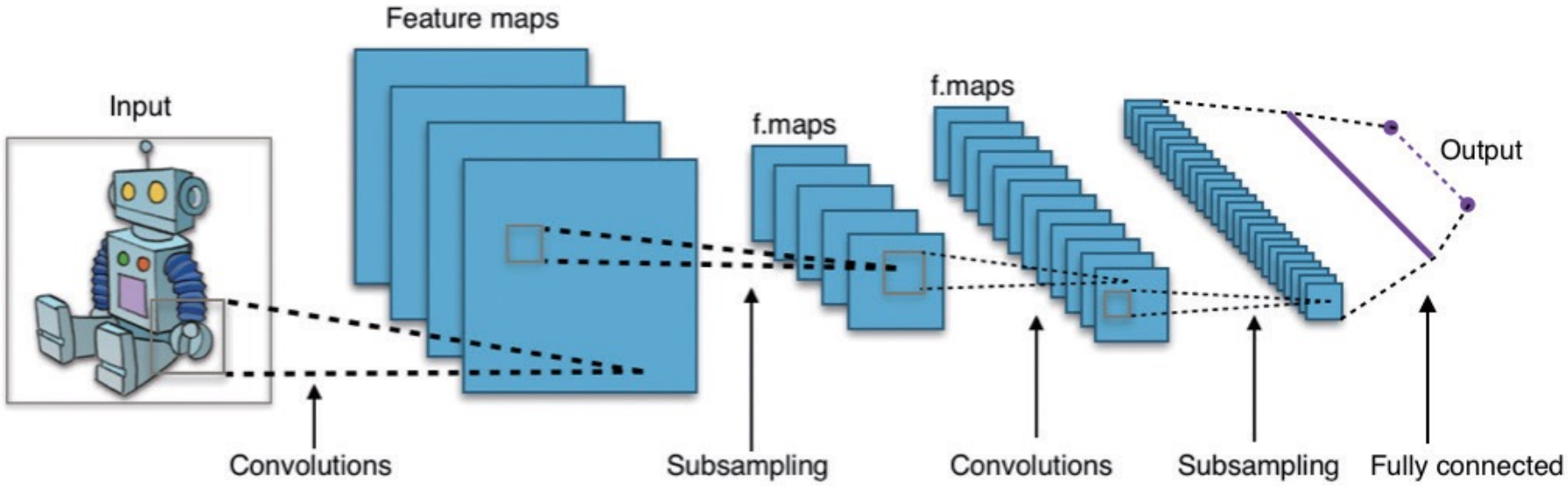
강의 목표

1. 컨볼루션 신경망(CNN)의 구조와 개념을 이해합니다.
2. TensorFlow 2.0을 이용해서 CNN 구조를 구현해봅니다.

컨볼루션 신경망(CNN)

- **컨볼루션 신경망**Convolutional Neural Networks(CNN)은 이미지 분야를 다루기에 최적화된 인공신경망 구조입니다.
- 컨볼루션 신경망은 크게 **컨볼루션층**Convolution Layer과 **풀링층**Pooling(Subsampling) Layer으로 구성되어 있습니다. **풀링**Pooling은 **서브샘플링**Subsampling이라고도 불립니다.

