sum_{i=1}^{3} x_i w_{ji}$: 이전 Layer의 출력값 x_i은 가중치 w_{ji} 에 곱해져서 합해진다. (weight multiplication → aggregation)



Recap from Section 2

- 참고로 w_{ji}에서
  - j 은 현재 Hidden layer에서 j 번째 뉴론을 의미
  - i 은 이전 layer에서 i 번째 뉴론을 의미

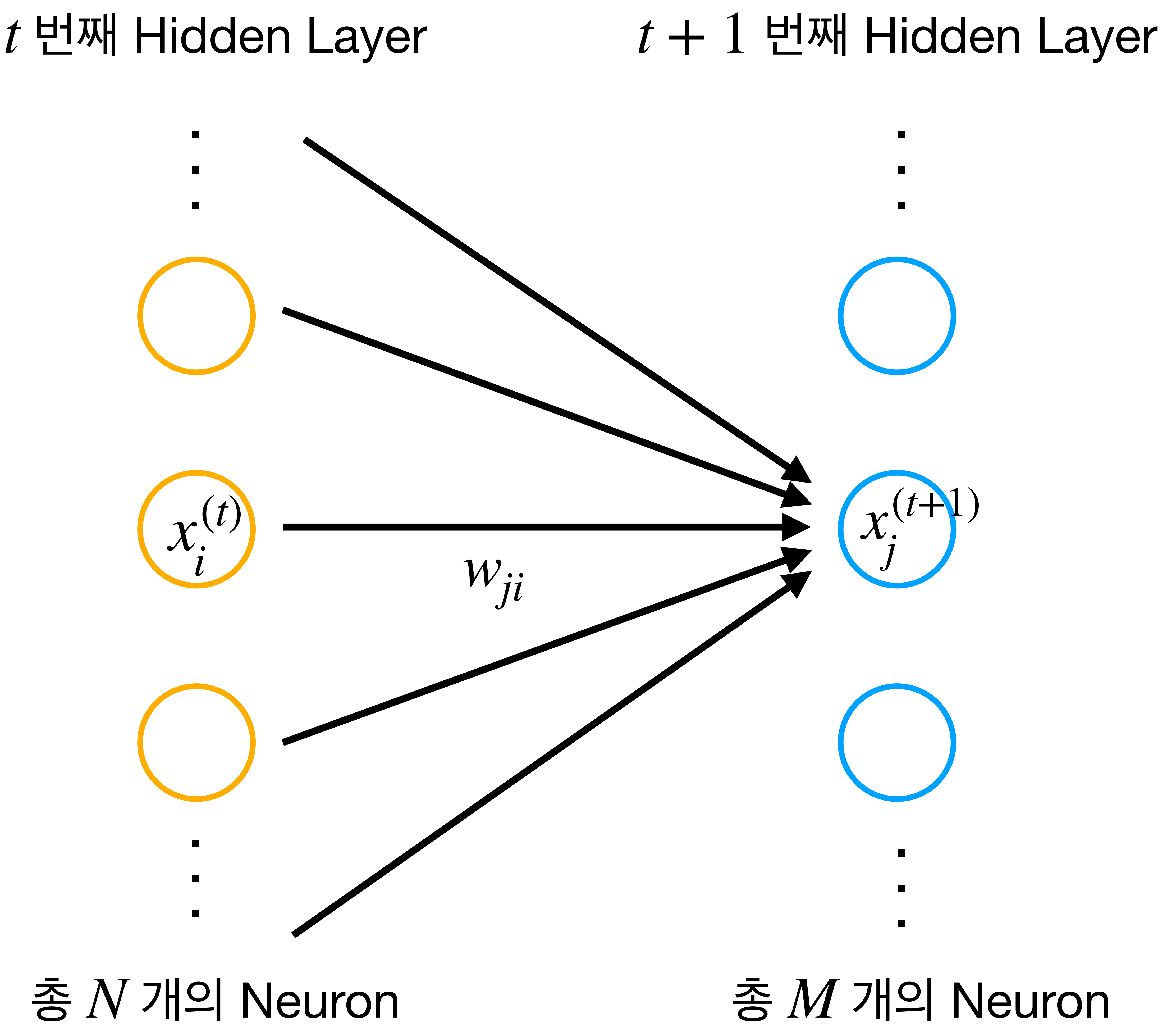


Forward Pass as Matrix Multiplication

Forward Pass

$$x_j^{(t+1)} = \sum_{i=1}^{N} w_{ji} x_i^{(t)}$$

$$= w_{j1}x_1 + w_{j2}x_2 + \cdots + w_{jN}x_N$$

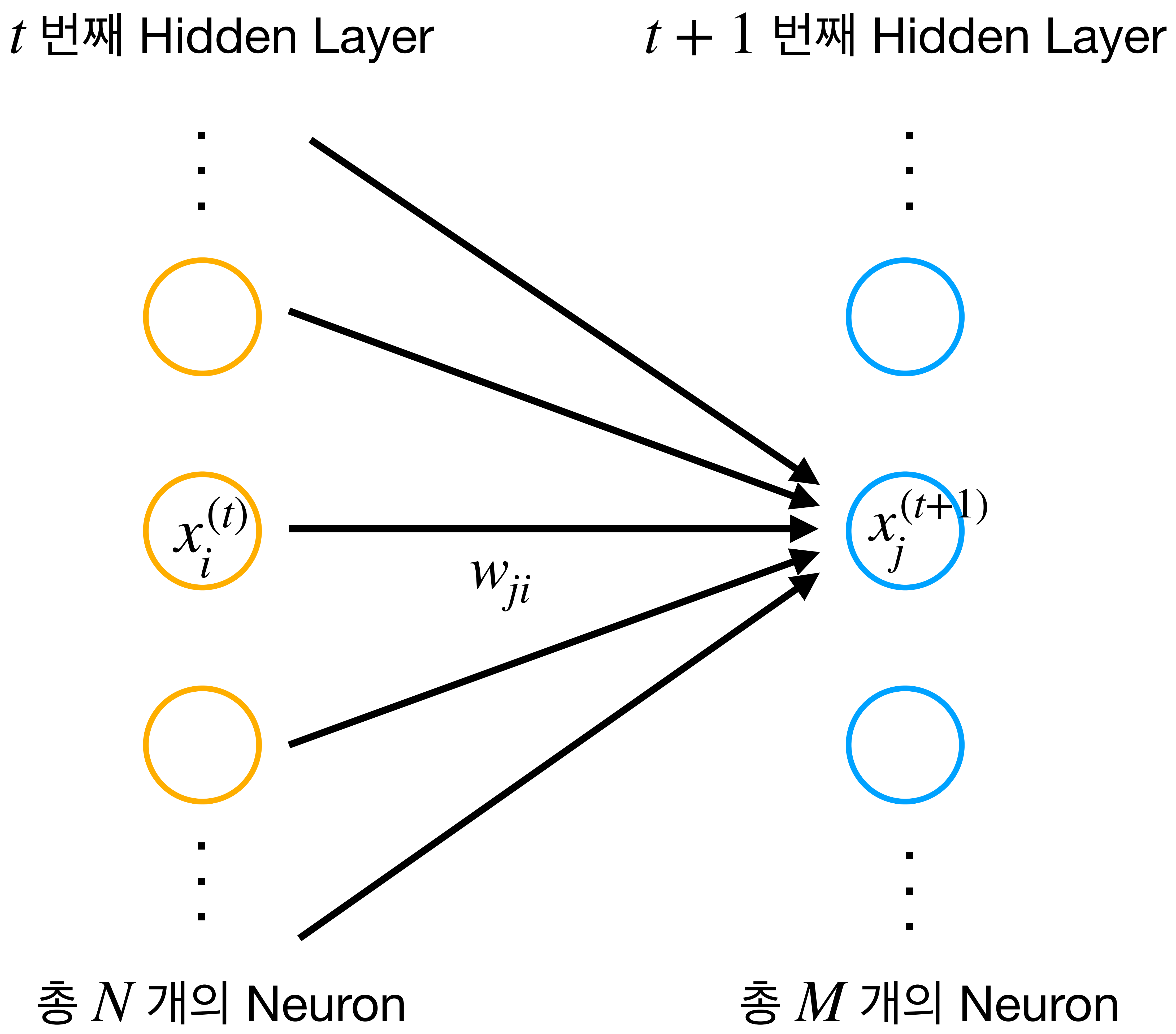


Forward Pass

$$x_j^{(t+1)} = \sum_{i=1}^{N} w_{ji} x_i^{(t)}$$

$$\boxed{= w_{j1}x_1 + w_{j2}x_2 + \cdots + w_{jN}x_N}$$

Vector의 내적으로 표현할 수 있다!



Forward Pass

행렬의 곱 Recap

$$\begin{pmatrix} w_{j1} & \dots & w_{ji} & \dots & w_{jN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \dots \\ x_i^{(t)} \\ \dots \\ x_N^{(t)} \end{pmatrix} = w_{j1}x_1^{(t)} + \cdots + w_{ji}x_i^{(t)} + \cdots + w_{jN}x_N^{(t)}$$

Forward Pass

행렬의 곱 Recap

$$\begin{pmatrix} \boxed{w_{j1}} & \cdots & w_{ji} & \cdots & w_{jN} \end{pmatrix} \begin{pmatrix} \boxed{x_1^{(t)}} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = \boxed{w_{j1}x_1^{(t)}} + \cdots + w_{ji}x_i^{(t)} + \cdots + w_{jN}x_N^{(t)}$$

