2020531001 윤정인

인공지능

CHAP 15 – 신경망4 (CNN)

Contents

신경망4 (CNN)

01

영상 인식이란?

영상 인식

영상 안의 물체를 인식하거나 분류하는 것

CHAP 15에서는 컨볼루션 신경망(CNN)을 이용한 영상 인식 방법을 학습

영상 분류기

- 고양이 : 83%
- 강아지 : 10%
- 모피 : 5%
- …

02

전통적인 영상 인식 시스템의 구조

전통적인 영상 인식 시스템의 구조

전처리
(Preprocessing)

특징 추출기

특징 추출
(Feature
Extraction)

분류기

분류
(Classification)

강아지

고양이

카메라가
동물의 영상을 캡처

잡음 제거
동물을 배경에서 분리

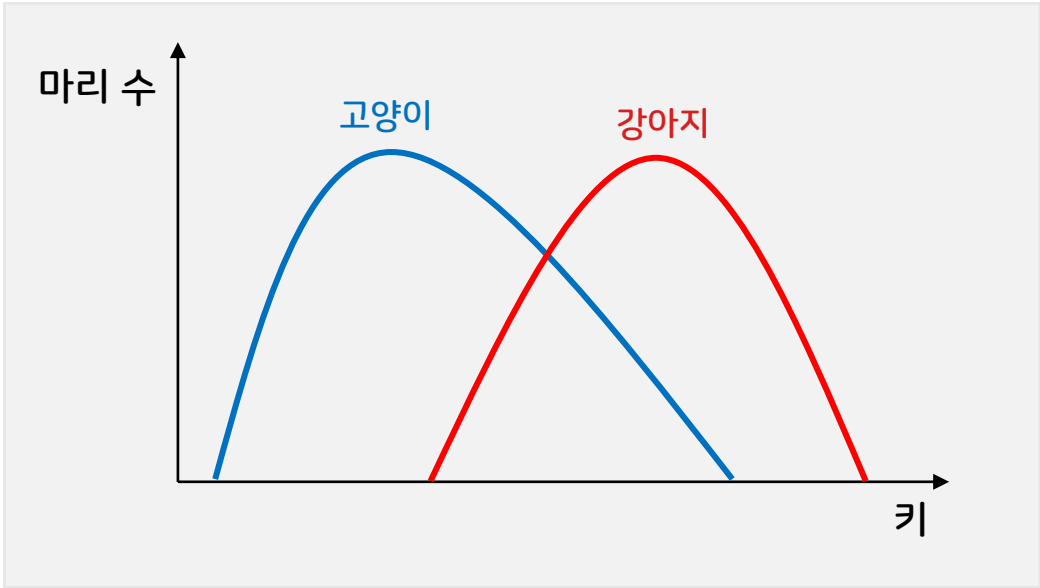
동물들의 특징 값
추출

특징 값들이
분류기로 보내짐

최종판단

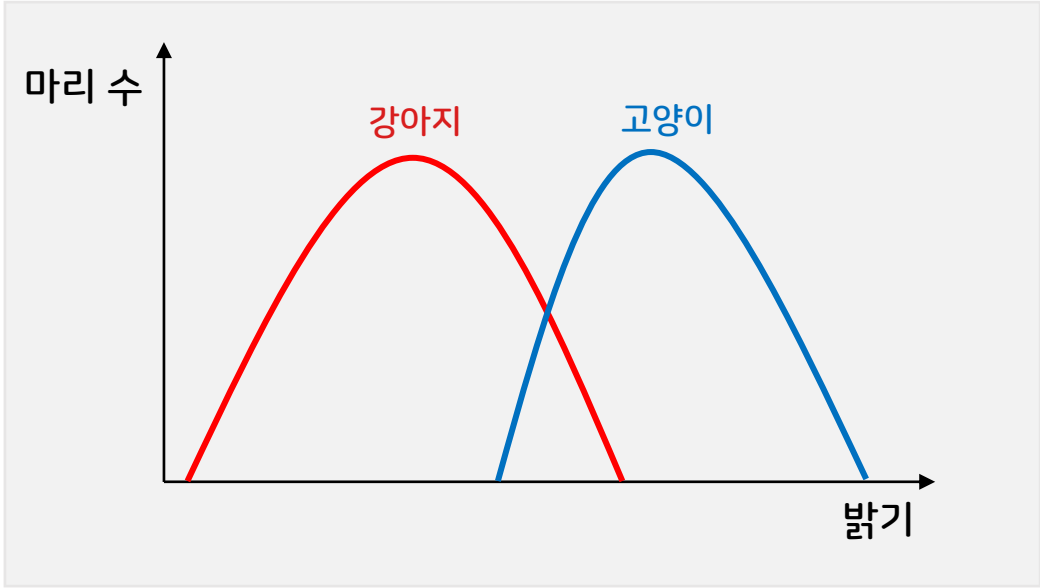
전통적인 영상 인식 시스템의 구조

[키를 특징으로 사용]