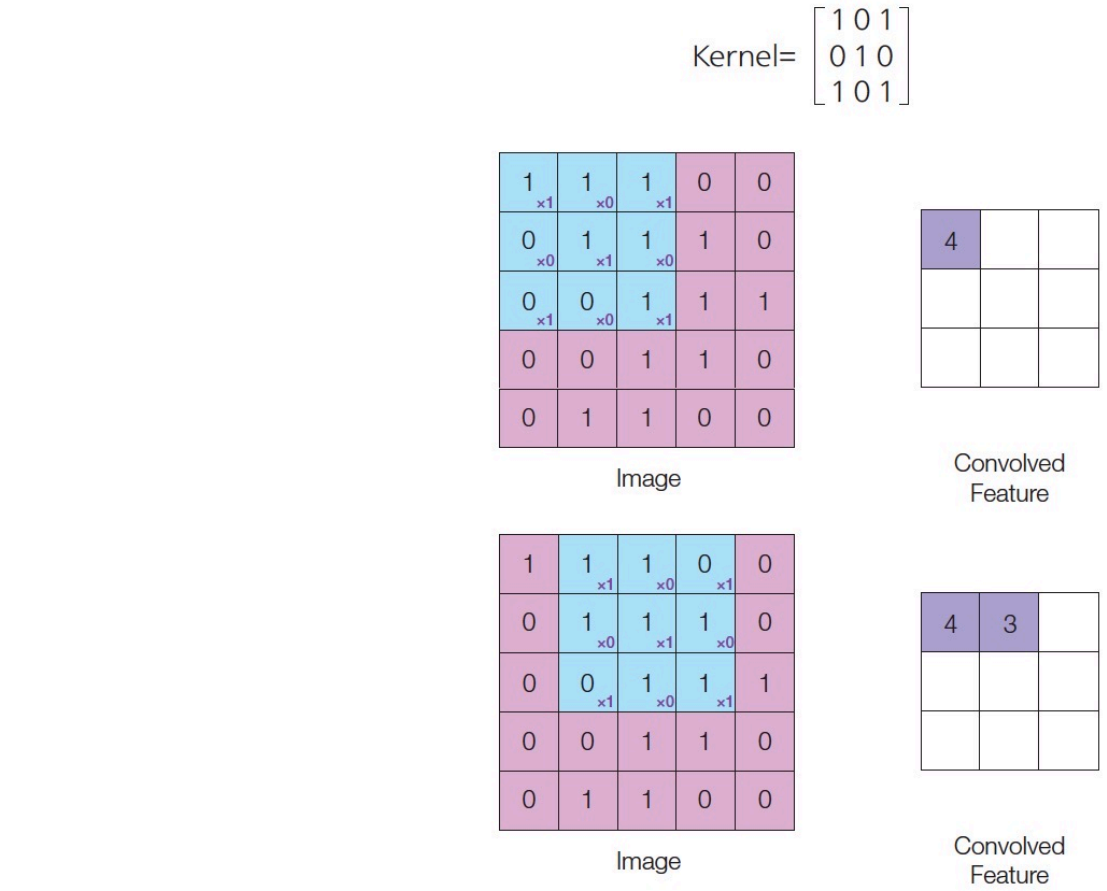


컨볼루션과 풀링

- **컨볼루션층**Convolution Layer에 대해 알아봅시다. 컨볼루션층은 컨볼루션 연산을 통해서 이미지의 특징을 추출해내는 역할을 합니다.
- 먼저 **컨볼루션**Convolutoin이라는 수학 연산자의 동작 과정을 이해해봅시다.
- 컨볼루션은 우리말로 **합성곱**이라고도 불리는데 **커널**Kernel 또는 **필터**Filter라고 불리는 윈도우 크기Window Size만큼의 X×X 크기의 행렬을 Y×Y 크기의 이미지 행렬의 X×X 크기 부분과 곱해서 모두 더하는 수학 연산자입니다.
- 행렬곱의 결과로 이미지 행렬의 X×X 크기 부분의 값들은 모두 더해져 하나의 값으로 모아질 것입니다. 이런 X×X 크기의 행렬 곱셈을 Y×Y 크기의 이미지 행렬의 가장 왼쪽 위부터 가장 오른쪽 아래까지 순차적으로 수행하는 연산이 컨볼루션층에서 이루어지는 동작입니다.