Forward Pass

행렬의 곱 Recap

$$\begin{pmatrix} w_{j1} & \cdots & \boxed{w_{ji}} & \cdots & w_{jN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ \boxed{x_i^{(t)}} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = w_{j1}x_1^{(t)} + \cdots + \boxed{w_{ji}x_i^{(t)}} + \cdots + w_{jN}x_N^{(t)}$$

Forward Pass

행렬의 곱 Recap

$$\begin{pmatrix} w_{j1} & \cdots & w_{ji} & \cdots & \boxed{w_{jN}} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ \boxed{x_N^{(t)}} \end{pmatrix} = w_{j1}x_1^{(t)} + \cdots + w_{ji}x_i^{(t)} + \cdots + \boxed{w_{jN}x_N^{(t)}}$$

ACADENTIAL

Forward Pass

행렬의 곱 Recap

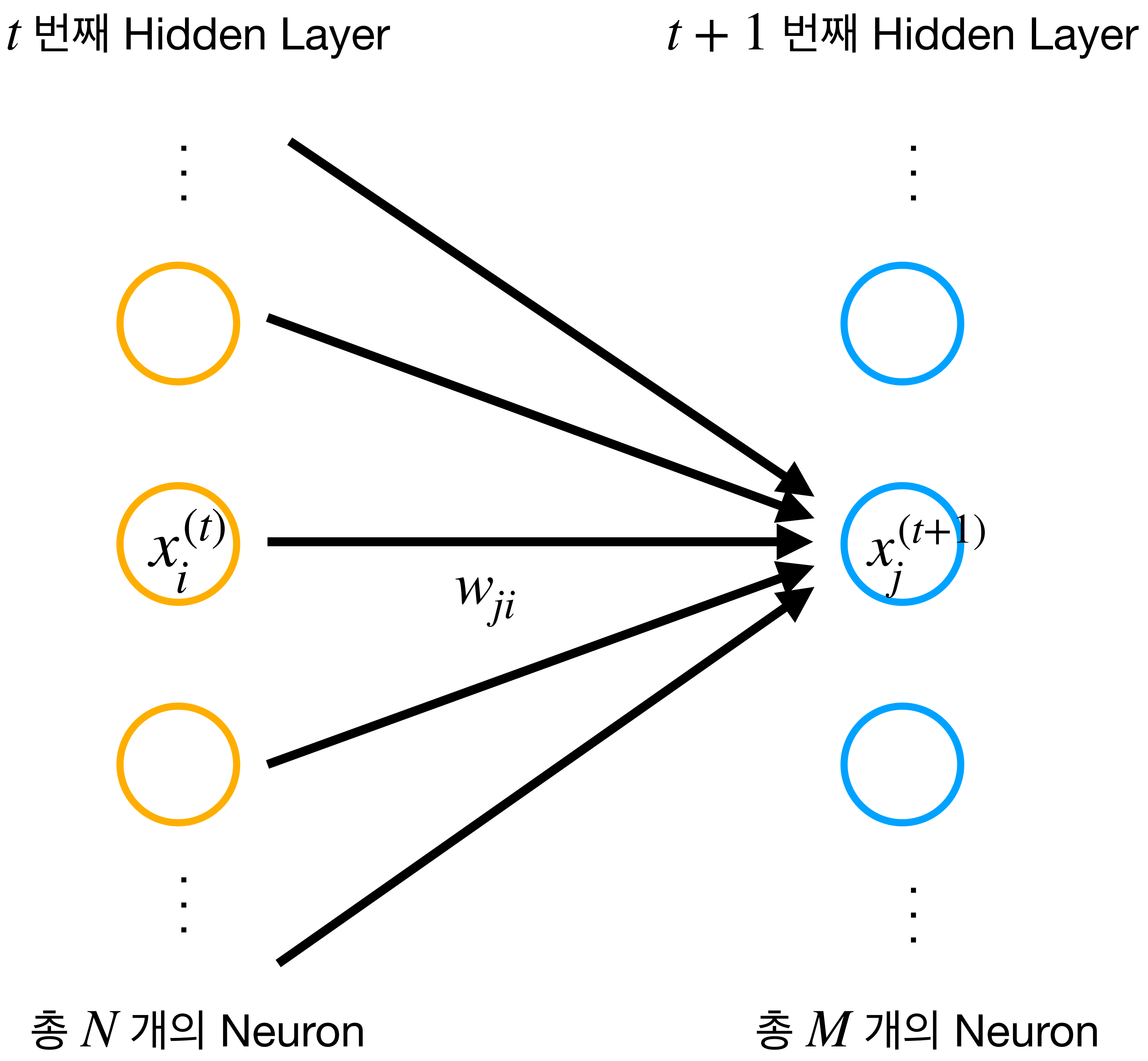
$$\begin{pmatrix} w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = w_{j1}x_1^{(t)} + \cdots + w_{ji}x_i^{(t)} + \cdots + w_{jN}x_N^{(t)}$$

$$= \sum_{i=1}^{N} w_{ji}x_i^{(t)}$$

Forward Pass

행렬의 곱 Recap

이것은 t+1 번째 Hidden layer에서 j 번째 neuron에 해당된다.
나머지 1~M개의 neuron에 대해서도 확장해보자!

