

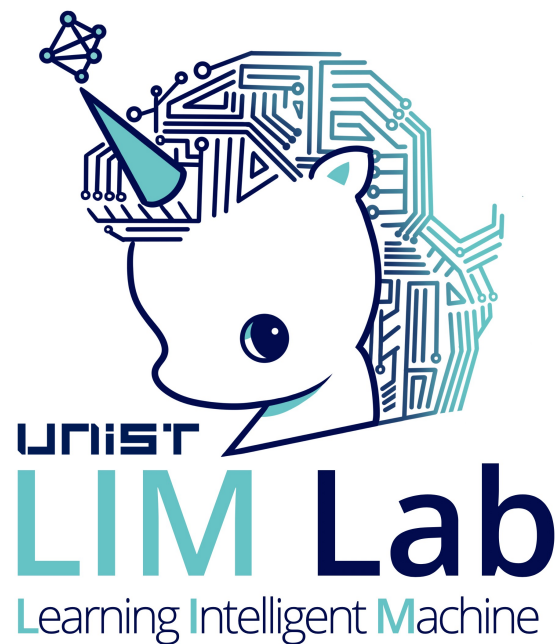
Mathematics for Artificial Intelligence

9강: CNN 첫걸음

임성빈

UNIST

인공지능대학원 & 산업공학과
Learning Intelligent Machine Lab

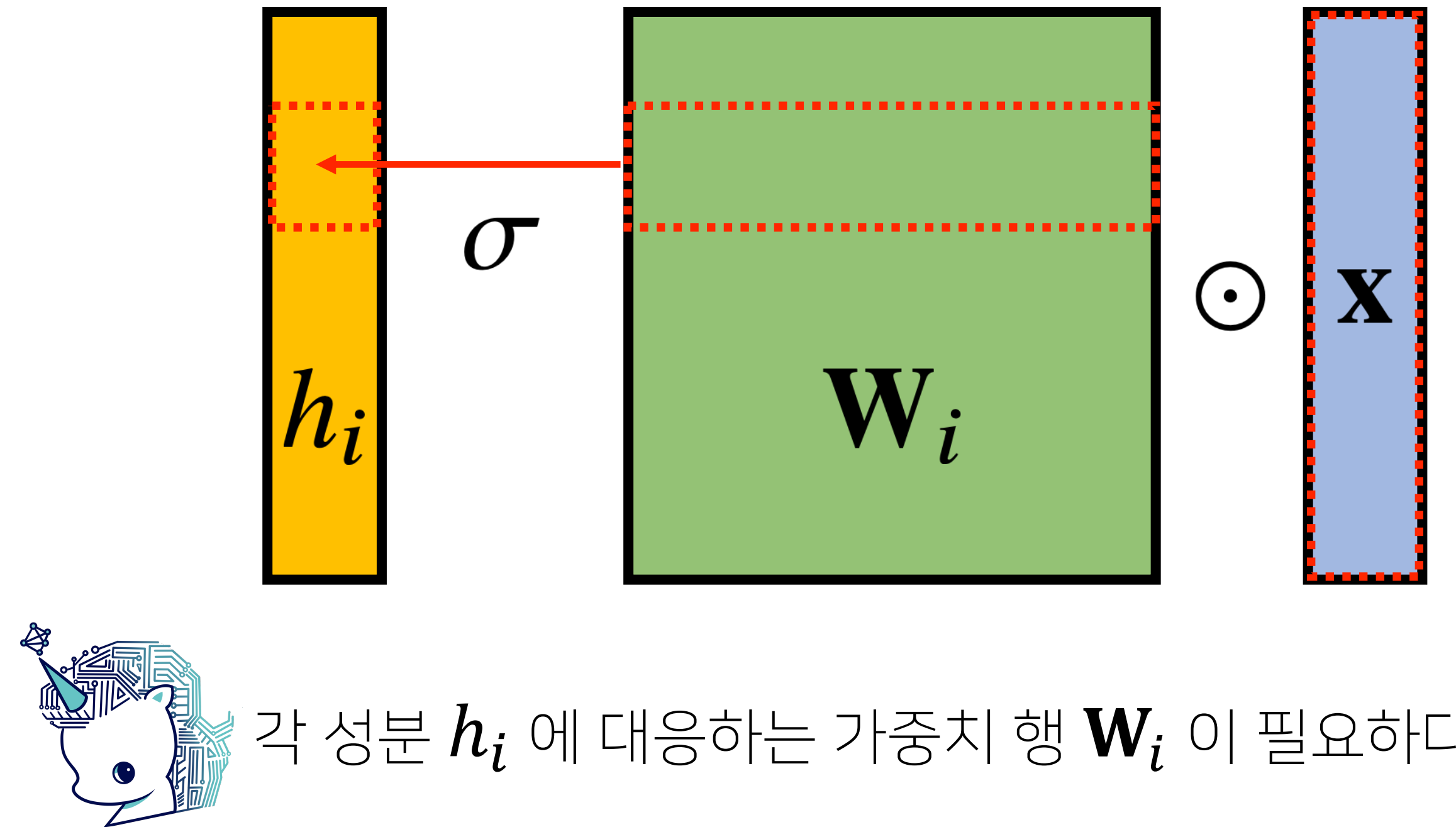


Convolution 연산 이해하기

- 지금까지 배운 다층신경망(MLP)은 ^T각 뉴런들이 선형모델과 활성화함수로 모두 연결된 (fully connected) 구조였습니다

$$h_i = \sigma \left(\sum_{j=1}^p W_{ij} x_j \right)$$

활성함수 σ 가중치 행렬 W_{ij}

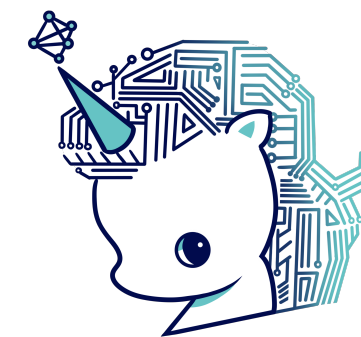
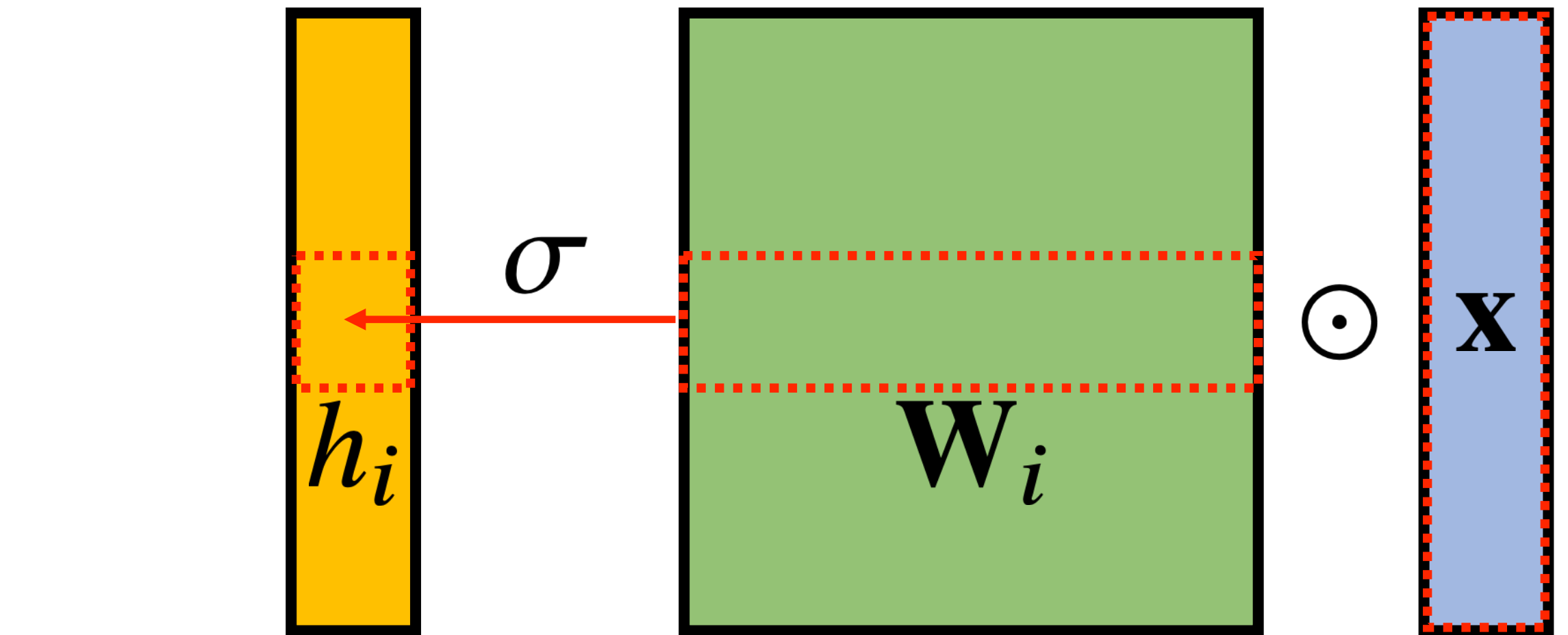


