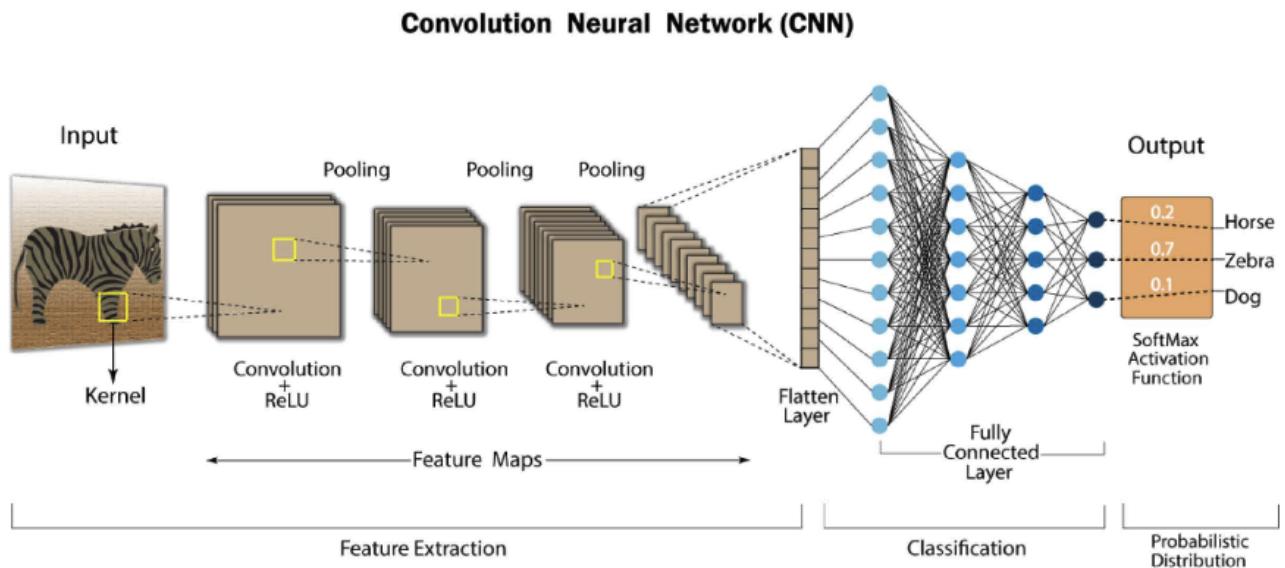


CNN

CNN: Convolutional Neural Network

이미지 데이터를 학습하고 인식하는 것에 특화된 알고리즘. 각 객체 및 형상을 분리하여 해석하는 메커니즘이다. 전체적인 이미지가 무엇인지 알려준다.

과제: CNN의 기본 구성 요소(필터를 통한 합성곱, 활성화 함수([Activation Function](#)), 풀링(Pooling), 완전연결층([Fully Connected Layer](#))에 대한 설명과 이들이 어떻게 상호 작용해서 이미지에 대한 학습을 진행하는지 그림과 함께 설명해주세요.