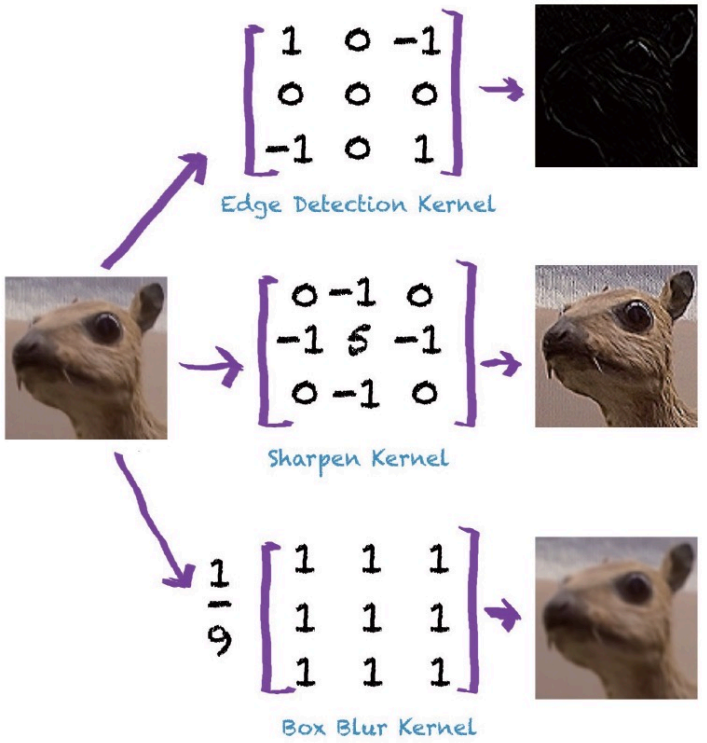
컨볼루션과 풀링

- 이제 그림을 통해서 컨볼루션층에서 이루어지는 동작을 직관적으로 이해해봅시다. 아래 그림은 3×3 크기의 커널을 이용해서 5×5 크기의 이미지 행렬에 컨볼루션 연산을 수행하는 과정을 단계별로 보여줍니다.



컨볼루션과 풀링

- 원본 이미지에 커널을 이용해서 컨볼루션을 수행하면 커널의 종류에 따라 **원본 이미지의 특징들** Features이 **활성화 맵** Activation Map으로 추출되게 됩니다.
- 이때 어떤 커널을 사용하느냐에 따라 원본 이미지에서 다양한 특징을 추출할 수 있습니다. 아래 그림은 원본 이미지에서 커널의 종류에 따라 다양한 특징이 추출되는 결과를 보여줍니다.



컨볼루션과 풀링

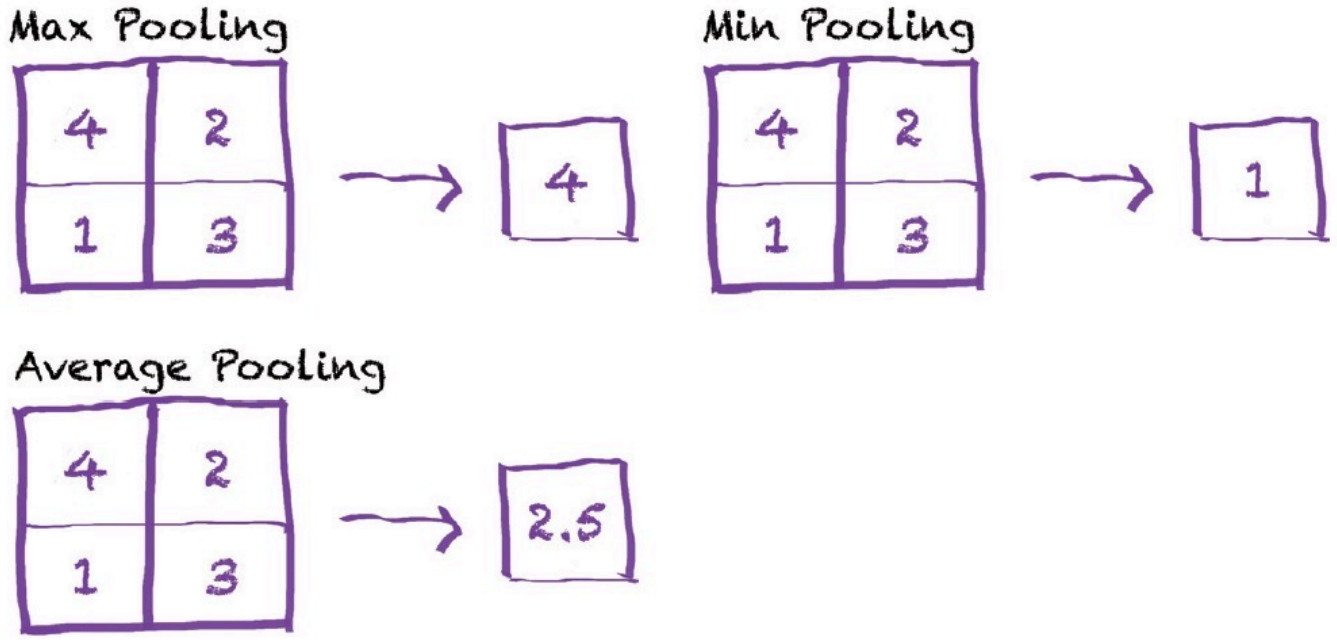
- 그림에서 볼 수 있듯이, 커널의 종류에 따라 원본 이미지의 모서리$_{Edge}$를 추출하거나 원본 이미지를 좀 더 명확한 이미지$_{Sharpen}$로 바꾸거나 원본 이미지를 흐리게$_{Blur}$ 만들 수 있습니다.
- 이렇게 추출한 **활성화 맵은 원본 이미지에서 명확히 드러나지 않던 특징들**을 보여줍니다. 예를 들어, 우리가 어떤 사진이 자동차인지 사람인지를 분류하고자 한다면, 원본 이미지 자체를 사용하는 것보다 모서리만 추출된 특징 이미지를 사용하는 것이 더 효율적일 것입니다. 좀 더 구체적으로 말하면, 모서리가 추출된 이미지를 통해 분류기를 학습하면 컴퓨터는 모서리가 각진 형태면 자동차, 모서리가 둥근 형태면 사람이라고 손쉽게 구분할 수 있을 것입니다.