Convolution 연산 이해하기

- 지금까지 배운 다층신경망(MLP)은 각 뉴런들이 선형모델과 활성화함수로 모두 연결된 (fully connected) 구조였습니다

$$h_i = \sigma \left(\sum_{j=1}^p W_{ij} x_j \right)$$

활성화함수 σ 가중치 행렬 W_{ij}

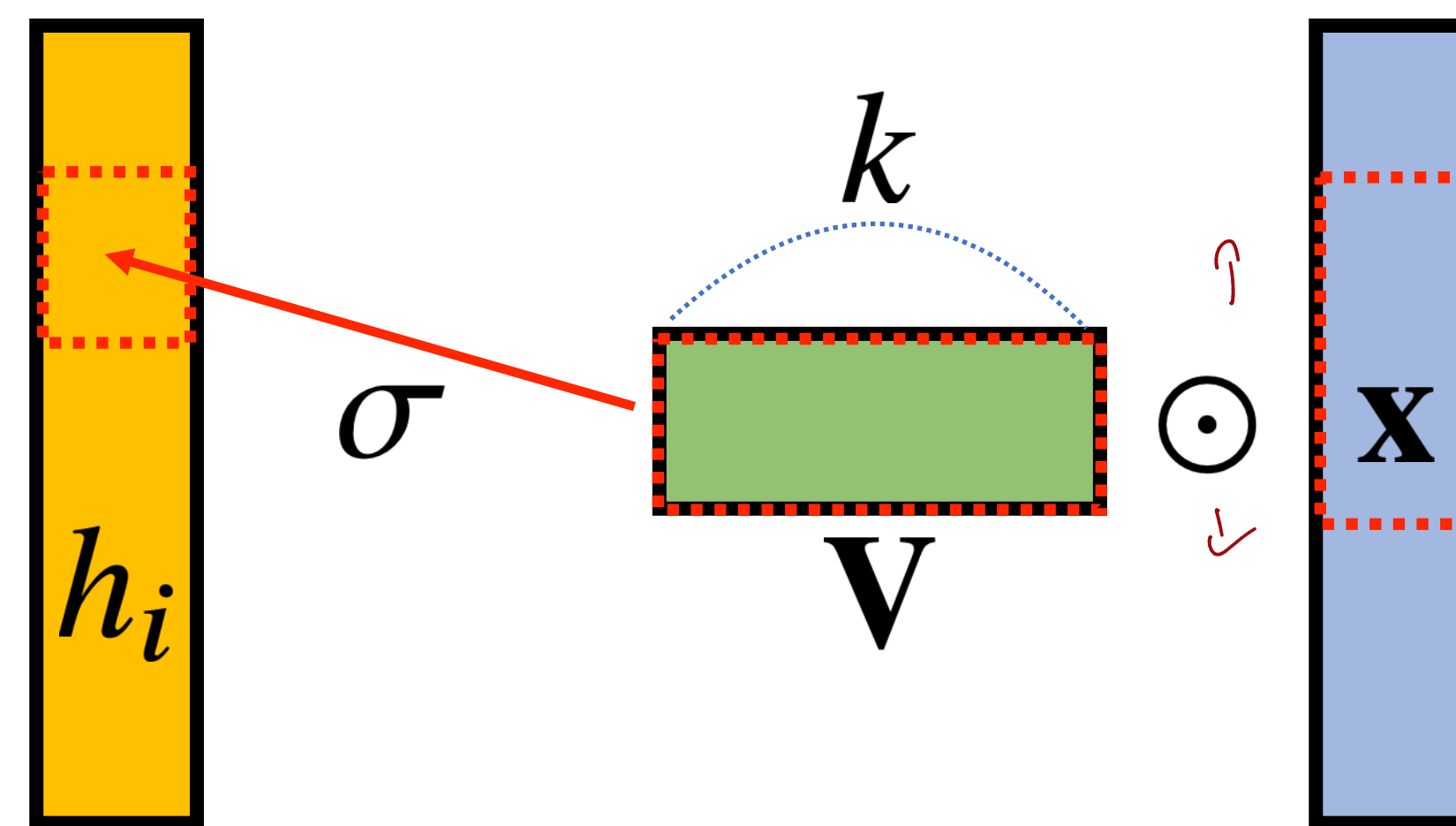


만일 i 가 바뀌면 사용되는 가중치도 바뀝니다
(가중치 행렬의 구조의 차이로 인해)

Convolution 연산 이해하기

- Convolution 연산은 이와 달리 커널(kernel)을 입력벡터 상에서 움직여가면서 선형모델과 합성함수가 적용되는 구조입니다

$$h_i = \sigma \left(\sum_{j=1}^k V_j x_{i+j-1} \right)$$

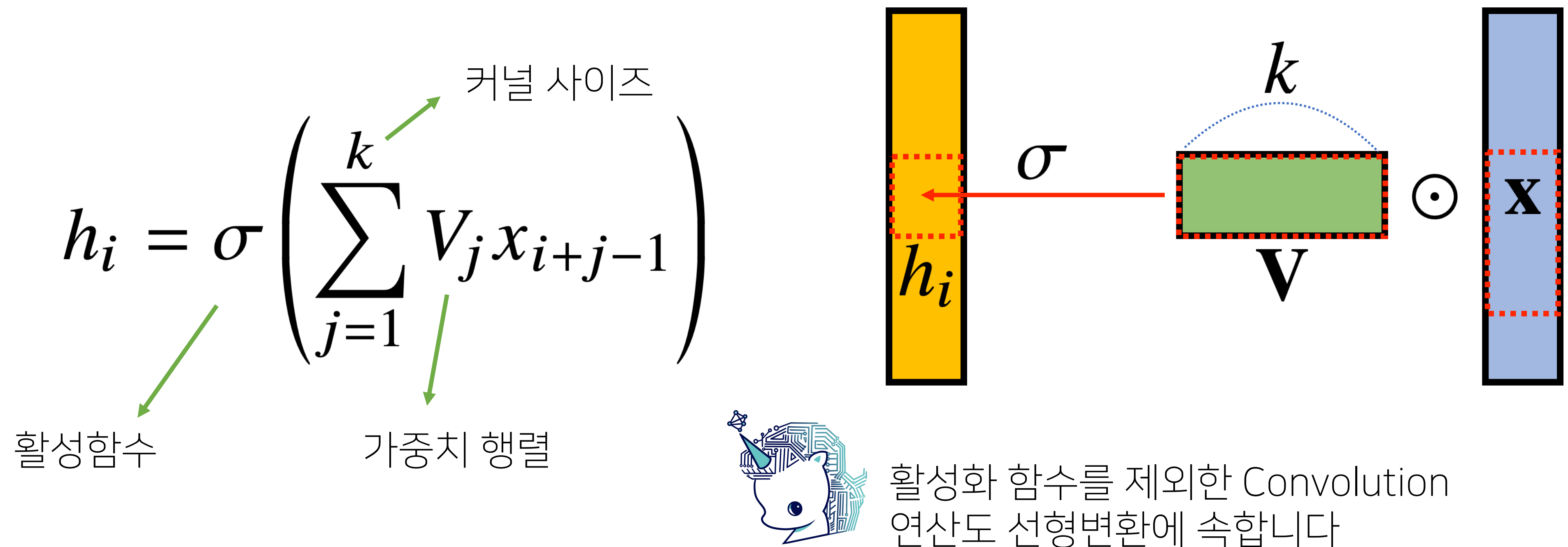


모든 i 에 대해 적용되는 커널은 \mathbf{V} 로 같고 커널의
사이즈만큼 \mathbf{x} 상에서 이동하면서 적용합니다

⇒ 가공의 행정력이 1세 전라시 비공식적이 아니라, 고장도 귀속을 인정 받기 상에 있어 움직여가면서 계층을 정립

Convolution 연산 이해하기

- Convolution 연산은 이와 달리 커널(kernel)을 입력벡터 상에서 움직여가면서 선형모델과 합성함수가 적용되는 구조입니다



Convolution 연산 이해하기

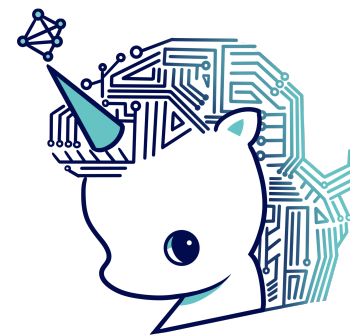
- Convolution 연산의 수학적 의미는 신호(signal)를 커널을 이용해 국소적으로 증폭 또는 감소시켜서 정보를 추출 또는 필터링하는 것입니다

continuous 연속 $[f * g](x) = \int_{\mathbb{R}^d} f(z)g(x-z)dz = \int_{\mathbb{R}^d} f(x-z)g(z)dz = [g * f](x)$

적분

discrete 이산 $[f * g](i) = \sum_{a \in \mathbb{Z}^d} f(a)g(i-a) = \sum_{a \in \mathbb{Z}^d} f(i-a)g(a) = [g * f](i)$