상대적으로 겹치는 부분이 많아 분류 신뢰성 ↓

[밝기를 특징으로 사용]

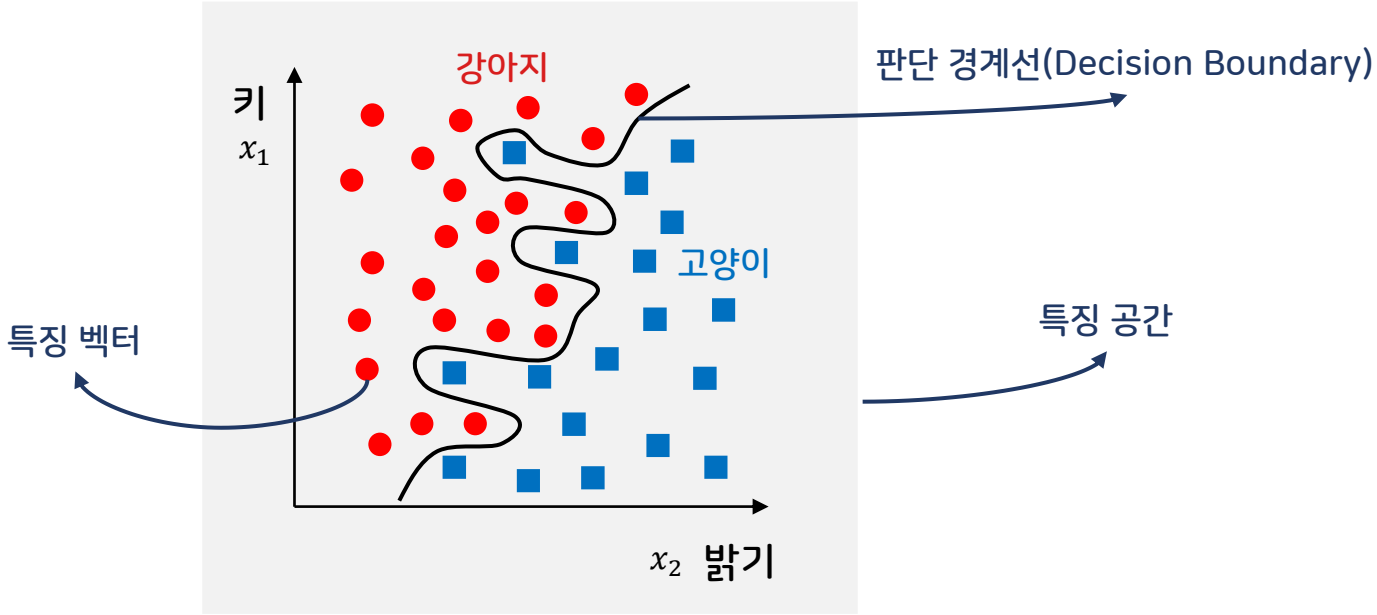


특징을 한가지 더해도 겹치는 부분이 존재

강아지와 고양이를 더 잘 구분할 수 있는 방법은?

전통적인 영상 인식 시스템의 구조

[키와 밝기를 특징으로 사용]



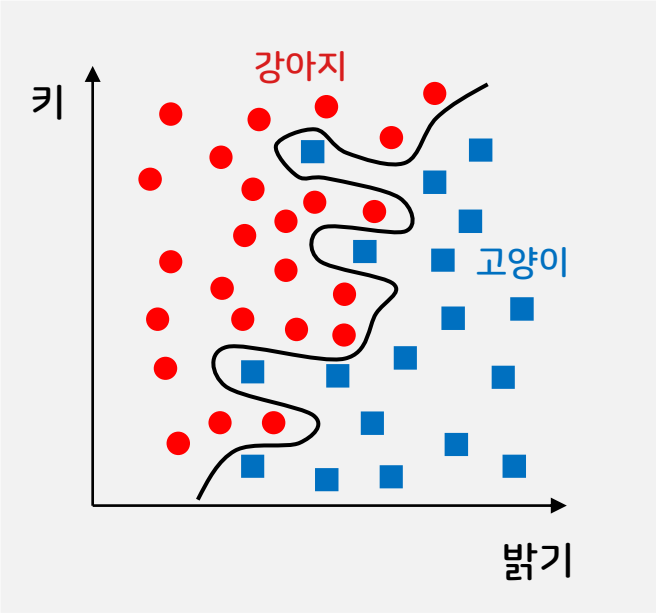
$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