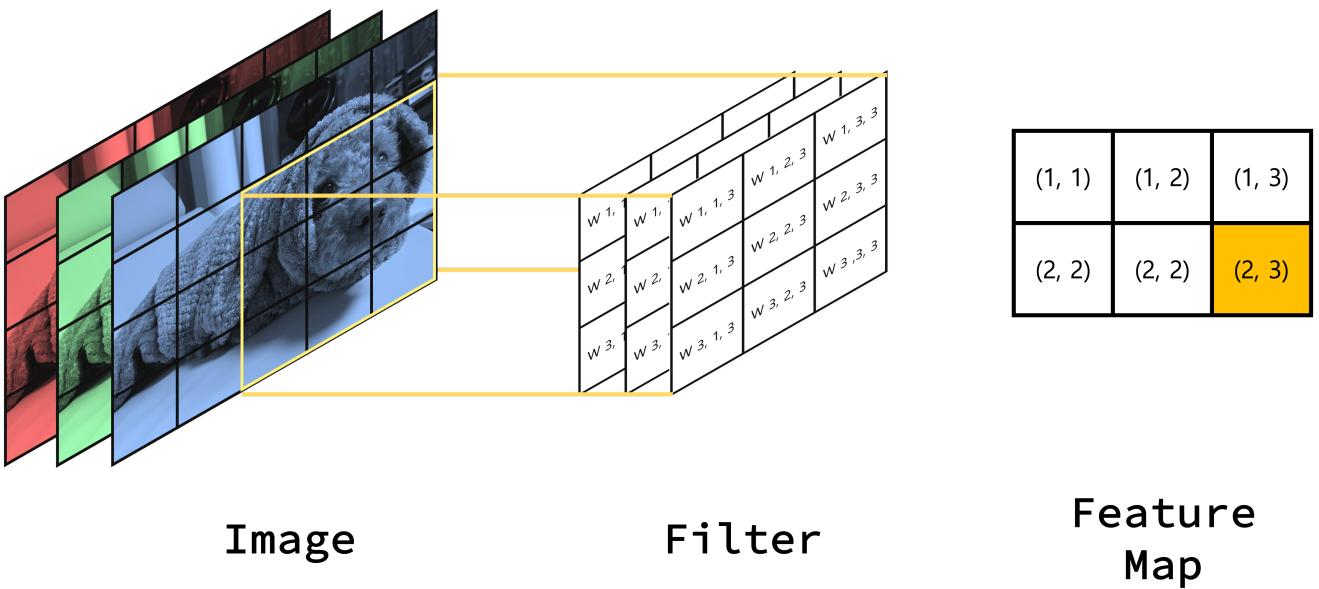
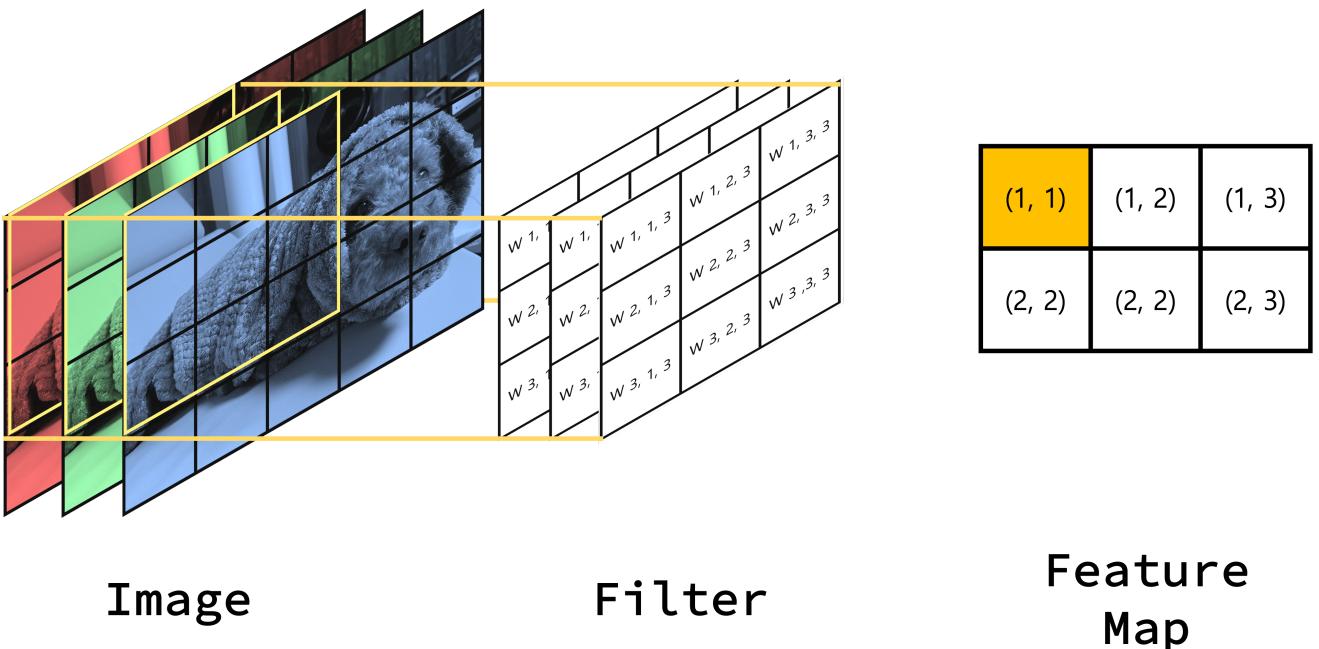


CNN은 이미지의 특징을 추출하는 부분(Feature extraction)과 클래스를 분류하는 부분(Classification)으로 나눌 수 있다.

- Feature Extraction: **Convolution Layer**와 **Pooling Layer**를 여러 겹 쌓는 형태로 구성된다. 이미지를 형상으로 분할하고 분석한다.
- Classification: CNN 마지막 부분에는 이미지 분류를 위한 [Fully Connected Layer](#)가 추가된다.