구분



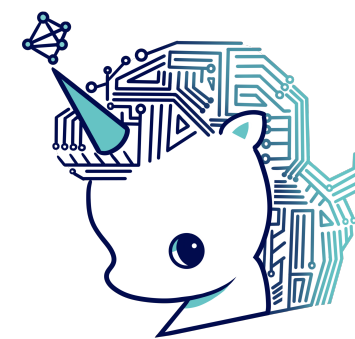
Convolution 을 수식으로만 이해하는 것은 매우 어렵습니다

Convolution 연산 이해하기

- Convolution 연산의 수학적 의미는 신호(signal)를 **커널을 이용해 국소적으로 증폭 또는 감소**시켜서 정보를 추출 또는 필터링하는 것입니다

continuous $[f * g](x) = \int_{\mathbb{R}^d} f(z)g(x \oplus z)dz = \int_{\mathbb{R}^d} f(x \oplus z)g(z)dz = [g * f](x)$

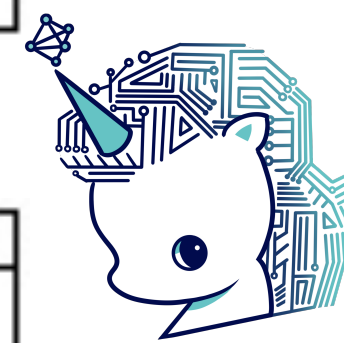
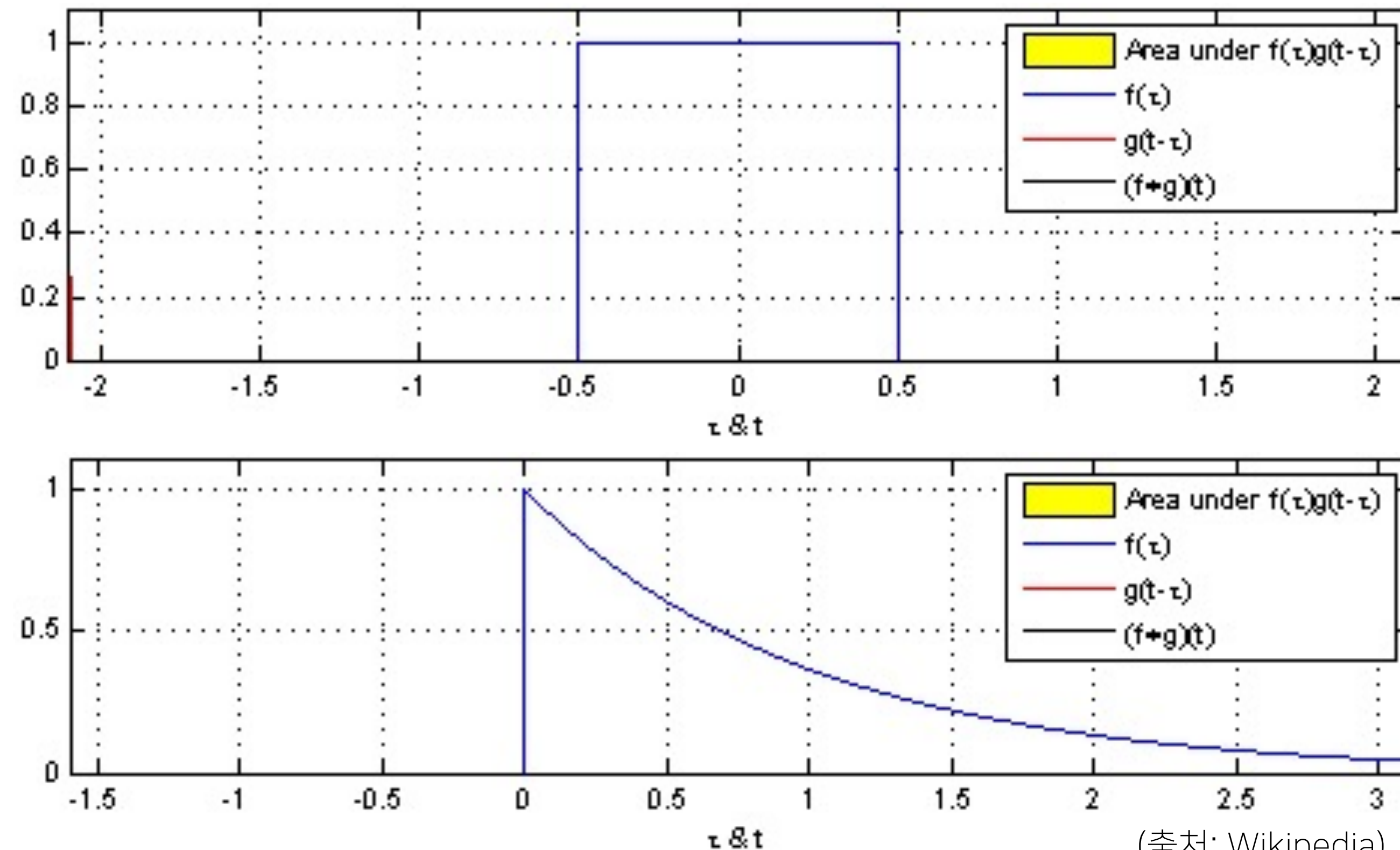
discrete $[f * g](i) = \sum_{a \in \mathbb{Z}^d} f(a)g(i \oplus a) = \sum_{a \in \mathbb{Z}^d} f(i \oplus a)g(a) = [g * f](i)$



CNN 에서 사용하는 연산은 사실 convolution 이 아니고 cross-correlation 이라 부릅니다

Convolution 연산 이해하기

- 커널은 정의역 내에서 움직여도 변하지 않고(translation invariant) 주어진 신호에 국소적(local)으로 적용합니다



파란색이 신호
빨간색이 커널
검은색이 결과입니다

영상처리에서 Convolution

0	0	0
0	1	0
0	0	0

custom ▼



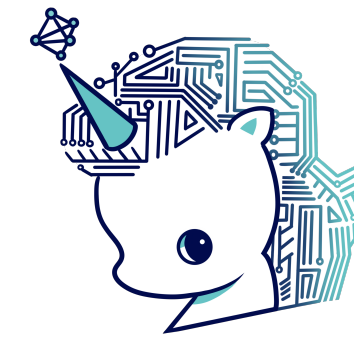
The **custom** kernel is whatever you make it.

For more, have a look at Gimp's excellent documentation on using [Image kernel's](#). You can also apply your own custom filters in Photoshop by going to Filter -> Other -> Custom...

다양한 차원에서의 Convolution

- Convolution 연산은 1차원뿐만 아니라 다양한 차원에서 계산 가능합니다

1D-conv $[f * g](i) = \sum_{p=1}^d f(p)g(i + p)$



데이터의 성격에 따라
사용하는 커널이 달라집니다

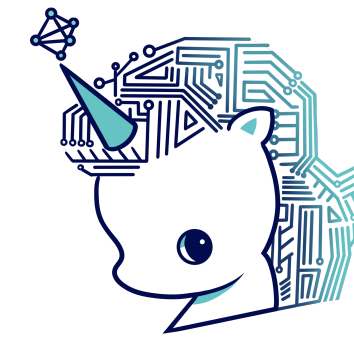
2D-conv $[f * g](i, j) = \sum_{p, q} f(p, q)g(i + p, j + q)$

3D-conv $[f * g](i, j, k) = \sum_{p, q, r} f(p, q, r)g(i + p, j + q, k + r)$

다양한 차원에서의 Convolution

- Convolution 연산은 1차원뿐만 아니라 다양한 차원에서 계산 가능합니다

1D-conv $[f * g](i) = \sum_{p=1}^d \boxed{f(p)} g(i + p)$



i, j, k 가 바뀌어도 커널 f 의 값은 바뀌지 않습니다

2D-conv $[f * g](i, j) = \sum_{p, q} \boxed{f(p, q)} g(i + p, j + q)$

3D-conv $[f * g](i, j, k) = \sum_{p, q, r} \boxed{f(p, q, r)} g(i + p, j + q, k + r)$