특징 추출기는 입력 영상을 다음과 같은 특징 벡터(Feature Vector)로 변환

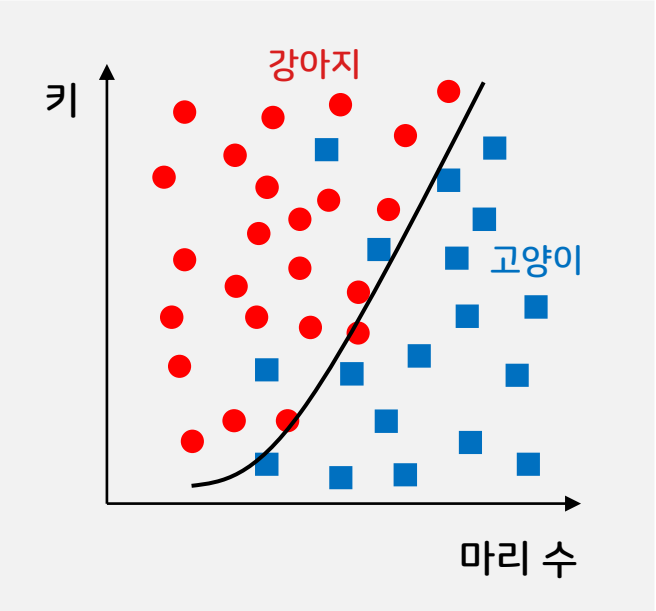
특징 벡터들이 존재하는 공간을 특징 공간(Feature Space)라고 함

특징 벡터가 판단경계선을 기준으로 오른쪽에 있으면 고양이, 왼쪽에 있으면 강아지로 분류

전통적인 영상 인식 시스템의 구조



[과잉적합]



[단순성 ↑]

판단경계선은 학습에서 사용한 샘플은 완벽하게 분류하지만 새로운 샘플에 대해서는 좋지 않은 성능을 나타낼 수 도 있음

성능과 분류기의 단순성을 적절하게 조합한 판단 경계선이 바람직할 수 도 있음

전통적인 영상 인식 시스템의 한계

이러한 방법의 문제점은 인간이 개입해야 한다는 점

특징을 선택하고 이미지에서 특징 값을 추출하는 과정은 인간이 해주어야 함

특징이 추출된 후에 분류하는 과정은 자동화가 가능

신경망을 연구하는 학자들은 특징의 선택과 추출도 학습을 연구

컨볼루션 신경망(Convolutional Neural Network :CNN)

03

컨볼루션 신경망(CNN)

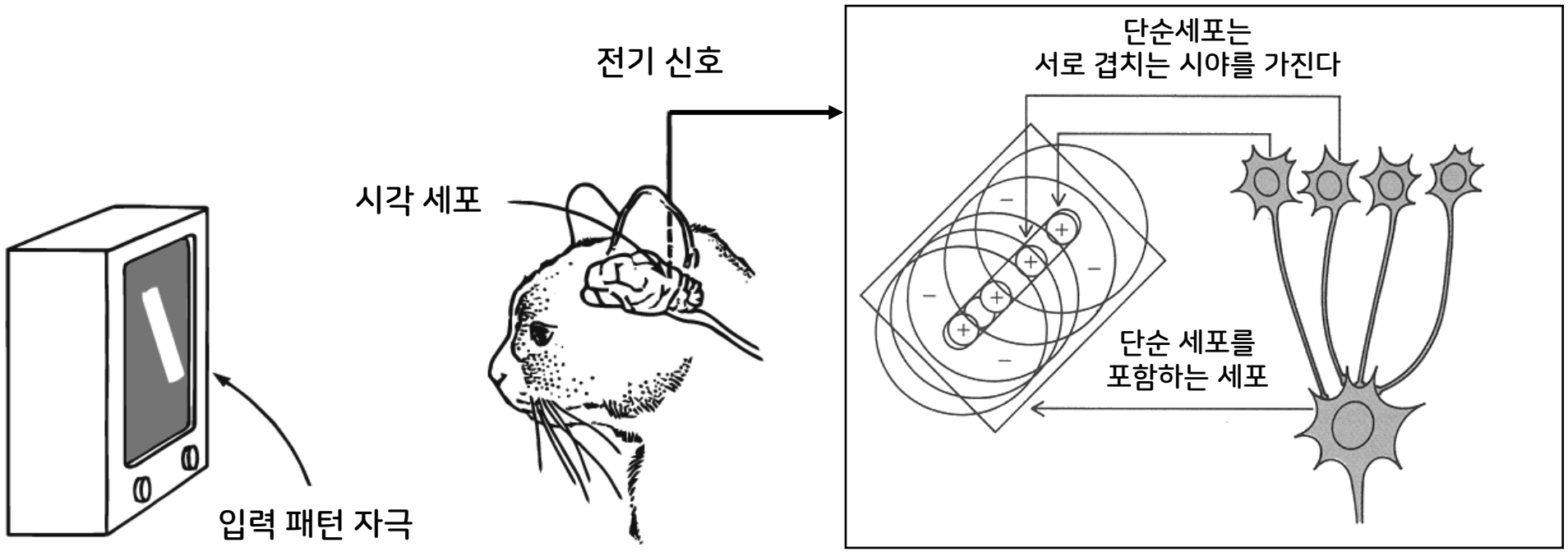
컨볼루션 신경망(CNN : Convolution Neural Network)