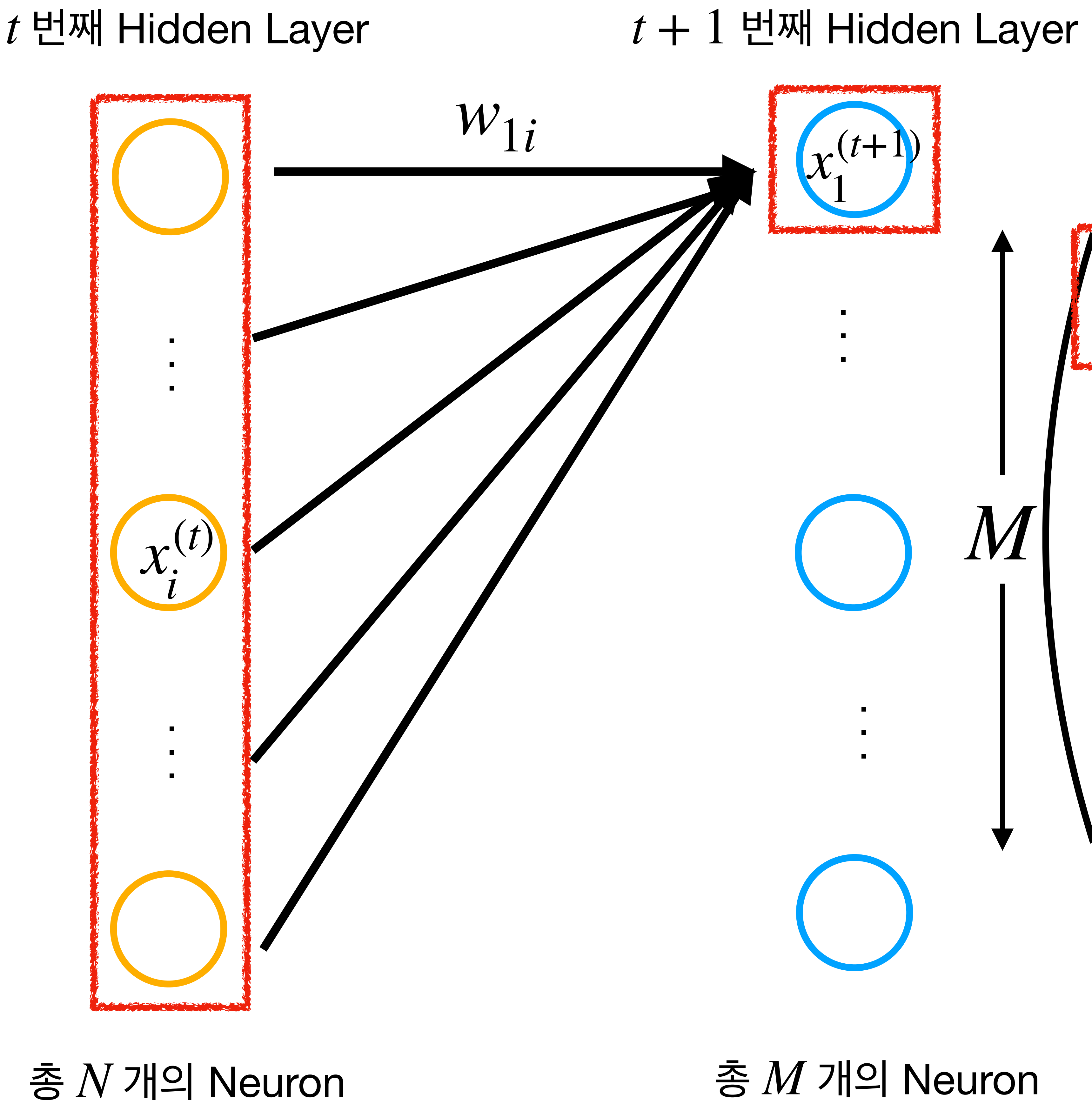


$$\boxed{\begin{pmatrix} w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \end{pmatrix}} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = \sum_{i=1}^{N} w_{ji} x_i^{(t)} = x_j^{(t+1)}$$