2차원 Convolution 연산 이해하기

- 2D-Conv 연산은 이와 달리 커널(kernel)을 입력벡터 상에서 움직여가면서 선형모델과 합성함수가 적용되는 구조입니다

$$[f * g](i, j) = \sum_{p, q} f(p, q) g(i + p, j + q)$$

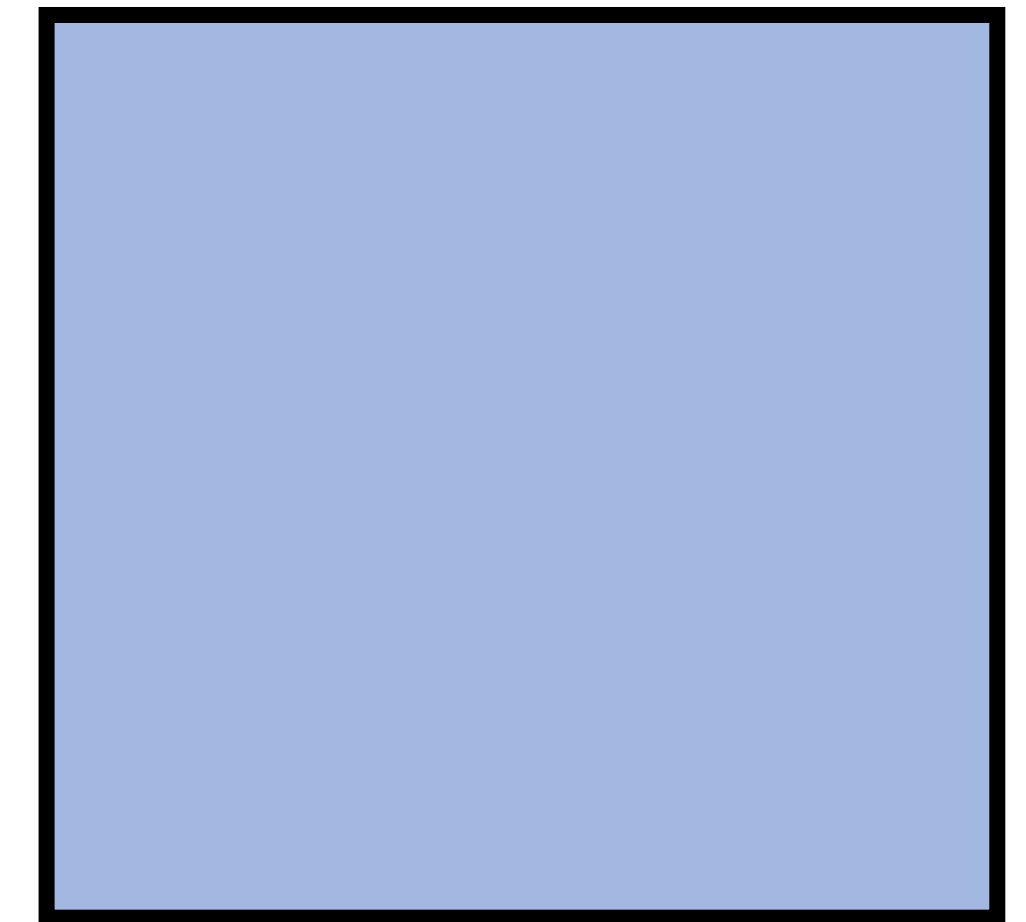
커널

입력

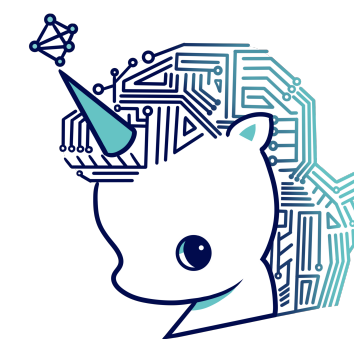
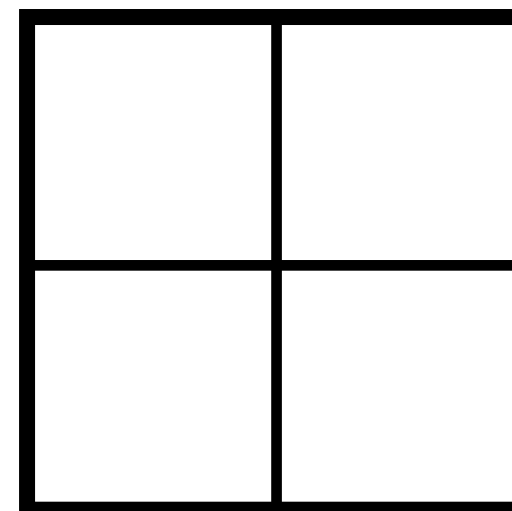


커널

*



입력



$$0 \times 0 + 1 \times 1 + 2 \times 3 + 3 \times 4 = 19$$

2차원 Convolution 연산 이해하기

- 2D-Conv 연산은 이와 달리 커널(kernel)을 입력벡터 상에서 움직여가면서 선형모델과 합성함수가 적용되는 구조입니다

$$[f * g](i, j) = \sum_{p, q} f(p, q) g(i + p, j + q)$$

커널 (green arrow pointing to $f(p, q)$)

입력 (green arrow pointing to $g(i + p, j + q)$)

0	1
2	3

커널

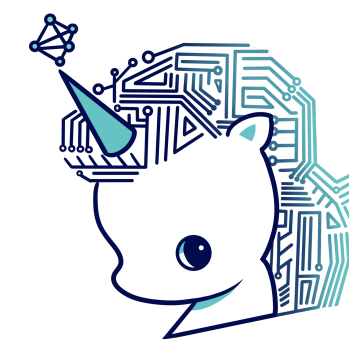


성분 곱

0	1	2
3	4	5

입력

19	



$$0 \times 0 + 1 \times 1 + 2 \times 3 + 3 \times 4 = 19$$

2차원 Convolution 연산 이해하기

- 2D-Conv 연산은 이와 달리 커널(kernel)을 입력벡터 상에서 움직여가면서 선형모델과 합성함수가 적용되는 구조입니다

$$[f * g](i, j) = \sum_{p, q} f(p, q) g(i + p, j + q)$$

커널 (green arrow pointing to $f(p, q)$)
입력 (green arrow pointing to $g(i + p, j + q)$)

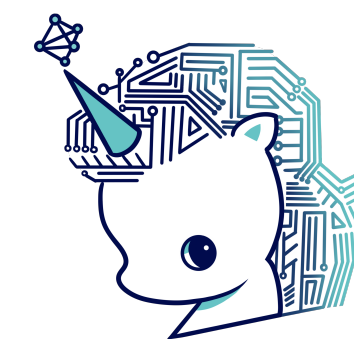
0	1
2	3

커널
(변화 x)

0	1	2
3	4	5

입력

19	25



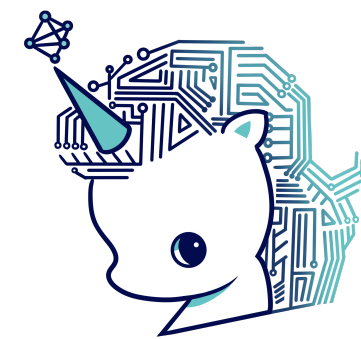
$$0 \times 1 + 1 \times 2 + 2 \times 4 + 3 \times 5 = 25$$

2차원 Convolution 연산 이해하기

- 2D-Conv 연산은 이와 달리 커널(kernel)을 입력벡터 상에서 움직여가면서 선형모델과 합성함수가 적용되는 구조입니다
- 입력 크기를 (H, W) , 커널 크기를 (K_H, K_W) , 출력 크기를 (O_H, O_W) 라 하면 출력 크기는 다음과 같이 계산합니다

$$O_H = H - K_H + 1$$

$$O_W = W - K_W + 1$$

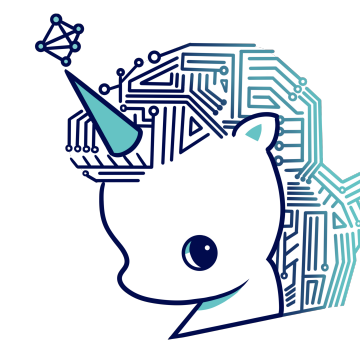
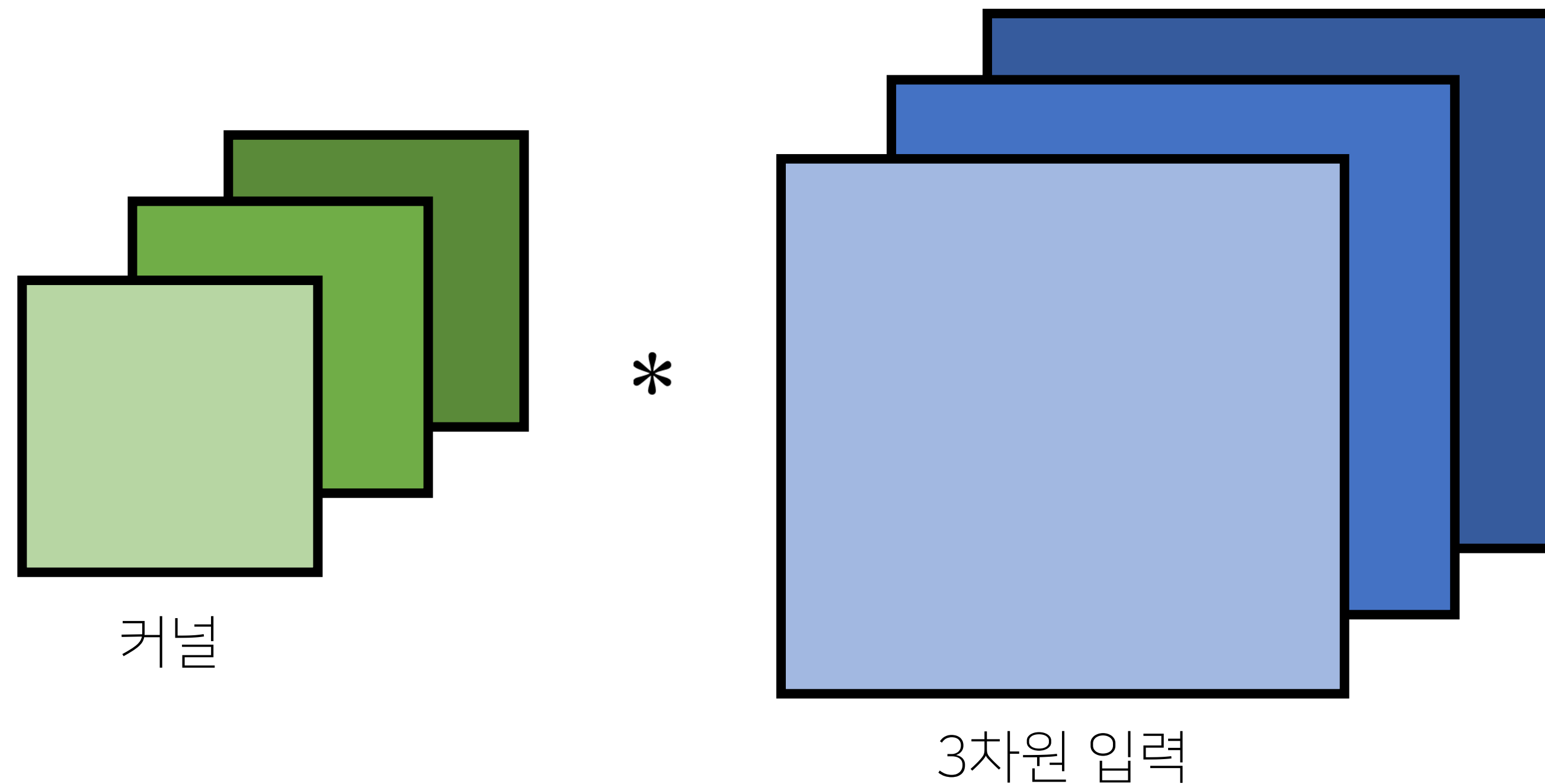