동물의 시각세포에 영감을 받아 개발된 인공 신경망의 한 종류

DNN과는 달리 하위 계층의 노드들과 상위 계층의 노드들이 부분적으로만 연결되어 있음

영상 및 비디오 인식, 추천 시스템 및 자연어 처리 분야 등 폭넓게 사용

컨볼루션 신경망(CNN)의 기원



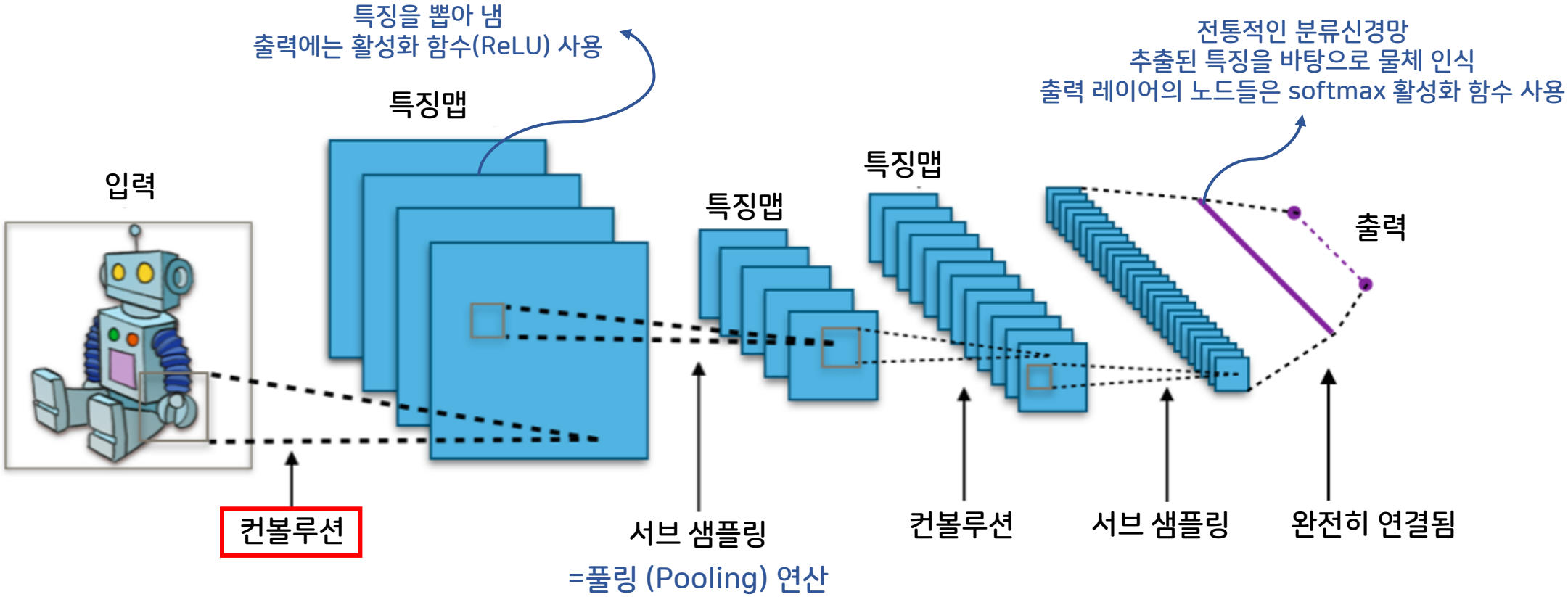
https://www.researchgate.net/figure/Fig-In-the-classic-neuroscience-experiment-Hubel-and-Wiesel-discovered-a-cats-visual_fig1_335707980

http://fourier.eng.hmc.edu/e180/lectures/v1/node7.html

시야의 특정 부위가 자극을 받으면 신경 피질에 있는 특정 뉴런이 활성화 된다는 사실을 알아냄

이 뉴런을 수용장(Receptive Fields)이라고 부르며, 수용장에 있는 뉴런들은 **모두 부분적으로 겹치는 시야** 를 갖고 있다

컨볼루션 신경망(CNN)의 구조



컨볼루션이란?