Convolution Layer (합성곱 층)

이미지의 특성 추출에 사용되는 Layer로, 입력되는 데이터에 대해 합성곱 연산을 수행한다. 작은 Filter를 이용하여 이미지를 훑으며 이미지와 필터의 합성곱을 계산한다. Filter는 공간적인 관계를 고려하여 특정한 패턴이나 특징을 감지하도록 설계되어, Filter를 통한 합성곱 과정은 입력 이미지의 특성을 추출한다.



Stride

위의 예시에서 Stride는 1로 설정되었다. 즉, Stride는 Filter의 이동량을 의미한다.

Zero Padding

합성곱 과정에서 가장자리 픽셀들의 정보는 점점 사라지게 된다. 가장자리 픽셀에 대해 합성곱 Filter가 적게 훑기 때문이다. 이 문제를 해결하기 위해 이미지 가장자리에 Zero Padding, 특정 값으로 설정된 픽셀들을 가장자리에 추가한다.

위의 예시에서 4x5였던 입력 이미지가 2x3으로 줄어들었다. Zero Padding을 적용하면 이미지 손실을 줄일 수 있다. 아래 그림에서, Zero Padding을 적용하지 않은 경우 출력 이미지는 2x2로 줄어들었지만,

Zero Padding을 적용한 경우에는 손실이 일어나지 않았다.

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)



(1, 1)	(1, 2)
(2, 1)	(2, 2)

0	0	0	0	0	0
0	(1, 1)	(1, 2)	(1, 3)	(1, 4)	0
0	(2, 1)	(2, 2)	(2, 3)	(2, 4)	0
0	(3, 1)	(3, 2)	(3, 3)	(3, 4)	0
0	(4, 1)	(4, 2)	(4, 3)	(4, 4)	0
0	0	0	0	0	0

Without Zero-Padding

1, 1	1, 2	1, 3
2, 1	2, 2	2, 3
3, 1	3, 2	3, 3



1, 1	1, 2	1, 3
2, 1	2, 2	2, 3
3, 1	3, 2	3, 3

With Zero-Padding

Activation Function

Filter를 통해 얻은 Feature Map에 활성화 함수(보통 ReLu 함수)를 적용한다. Feature Map에는 특징이 있으면 큰 값, 없으면 0에 가까운 값이 픽셀에 담긴다. 이 값들은 연속적이기 때문에 특징이 있다/없다의 비선형적인 값들로 바꿔주어야 하며, 이 역할을 수행하는 것이 활성화 함수이다.

Pooling Layer

그러나 이미지 크기를 유지하면 연산량이 늘어나기 때문에 추출한 Feature는 적절히 강조하고 크기를 줄여야 하는데, 이 역할을 Pooling Layer가 수행한다.