가령 28x28 입력을 3x3 커널로
2D-Conv 연산을 하면 26x26 이 된다

$$(28 - 3 + 1)$$

2차원 Convolution 연산 이해하기

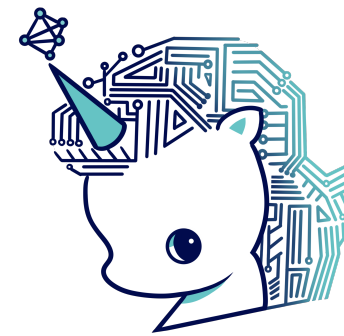
- 채널이 여러개인 2차원 입력의 경우 2차원 Convolution 을 채널 개수만큼 적용한다고 생각하면 됩니다



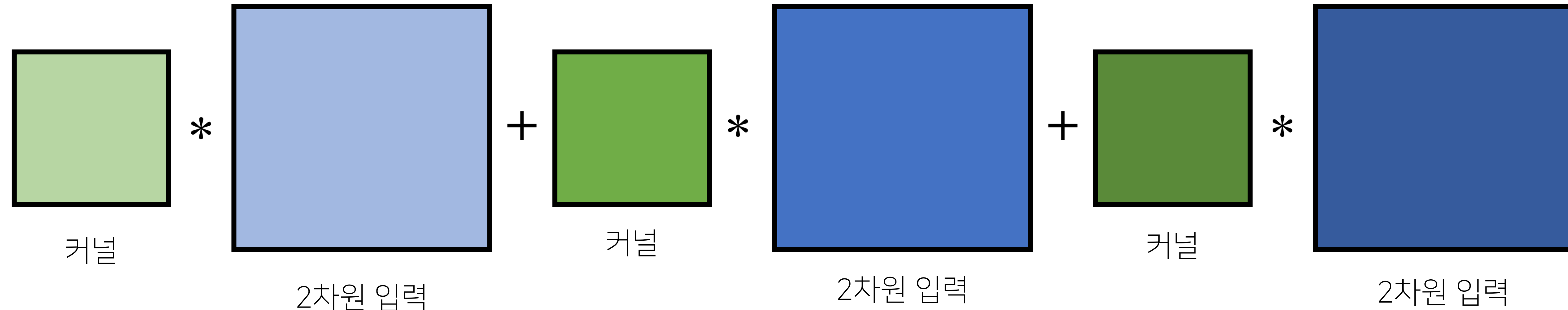
3차원부터는 행렬이 아닌 텐서라 부릅니다

2차원 Convolution 연산 이해하기

- 채널이 여러개인 2차원 입력의 경우 2차원 Convolution 을 채널 개수만큼 적용한다고 생각하면 됩니다

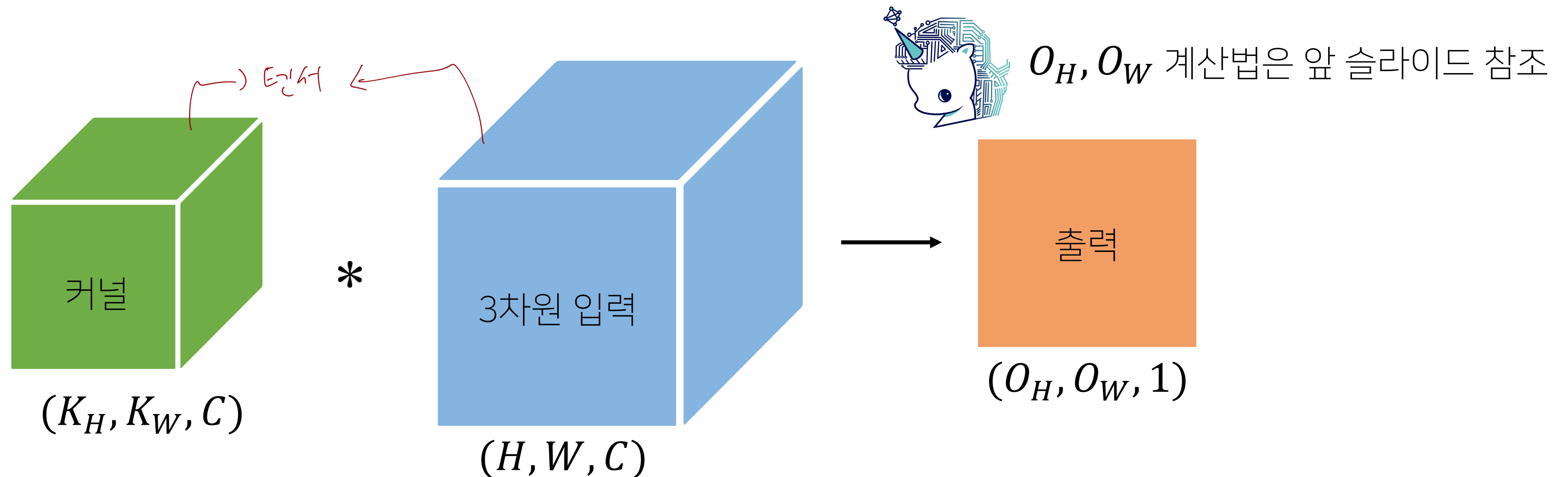


채널이 여러개인 경우 커널의 채널 수와
입력의 채널수가 같아야 합니다



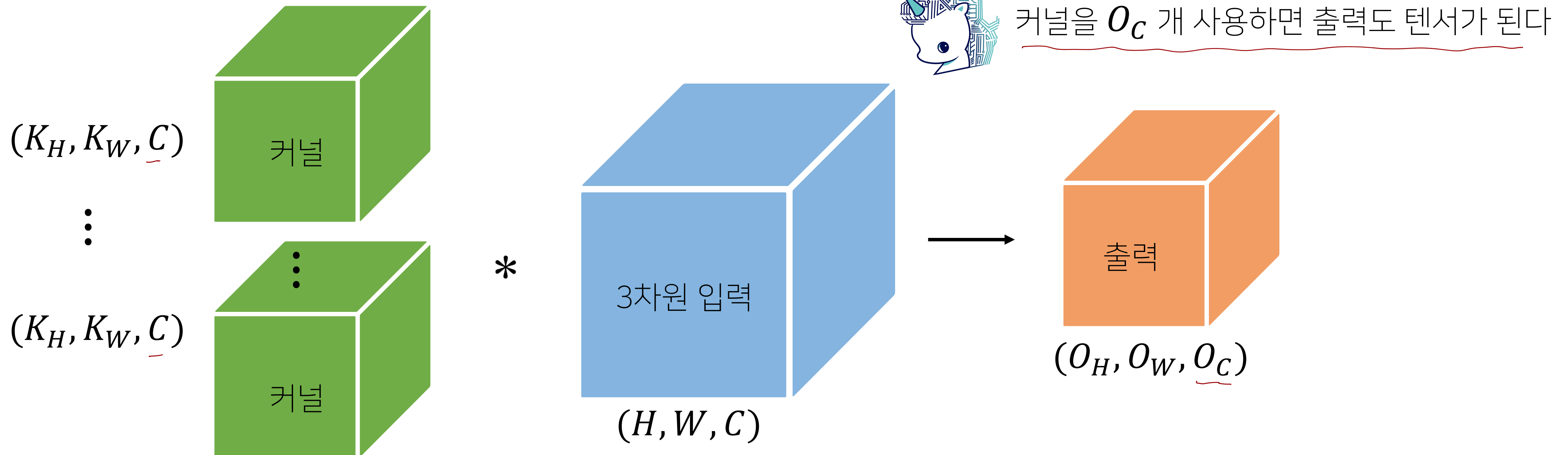
2차원 Convolution 연산 이해하기

- 채널이 여러개인 2차원 입력의 경우 2차원 Convolution 을 채널 개수만큼 적용한다고 생각하면 됩니다. 텐서를 직육면체 블록으로 이해하면 좀 더 이해하기 쉽습니다



2차원 Convolution 연산 이해하기

- 채널이 여러개인 2차원 입력의 경우 2차원 Convolution 을 채널 개수만큼 적용한다고 생각하면 됩니다. 텐서를 직육면체 블록으로 이해하면 좀 더 이해하기 쉽습니다