영상 처리나 컴퓨터 시각에서는 가장 기본적인 연산

주변 화소값들에 가중치를 곱해서 더한 후에 이것을 새로운 화소값으로 하는 연산

ReLU(1/9X3 + 1/9X6 + 1/9X6 +
1/9X3 + 1/9X4 + 1/9X3 +
1/9X5 + 1/9X7 + 1/9X7) = 4.88

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

가중치 마스크

3	6	6	4	7	8	2	1
3	4	3	8	8	3	3	2
5	7	7	7	7	4	3	2
8	9	9	9	9	9	3	2
8	3	3	4	3	2	1	1
8	9	9	8	8	3	3	2
6	4	3	8	8	3	3	2
7	4	3	8	8	3	3	2

입력 계층

	4.88						

출력 계층

컨볼루션이란?

ReLU(1/9X6 + 1/9X6 + 1/9X4 +
1/9X4 + 1/9X3 + 1/9X8 +
1/9X7 + 1/9X7 + 1/9 X7) = 5.77

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

가중치 마스크

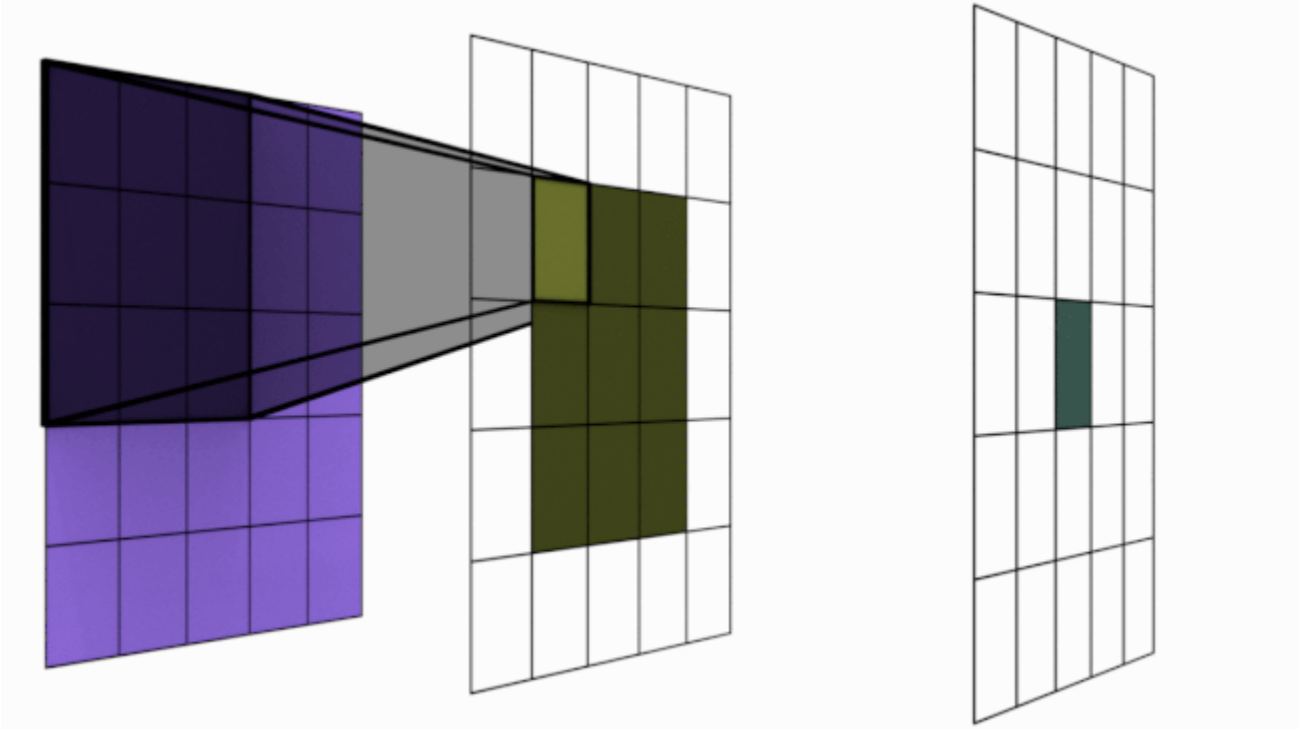
3	6	6	4	7	8	2	1
3	4	3	8	8	3	3	2
5	7	7	7	7	4	3	2
8	9	9	9	9	9	3	2
8	3	3	4	3	2	1	1
8	9	9	8	8	3	3	2
6	4	3	8	8	3	3	2
7	4	3	8	8	3	3	2

입력 계층

	4.88	5.77					

출력 계층

컨볼루션이란?