Forward Pass

행렬의 곱 Recap

$$\mathbf{x}^{(t+1)} = W\mathbf{x}^{(t)}$$



$$\begin{array}{cc} & \overset{\longleftarrow\; N \;\longrightarrow}{} \\ M & \begin{pmatrix} w_{11} & \cdots & w_{1i} & \cdots & w_{1N} \\ w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \\ w_{M1} & \cdots & w_{Mi} & \cdots & w_{MN} \end{pmatrix} \end{array} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N} w_{1i}x_i^{(t)} \\ \cdots \\ \sum_{i=1}^{N} w_{ji}x_i^{(t)} \\ \cdots \\ \sum_{i=1}^{N} w_{Mi}x_i^{(t)} \end{pmatrix} = \begin{pmatrix} x_1^{(t+1)} \\ \cdots \\ x_j^{(t+1)} \\ \cdots \\ x_N^{(t+1)} \end{pmatrix}$$

$W \in \mathbb{R}^{M\times N}$ $\quad$ $\mathbf{x}^{(t)} \in \mathbb{R}^{N}$ $\quad$ $\mathbf{x}^{(t+1)} \in \mathbb{R}^{M}$

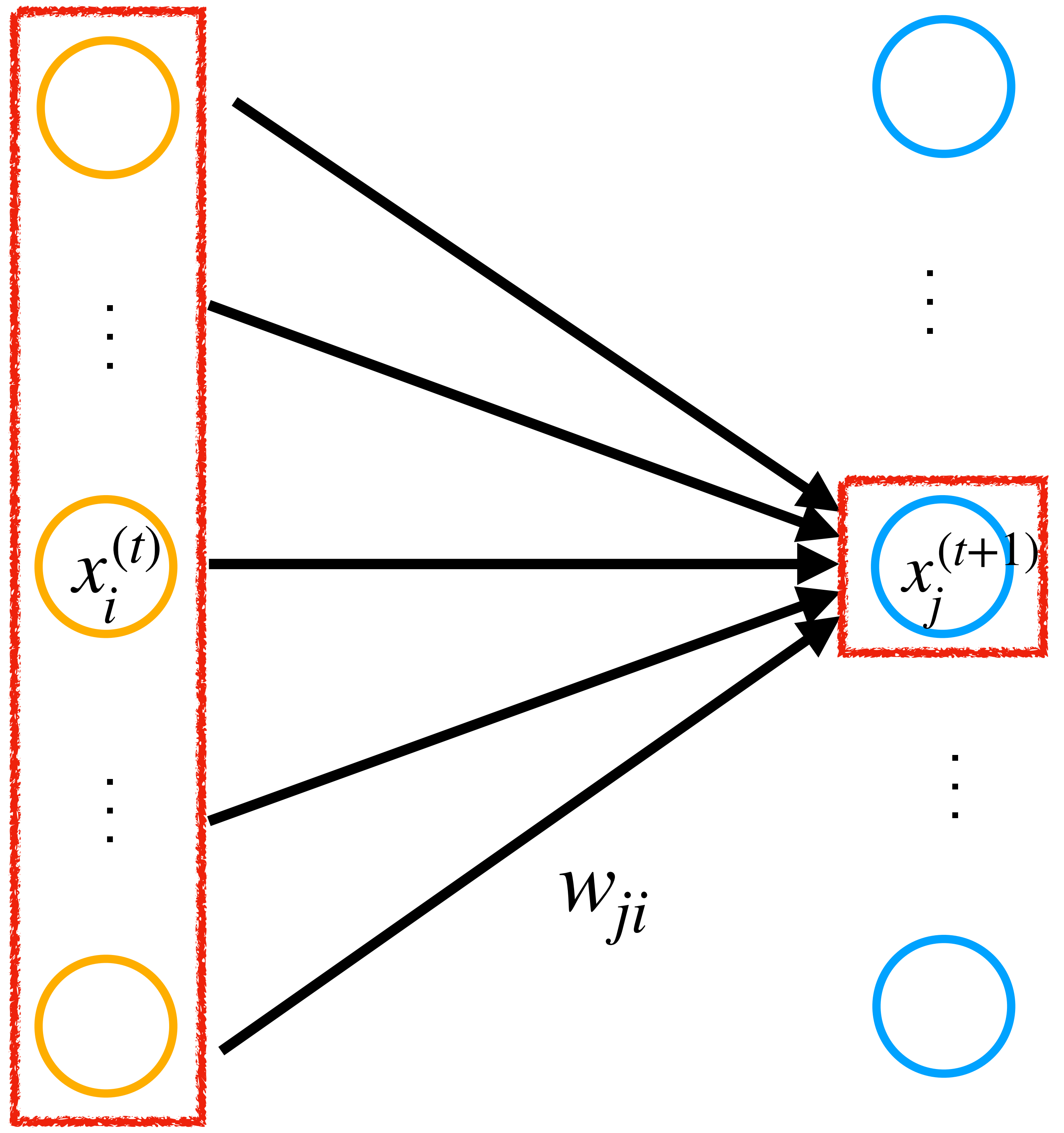