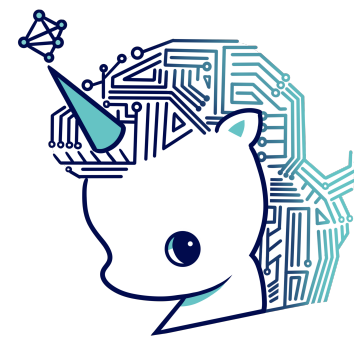


Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다

$$\begin{aligned}\frac{\partial}{\partial x}[f * g](x) &= \frac{\partial}{\partial x} \int_{\mathbb{R}^d} f(y)g(x-y)dy \\ &= \int_{\mathbb{R}^d} f(y) \frac{\partial g}{\partial x}(x-y)dy \\ &= [f * g'](x)\end{aligned}$$

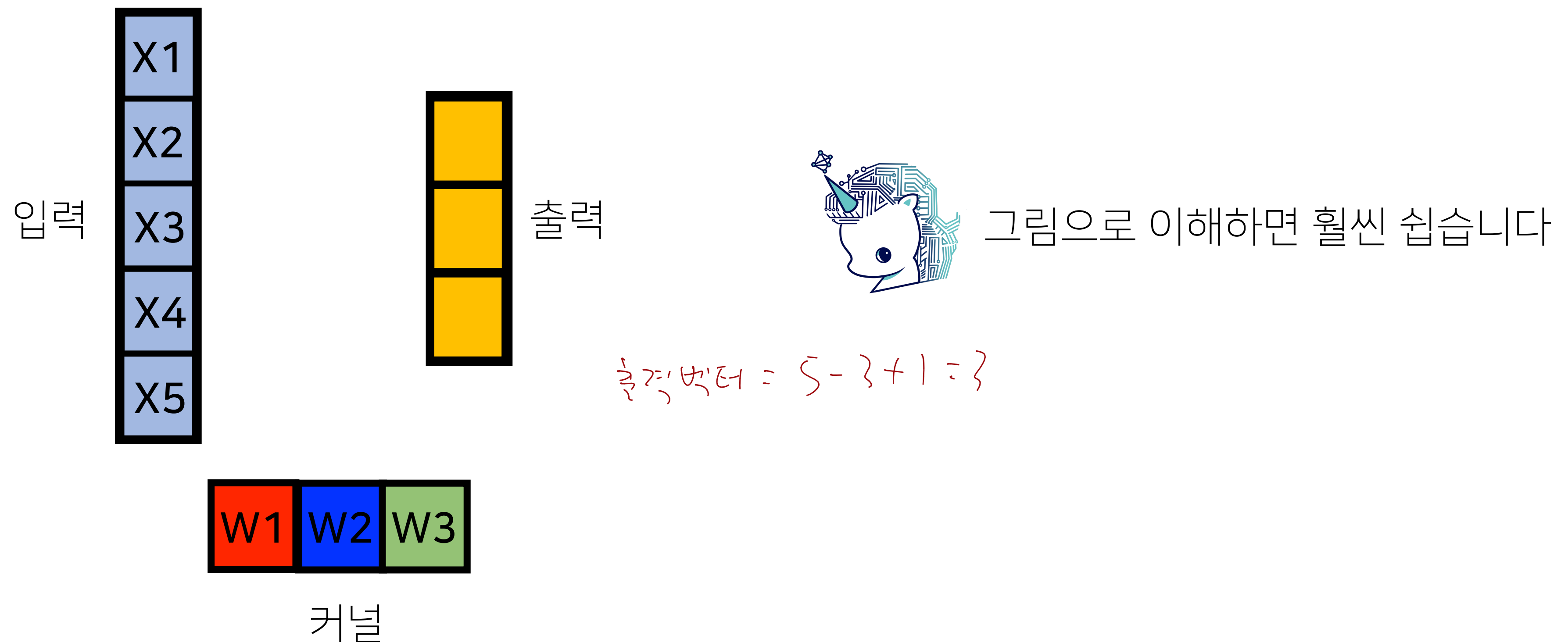
=> convolution 연산에 미분을 했을
convolution이 그대로 나온 것.



Discrete 일 때도 마찬가지로 성립한다

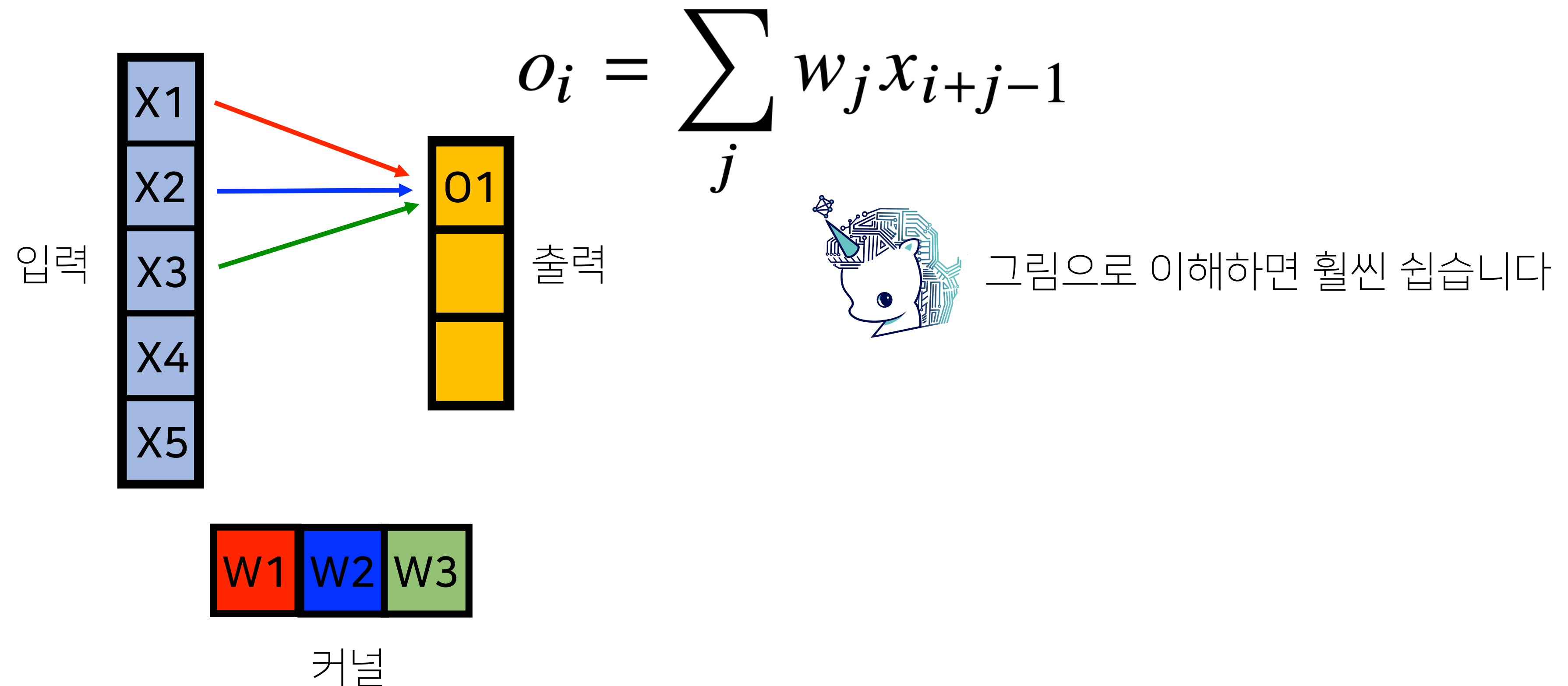
Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