https://imgur.com/gallery/AIFHqe9

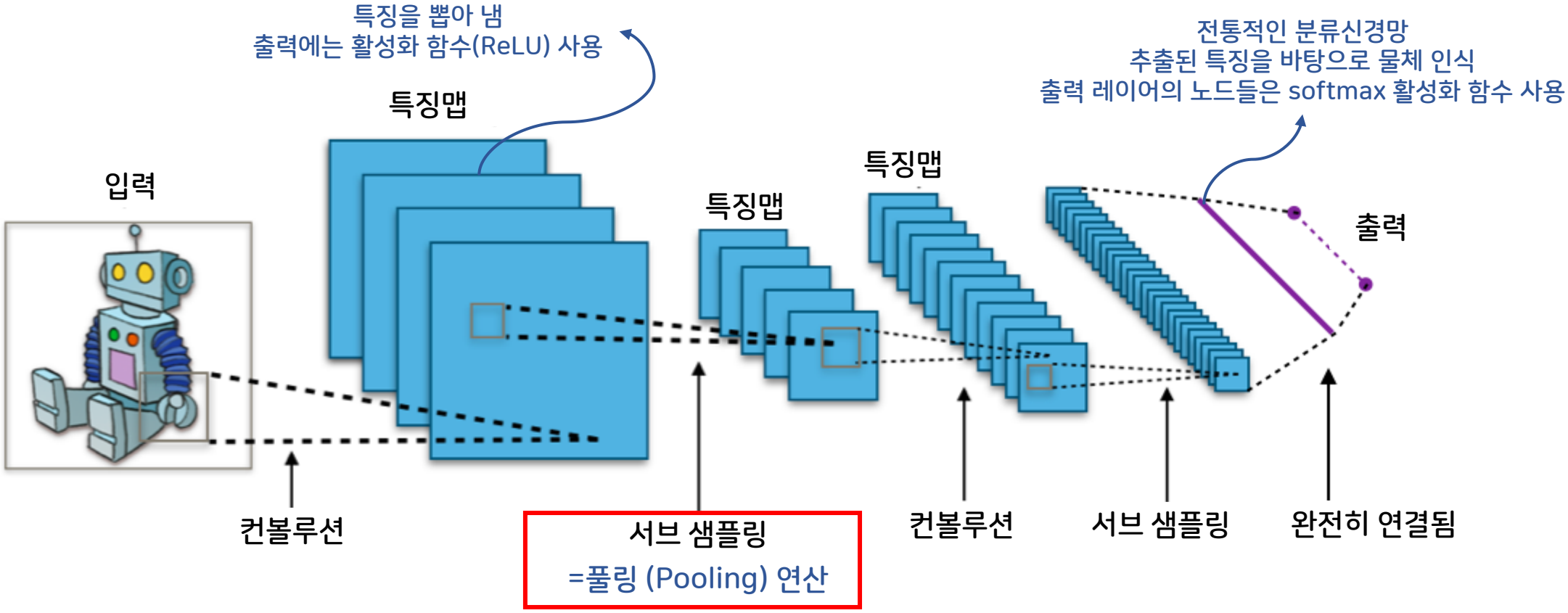
필터의 가중치 학습

필터(마스크)가 여러 개 일 경우 동시에 여러 개의 출력이 나와 동시에 여러 개 필터의 가중치 학습이 가능함

04

풀링 또는 서브 샘플링

컨볼루션 신경망(CNN)의 구조



풀링 or 서브 샘플링

입력 데이터의 크기를 줄이는 것 (입력 데이터의 깊이는 건드리지 않음)