Forward Pass

행렬의 곱 Recap

$$\mathbf{x}^{(t+1)} = W\mathbf{x}^{(t)}$$

t 번째 Hidden Layer

$t+1$ 번째 Hidden Layer



총 N 개의 Neuron

총 M 개의 Neuron

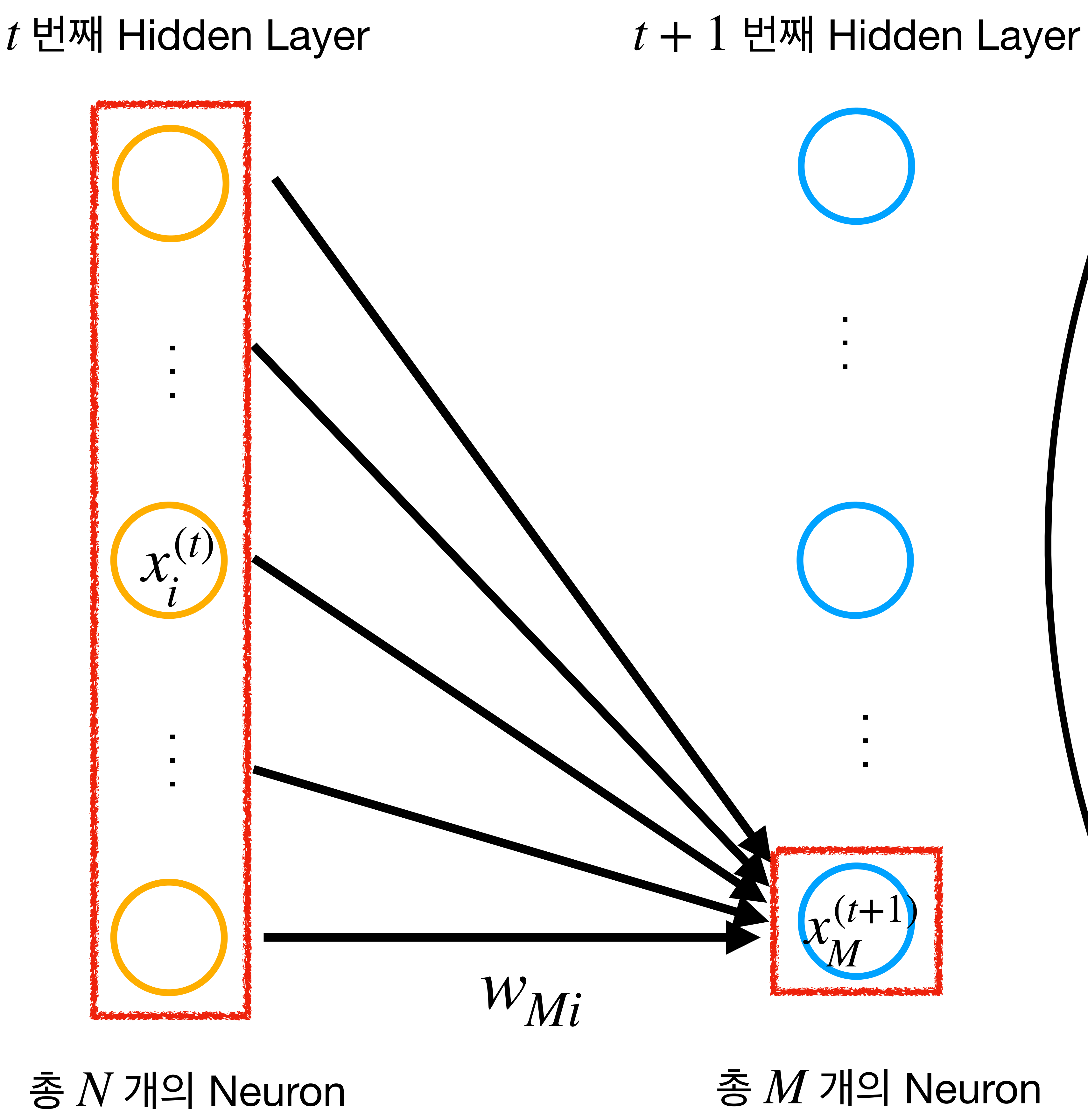
$$\begin{pmatrix} w_{11} & \cdots & w_{1i} & \cdots & w_{1N} \\ w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \\ w_{M1} & \cdots & w_{Mi} & \cdots & w_{MN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N} w_{1i} x_i^{(t)} \\ \cdots \\ \sum_{i=1}^{N} w_{ji} x_i^{(t)} \\ \cdots \\ \sum_{i=1}^{N} w_{Mi} x_i^{(t)} \end{pmatrix} = \begin{pmatrix} x_1^{(t+1)} \\ \cdots \\ x_j^{(t+1)} \\ \cdots \\ x_N^{(t+1)} \end{pmatrix}$$

$W \in \mathbb{R}^{M \times N}$ $\quad$ $\mathbf{x}^{(t)} \in \mathbb{R}^{N}$ $\quad$ $\mathbf{x}^{(t+1)} \in \mathbb{R}^{M}$