컨볼루션과 풀링

- 이제 CNN을 구성하는 2번째 요소인 **풀링**Pooling에 대해 알아봅시다. 풀링층은 차원을 축소하는 연산을 수행합니다.
- 풀링의 종류에는 **최대값 풀링**Max Pooling, **평균값 풀링**Average Pooling, **최소값 풀링**Min Pooling이 있습니다.
- 최대값 풀링은 이미지의 X×X 크기 부분에서 가장 큰 값Max Value 하나를 추출해서 원본 이미지의 X×X 개의 값을 1개의 값으로 축소합니다.
- 동일한 원리로 평균값 풀링, 최소값 풀링은 평균값, 최소값으로 축소합니다. 풀링층 역시 이미지의 좌측 상단에서 우측하단으로 순차적으로 전체 이미지에 대해 풀링을 수행합니다.

컨볼루션과 풀링

- 아래 그림은 2×2 크기의 윈도우를 이용한 풀링 수행 과정 예시를 보여줍니다.
- 풀링층은 크게 2가지 장점이 있습니다.
- **이미지의 차원을 축소함**으로써 필요한 연산량을 감소시킬 수 있고, **이미지의 가장 강한 특징만을 추출하는 특징 선별 효과**가 있습니다. 예를 들어, 모서리가 추출된 활성화 맵에서 최대값 풀링을 수행하면, 차원은 축소되고 흐릿하던 모서리 부분이 더욱 뚜렷해질 것입니다.