Feature Map

6	4	8	5
5	4	5	8
3	6	7	7
7	9	7	2

Max-Pooling

https://shafeentejani.github.io/2016-12-20/convolutional-neural-nets/

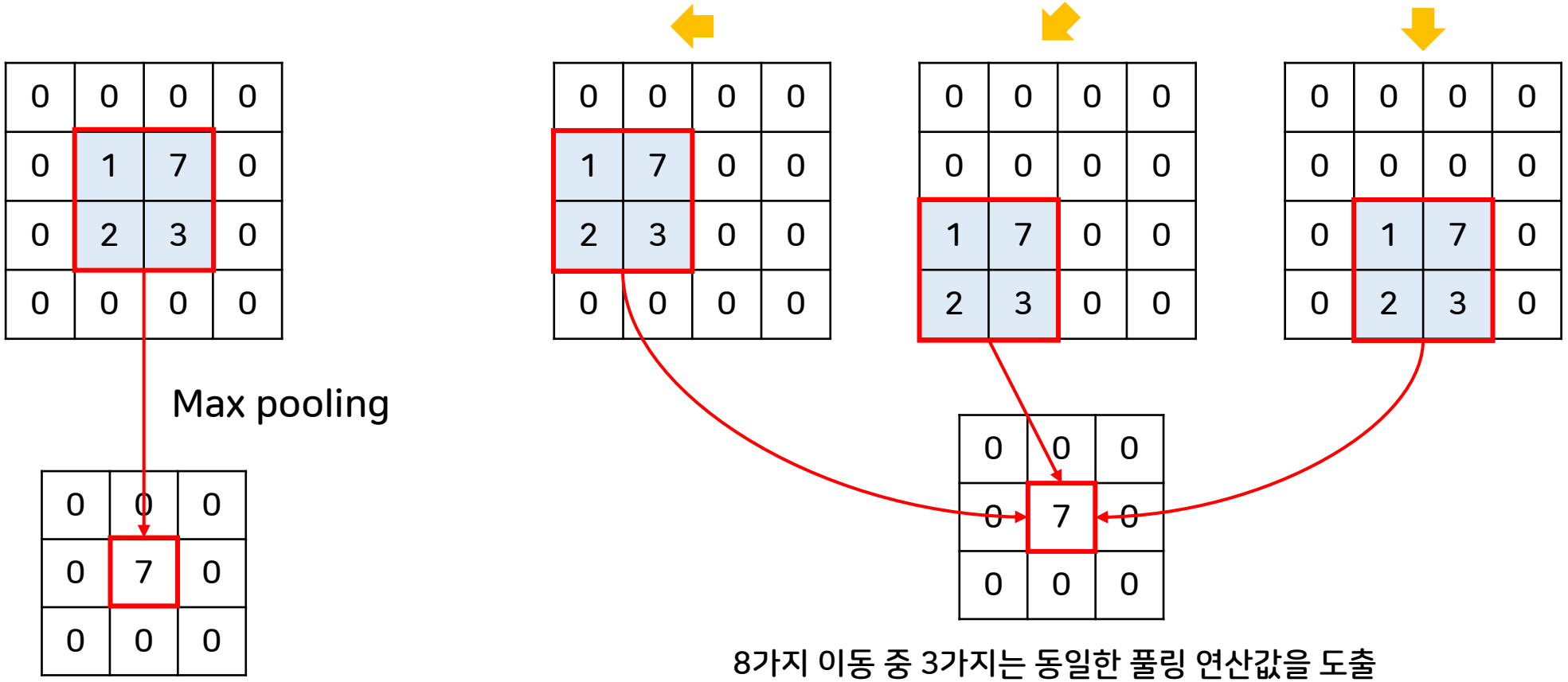
- 계층의 크기가 작아지므로 계산이 빨라진다
- 계층의 크기가 작아진다는 것은 신경망의 매개변수가 작아진다는 것을 의미한다. 따라서 과적합이 나올 가능성이 줄어든다
- 공간에서 물체의 이동이 있어도 결과는 변하지 않는다. 즉 물체의 공간이동에 대하여 둔감해지게 된다.

공간이동에 대해 둔감해지다

평행이동-불변(Translation-Invariant)

컴퓨터가 이미지 안에 있는 물체를 인식하기 위해서는 물체가 이동하더라도 동일하게 인식하여야 함

공간이동에 대해 둔감해지다



8가지 이동 중 3가지는 동일한 풀링 연산값을 도출

따라서 이미지에서 물체가 이동하여도 풀링 레이어의 값은 변하지 않을 가능성도 많음

= 이미지에서 물체가 이동하여도 동일하게 인식함

05

DNN을 이용한 영상분류

MNIST 데이터셋

Keras에서 제공하는 패션 MNIST 데이터셋을 사용

70,000개의 이미지 중에서 60,000개는 훈련하는데 이용 나머지는 성능 테스트에 사용

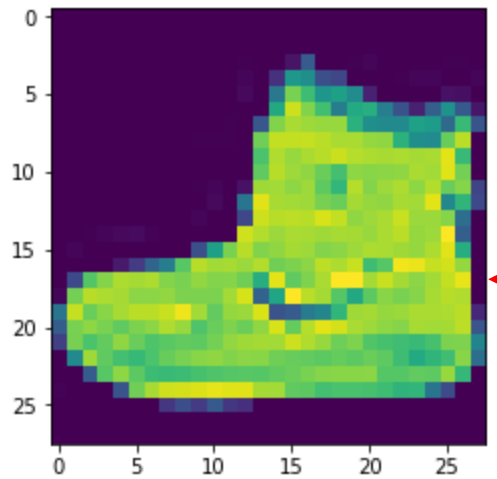
DNN : 완전연결 신경망 이용

```
import tensorflow as tf
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras import datasets, layers, models

#MNIST에 있는 fashion_image 불러오기
fashion_mnist=keras.datasets.fashion_mnist

#4개의 넘파이 배열 얻기, 앞에 두개는 훈련용 데이터셋, 뒤에 두개는 테스트용 데이터셋
(train_images, train_labels), (test_images, test_labels) =
fashion_mnist.load_data()

#입력 데이터를 화면에 이미지로 출력
plt.imshow(train_images[0])

train_images = train_images/255.0 #이미지 화소값 실수로 만들어주기 (정규화)
test_images = test_images/255.0
```

```
model = models.Sequential()

#28x28크기의 2차원 배열을 784크기의 1차원 배열로 변환
model.add(layers.Flatten(input_shape=(28,28)))

#128개의 노드로 fully-connected된 계층, 활성화 함수는 ReLU
model.add(layers.Dense(128,activation='relu'))

 #10개의 노드를 가지는 출력계층, 활성화함수는 sofrmax, 현재 이미지가 10개 범주 중 하나에 속할 확률 출력
model.add(layers.Dense(10,activation='softmax'))

#학습 방법 : 적응적 학습
#손실함수:다중분류 손실함수
#성능측정 지표 : 신경망이 올바르게 분류하는 이미지의 비율(정확도)
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

#학습 실행
model.fit(train_images, train_labels, epochs=5)

#테스트 실행
test_loss, test_acc = model.evaluate(test_images, test_labels)
print('정확도 :',test_acc)
```