Forward Pass

행렬의 곱 Recap

$$\mathbf{x}^{(t+1)} = W\mathbf{x}^{(t)}$$



$$\begin{pmatrix} w_{11} & \cdots & w_{1i} & \cdots & w_{1N} \\ w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \\ w_{M1} & \cdots & w_{Mi} & \cdots & w_{MN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N} w_{1i}x_i^{(t)} \\ \cdots \\ \sum_{i=1}^{N} w_{ji}x_i^{(t)} \\ \cdots \\ \sum_{i=1}^{N} w_{Mi}x_i^{(t)} \end{pmatrix} = \begin{pmatrix} x_1^{(t+1)} \\ \cdots \\ x_j^{(t+1)} \\ \cdots \\ x_N^{(t+1)} \end{pmatrix}$$

$W \in \mathbb{R}^{M\times N}$ $\quad \mathbf{x}^{(t)} \in \mathbb{R}^{N}$ $\quad \mathbf{x}^{(t+1)} \in \mathbb{R}^{M}$

Forward Pass

$$\begin{pmatrix} w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = x_j^{(t+1)}$$

정리하자면:

$$\boxed{x_j^{(t+1)}} = \sum_{i=1}^{N} \boxed{w_{ji}} \boxed{x_i^{(t)}}$$

Weight Matrix	Input vector	Output vector

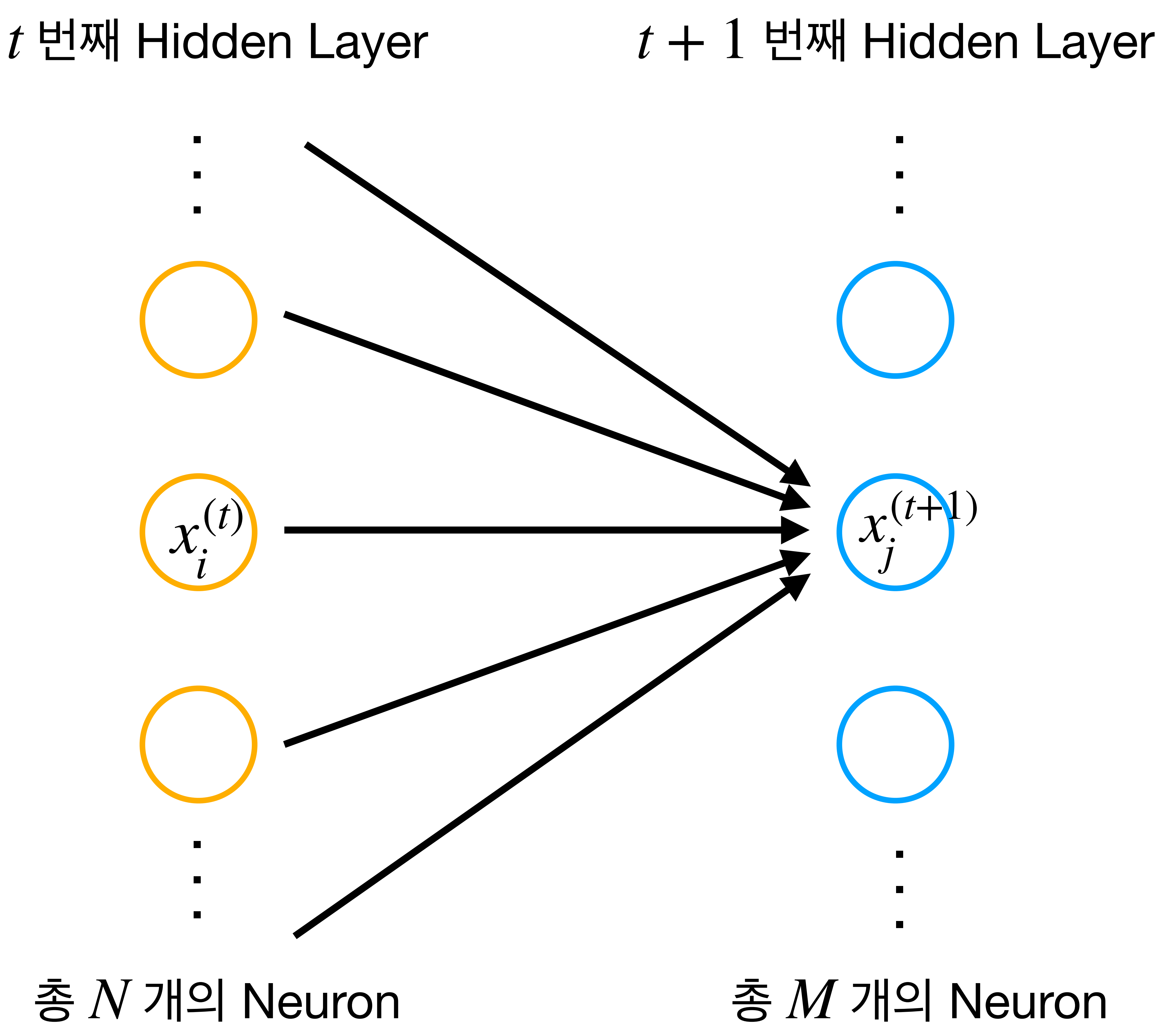
column = N개

row = M개

$$\begin{pmatrix} w_{11} & \cdots & w_{1i} & \cdots & w_{1N} \\ & & \cdots & & \\ w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \\ & & \cdots & & \\ w_{M1} & \cdots & w_{Mi} & \cdots & w_{MN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix} = \begin{pmatrix} x_1^{(t+1)} \\ \cdots \\ x_j^{(t+1)} \\ \cdots \\ x_M^{(t+1)} \end{pmatrix}$$