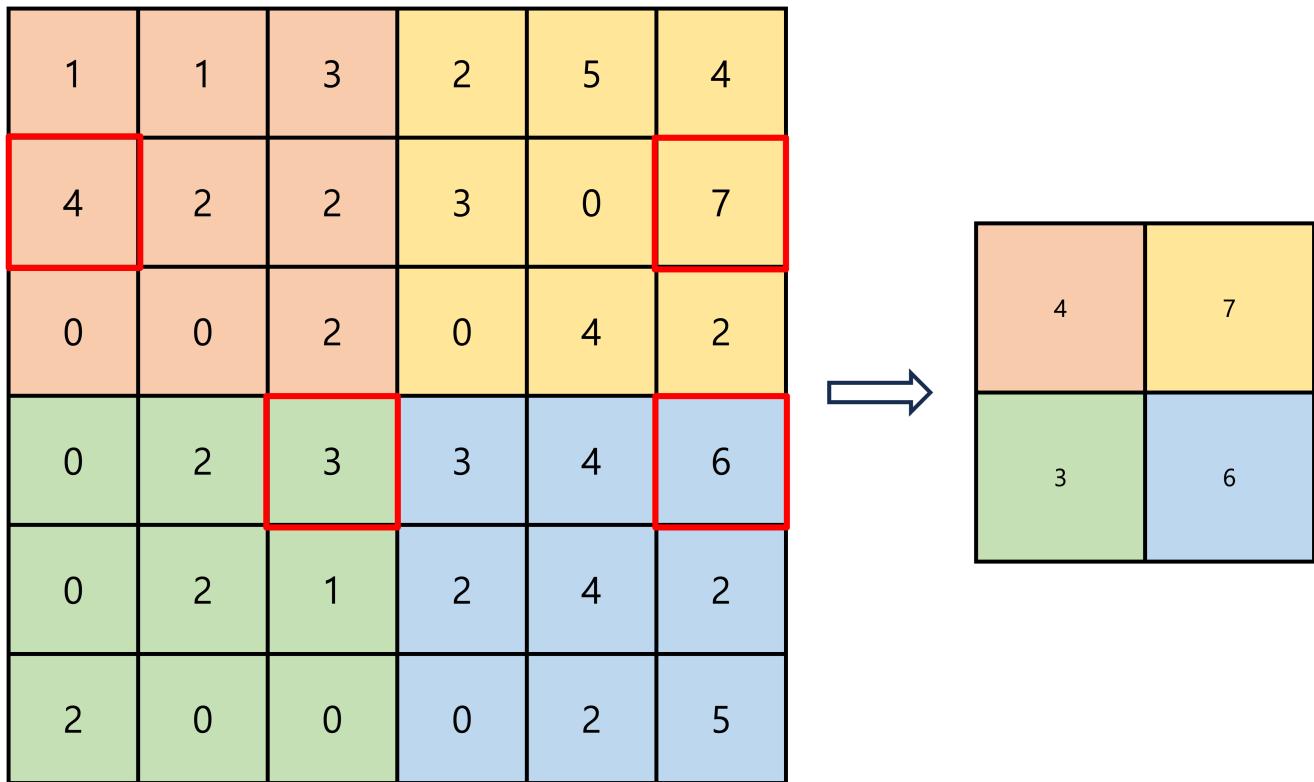
주로 Max Pooling을 사용하지만 Average Pooling, L2-norm Pooling 등 다양한 풀링 방법이 존재한다.

추출된 모든 특징을 사용할 필요가 없기 때문에 Feature Map를 인위적으로 줄인다.

Pooling Layer는 항상 적용하는 것이 아니라 이미지 크기를 줄이고 싶을 때 선택적으로 적용한다.

Max Pooling

정해진 크기 안에서 가장 큰 값을 뽑아내는 방식으로 Feature Map를 줄인다.



Max Pooling

위의 그림은 Stride = 3일 때 3x3인 filter를 통해서 Max Pooling을 적용한 것이다. 같은 색의 상자 안에서 가장 큰 값을 뽑아내는 것이 Max Pooling이다.

Average Pooling은 가장 큰 값이 아니라 평균을 구하는 것이다.

Fully Connected Layer

최종 도출된 Feature Map들을 Flatten하여 Fully Connected Layer에 집어넣는다. FC Layer의 입력값은 공간적 연계성을 더이상 갖지 않으며, 귀가 있는지? 털이 있는지? 등의 Feature를 갖는다.

Fully Connected Layer

한 layer의 모든 뉴런들이 다음 layer의 모든 뉴런과 연결된 상태를 의미한다.

목적: 1차원 배열의 형태로 flatten된 행렬을 통해 이미지를 분류하는데 사용된다.

이미지 2차원 벡터의 행렬을 1차원 배열로 flatten하는 것, Relu 함수로 뉴런을 활성화하는 것, Softmax 함수로 이미지를 분류하는 것까지 Fully Connected Layer이다.

이미지 데이터로 Fully Connected 신경망을 학습시켜야 할 때 3차원 이미지 데이터(컬러 이미지인 경우 3차원 데이터이다.)를 1차원으로 flatten해야 한다. 이 flatten 과정에서 공간 정보가 손실되어 학습이 비효율적이고 정확도를 높이는데 한계가 생긴다([DNN](#)의 문제점). 이미지의 공간 정보를 유지한 상태로 학습이 가능한 모델이 [CNN](#)이다.

Reference

- <https://dsbook.tistory.com/59>

- <https://rubber-tree.tistory.com/entry/%EB%94%A5%EB%9F%AC%EB%8B%9D-%EB%AA%A8%EB%8D%B8-CNN-Convolutional-Neural-Network-%EC%84%A4%EB%AA%85>
- <https://blog.naver.com/intelliz/221709190464>
- <https://mijeongban.medium.com/%EB%94%A5%EB%9F%AC%EB%8B%9D-%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D-cnn-convolutional-neural-networks-%EC%89%BD%EA%B2%8C-%EC%9D%B4%ED%95%B4%ED%95%98%EA%B8%B0-836869f88375>
- <https://mvje.tistory.com/128> -> Convolution Layer 의미
- <https://cding.tistory.com/5> -> 꽤 자세함