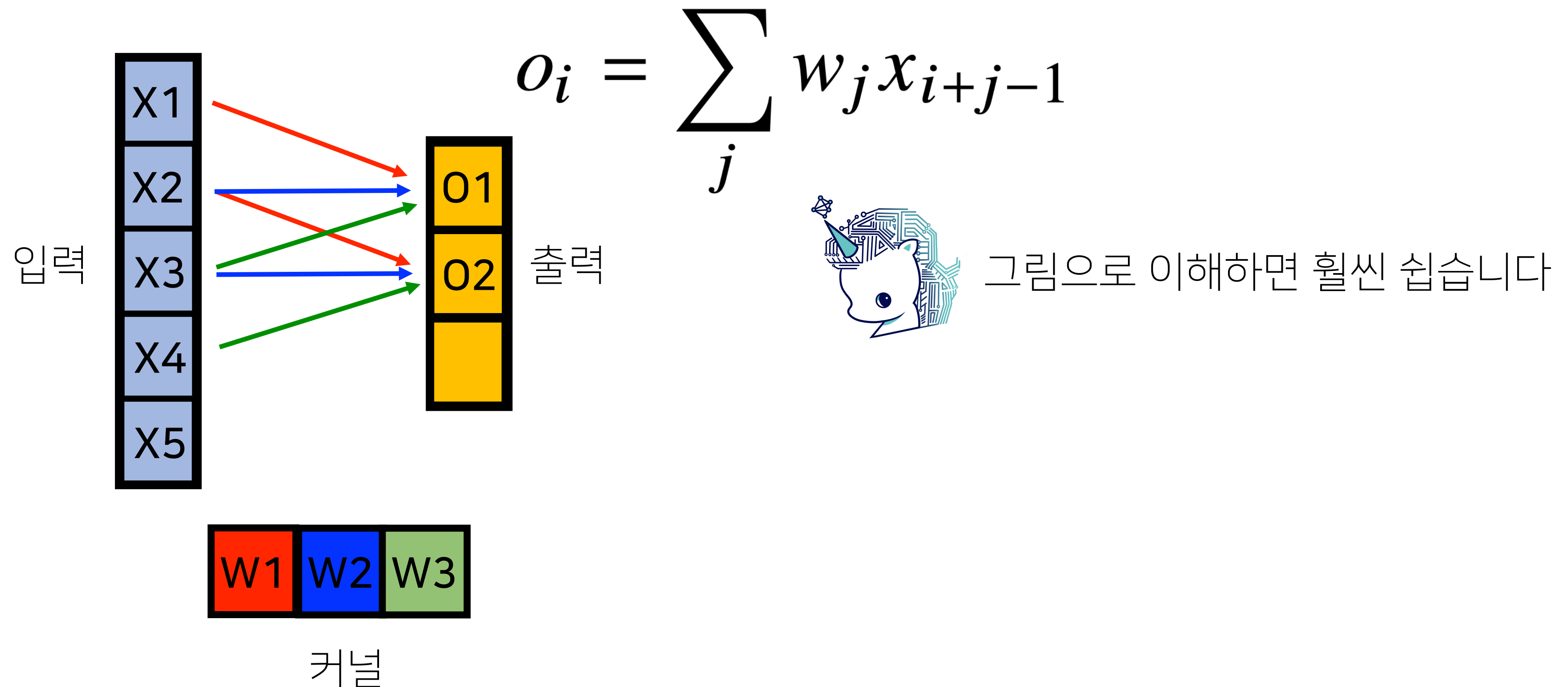
Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



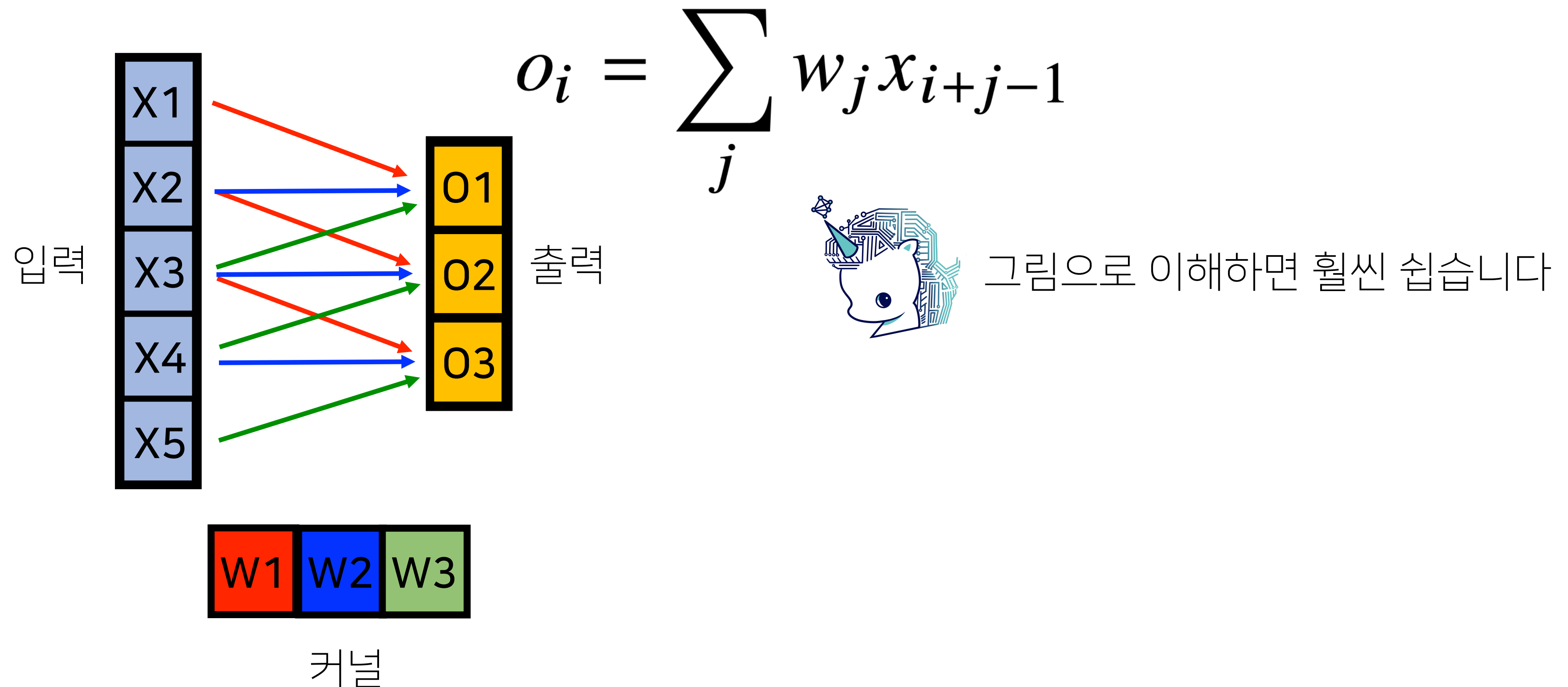
Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



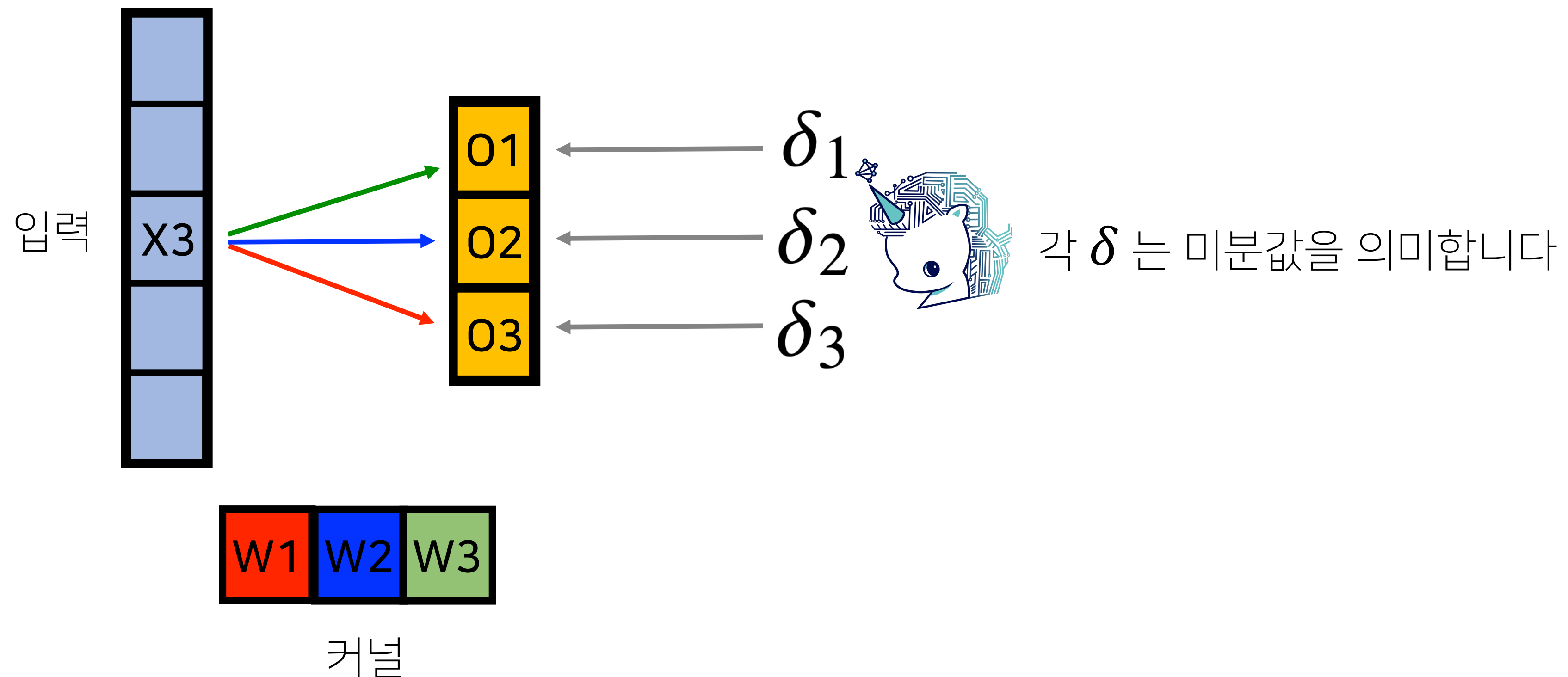
Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