M x N matrix | N vector | M vector

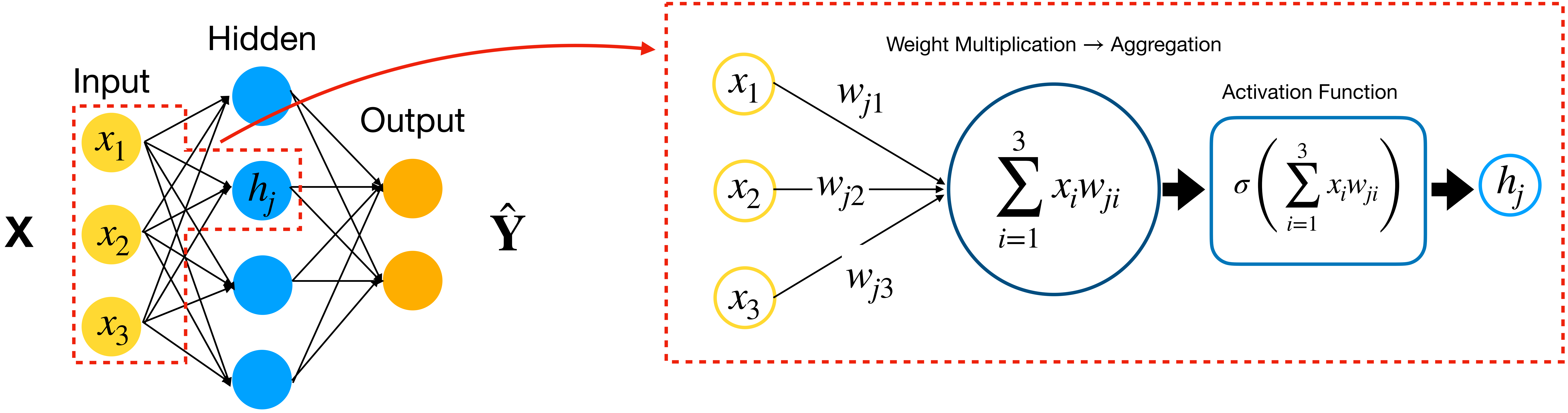
Forward Pass



즉, (Activation function이 제외된) Neural Network은 단순히 matrix의 곱이다!

$$\mathbf{x}^{(t+1)} = W\mathbf{x}^{(t)}$$

ACADENTIAL



만약에 Activation Function을 포함한다면,

$$\mathbf{h} = \sigma(W\mathbf{x})$$

ACADENTIAL

Activation Function도 포함하는 경우

$$\boxed{x_j^{(t+1)}} = \sigma\left(\sum_{i=1}^{N} \boxed{w_{ji}}\,\boxed{x_i^t}\right) \rightarrow \mathbf{x}^{(t+1)} = \sigma\left(W\mathbf{x}^{(t)}\right)$$

Weight Matrix **Input vector** **Output vector**

$$\sigma\left(\begin{pmatrix} w_{11} & \cdots & w_{1i} & \cdots & w_{1N} \\ & & \cdots & & \\ w_{j1} & \cdots & w_{ji} & \cdots & w_{jN} \\ & & \cdots & & \\ w_{M1} & \cdots & w_{Mi} & \cdots & w_{MN} \end{pmatrix} \begin{pmatrix} x_1^{(t)} \\ \cdots \\ x_i^{(t)} \\ \cdots \\ x_N^{(t)} \end{pmatrix}\right) = \begin{pmatrix} \sigma\left(\sum_{i=1}^{N} w_{1i}x_i^{(t)}\right) \\ \cdots \\ \sigma\left(\sum_{i=1}^{N} w_{ji}x_i^{(t)}\right) \\ \cdots \\ \sigma\left(\sum_{i=1}^{N} w_{Mi}x_i^{(t)}\right) \end{pmatrix} = \begin{pmatrix} x_1^{(t+1)} \\ \cdots \\ x_j^{(t+1)} \\ \cdots \\ x_M^{(t+1)} \end{pmatrix}$$