컨볼루션과 풀링

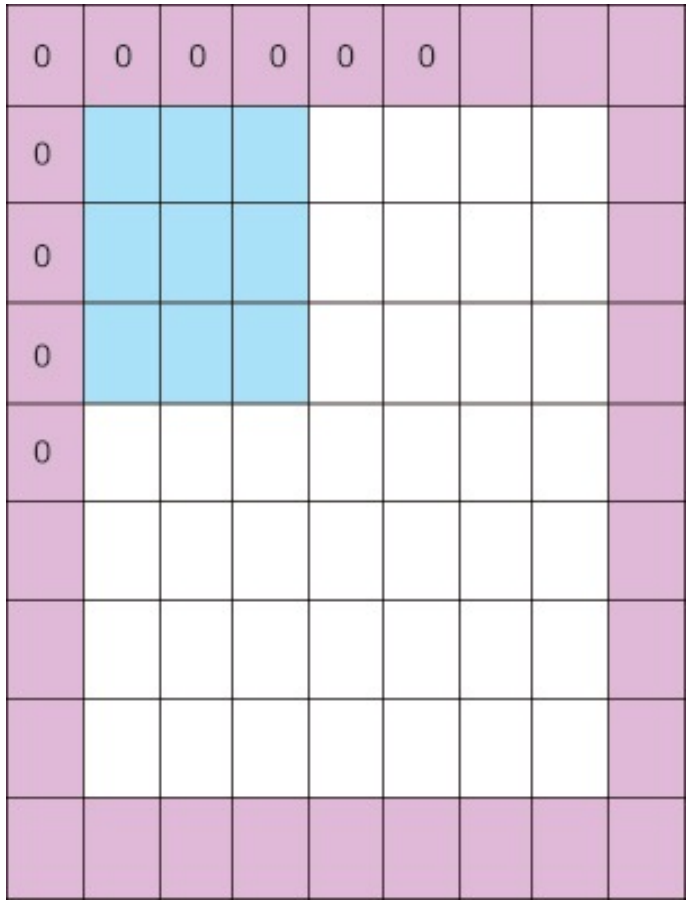
- 컨볼루션층을 거치면 인풋 이미지의 가로, 세로 차원이 축소되게 됩니다. 구체적으로 인풋 이미지의 가로 길이가 W_{in}이라면 컨볼루션층을 거친 출력 이미지의 가로 길이 W_{out}은 다음과 같이 계산됩니다.

$$W_{out} = \frac{W_{in} - F + 2P}{S} + 1$$

- 여기서 **F는 필터의 크기, S는 스트라이드를** 의미합니다.
- 스트라이드는 컨볼루션 연산시 건너뛰는 정도를 나타냅니다. 만약 스트라이드를 크게 잡아서 이미지를 성큼성큼 건너뛰어서 컨볼루션을 수행하면 차원이 많이 축소되고, 스트라이드를 작게 잡아서 이미지를 촘촘히 건너뛰면 차원이 조금 축소됩니다.
- 또한 $\frac{W_{in}-F}{S}$ 의 차원이 정수로 나누어떨어지지 않을 수도 있기 때문에 인풋 이미지의 상하좌우 모서리에 **P만큼 0을 채워 주는 제로패딩**Zero-Padding을 P만큼 적용해줍니다.

컨볼루션과 풀링

- 아래 그림은 제로패딩의 예시를 보여줍니다.



컨볼루션과 풀링

- 마찬가지로, 인풋 이미지의 세로 길이가 H_{in}이라면 컨볼루션층을 거친 출력 이미지의 세로 길이 H_{out}은 아래와 같이 계산됩니다.

$$H_{out} = \frac{H_{in} - F + 2P}{S} + 1$$

- 마지막으로, 컨볼루션층의 출력결과의 3번째 차원은 컨볼루션 필터 개수 K가 됩니다.
- 따라서 컨볼루션층의 결과로 출력되는 차원은 [W_{out}, H_{out}, K] 입니다.
- 예를 들어, [28×28×1] MNIST이미지에 4×4 크기의 필터(F=4)에 스트라이드가 2(S=2)이고, 제로 패딩을 1만큼 적용한(P=1), 64개의 필터개수(K=64)를 가진 컨볼루션층을 적용하면 출력 결과로 [14,14,64]. 즉 14×14 크기의 64개의 활성화맵이 추출될 것 입니다.