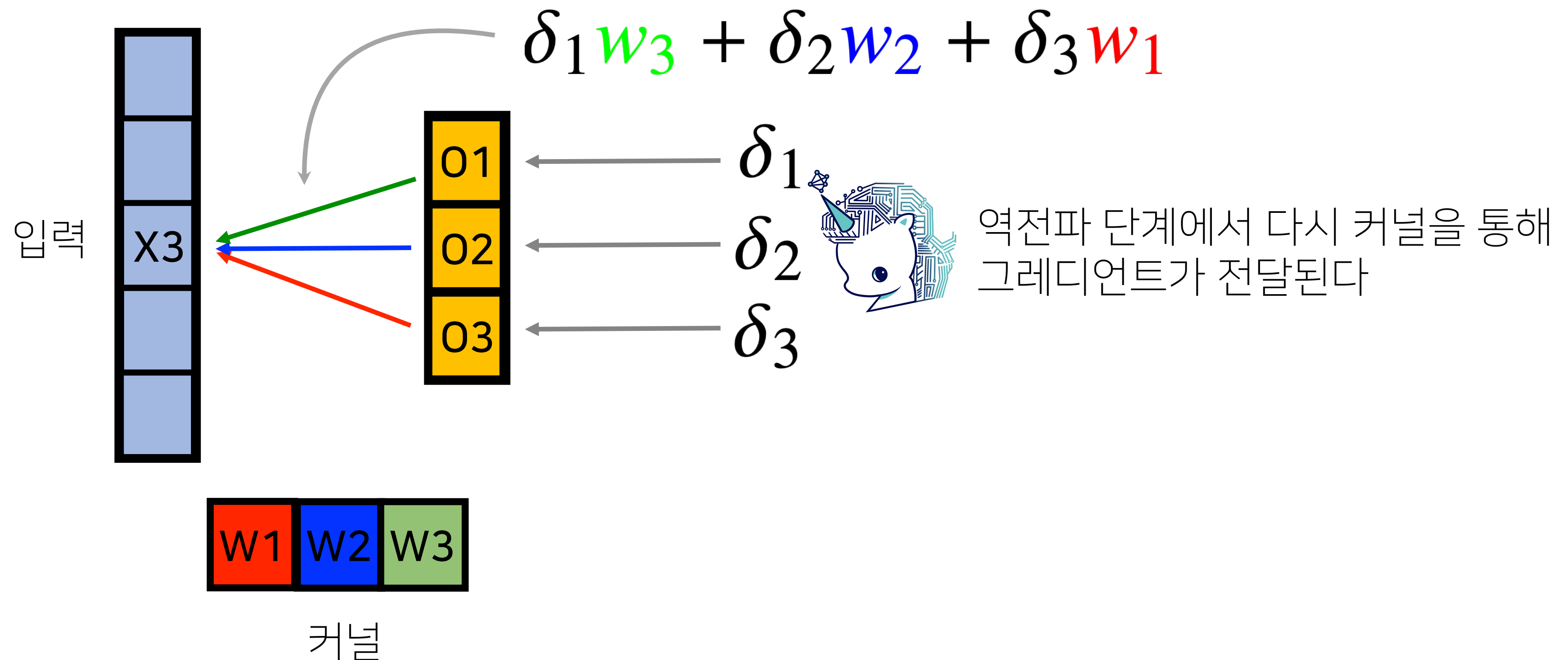
Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



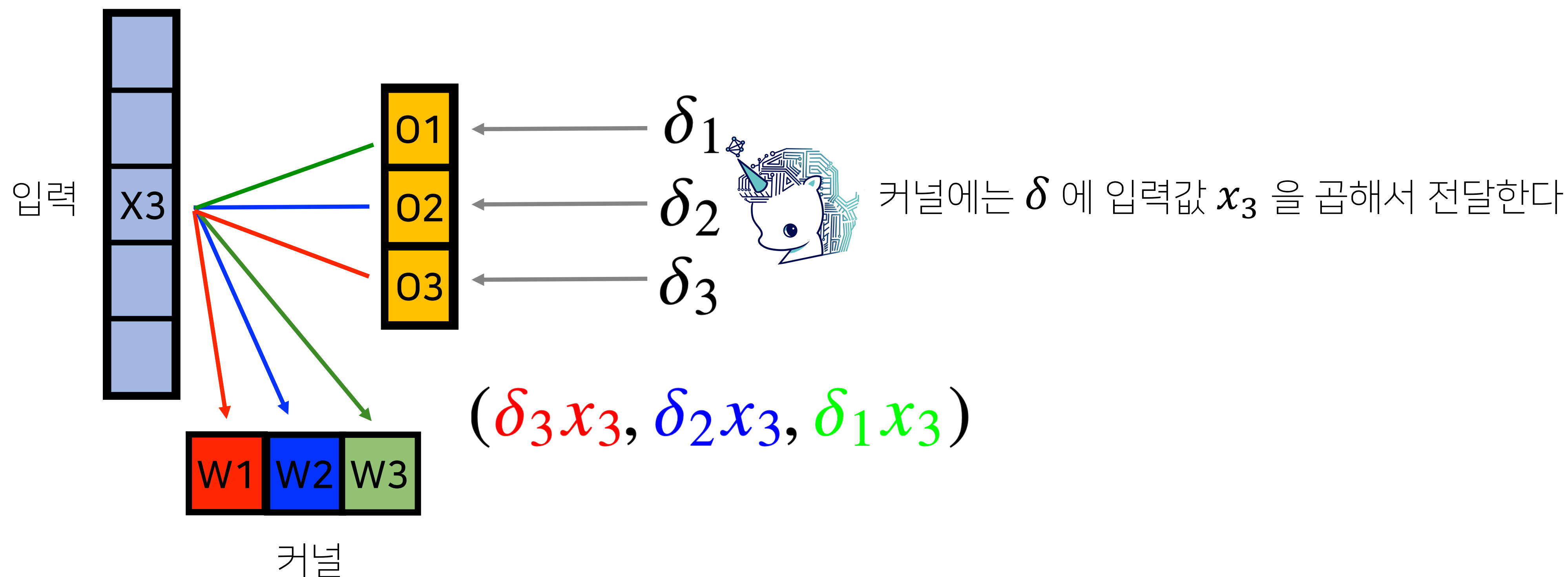
Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



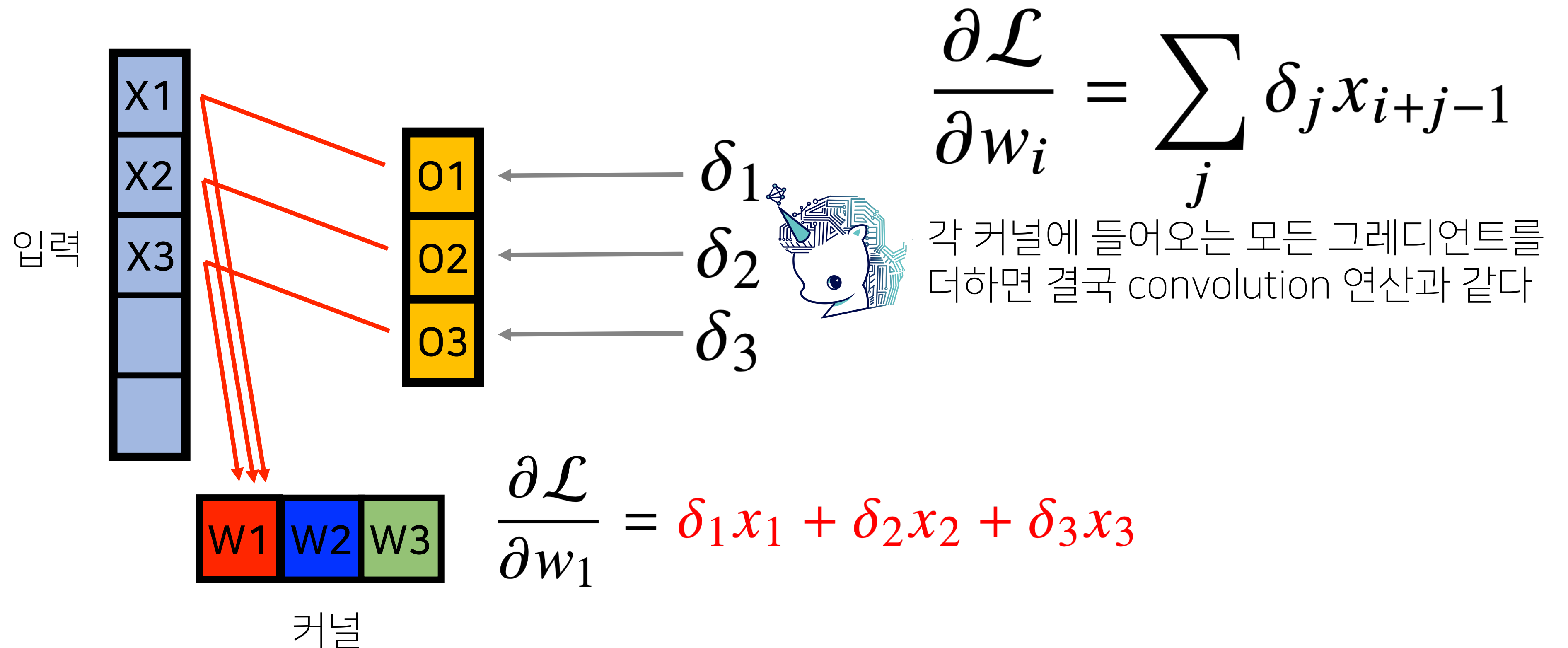
Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



Convolution 연산의 역전파 이해하기

- Convolution 연산은 커널이 모든 입력데이터에 공통으로 적용되기 때문에 역전파를 계산할 때도 convolution 연산이 나오게 됩니다



THE END

다음 시간에 보아요!