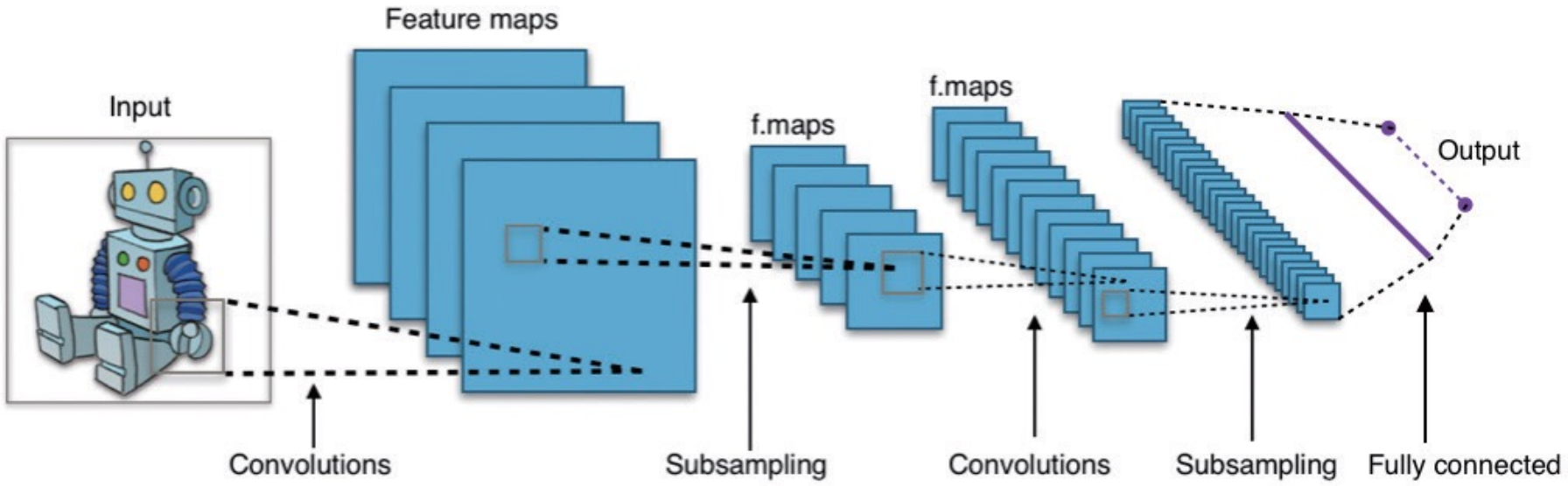
$$W_{out} = \frac{28 - 4 + 2 * 1}{2} + 1 = 14$$

$$H_{out} = \frac{28 - 4 + 2 * 1}{2} + 1 = 14$$

$$K = 64$$

컨볼루션 신경망(CNN)

- 분류 문제를 위한 CNN의 경우, 컨볼루션층과 풀링층을 거쳐서 추출된 활성화 맵들은 마지막에 Flattening으로 펼친 다음 우리가 배운 ANN 구조인 **완전 연결층**Fully Connected Layer의 인풋으로 들어가서 Softmax 분류를 수행하게 됩니다.



TensorFlow 2.0을 이용한 MNIST 숫자분류를 위한 CNN 구현

- MNIST 숫자분류를 위한 CNN을 TensorFlow 2.0 코드로 구현해봅시다.
- https://github.com/solaris33/deep-learning-tensorflow-book-code/blob/master/Ch07-CNN/mnist_classification_using_cnn_v2_keras.py

Chapter 5 - 인공신경망(Artificial Neural Networks) - ANN

- ANN을 이용한 MNIST 숫자분류기 구현 (Code) (TF v2 Code) (TF v2 Keras Code)

Chapter 6 - 오토인코더(Autoencoder)

- 오토인코더를 이용한 MNIST 데이터 재구축 (Code) (TF v2 Code) (TF v2 Keras Code)
- 오토인코더 + 소프트맥스 회귀를 이용한 MNIST 숫자 분류기 (Code) (TF v2 Code) (TF v2 Keras Code)

Chapter 7 - Convolutional Neural Networks(CNN)

- CNN을 이용한 MNIST 숫자 분류기 구현 (Code) (TF v2 Code) (TF v2 Keras Code)
- CNN을 이용한 CIFAR-10 분류기 구현 (Code) (TF v2 Code) (TF v2 Keras Code)
- tf.train.Saver API를 이용해서 모델과 파라미터를 저장(Save)하고 불러오기(Restore) (Code) (TF v2 Code) (TF v2 Keras Code)

Thank you!