DNN : 완전연결 신경망 이용 실행화면

```
Epoch 1/5
1875/1875 [==============================] - 2s 847us/step - loss: 0.5009 - accuracy: 0.8238
Epoch 2/5
1875/1875 [==============================] - 2s 858us/step - loss: 0.3754 - accuracy: 0.8633
Epoch 3/5
1875/1875 [==============================] - 1s 800us/step - loss: 0.3357 - accuracy: 0.8780
Epoch 4/5
1875/1875 [==============================] - 2s 804us/step - loss: 0.3113 - accuracy: 0.8863
Epoch 5/5
1875/1875 [==============================] - 2s 810us/step - loss: 0.2944 - accuracy: 0.8917
313/313 [==============================] - 0s 807us/step - loss: 0.3667 - accuracy: 0.8702
정확도 : 0.8701999783515930
```



입력데이터

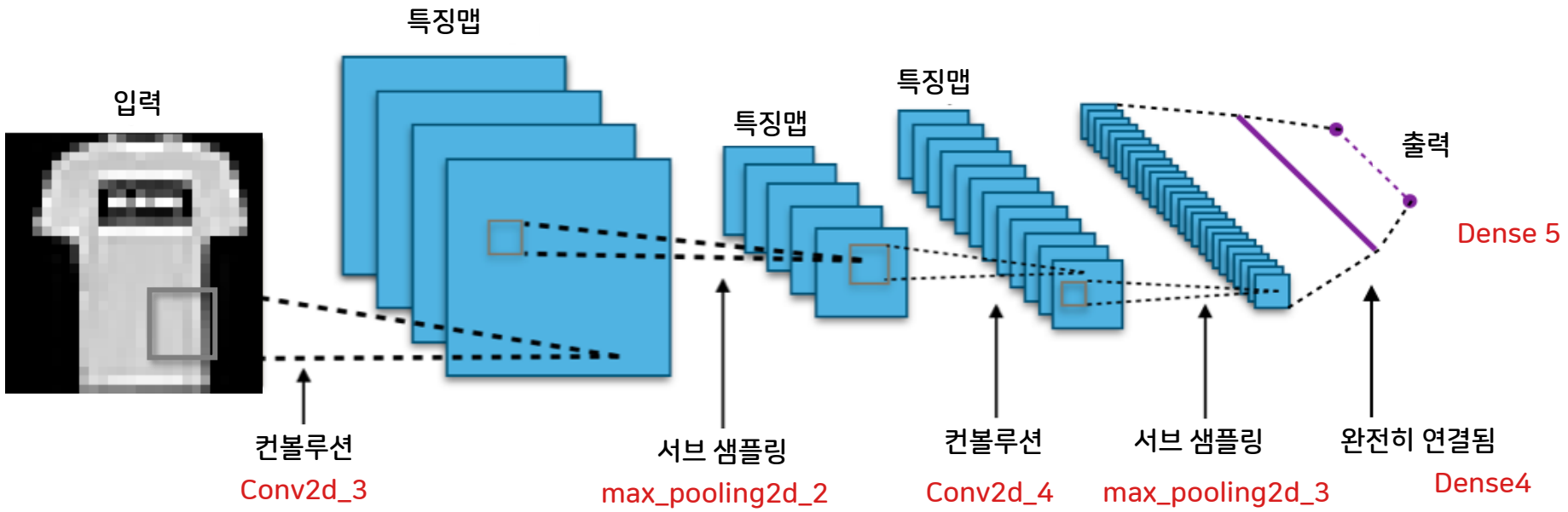
05

CNN을 이용한 영상분류

CNN : 컨볼루션 신경망 이용

```
import tensorflow as tf
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras import datasets, layers, models

fashion_mnist = keras.datasets.fashion_mnist

#4개의 넘파이 배열 얻기, 앞에 두개는 훈련용 데이터셋, 뒤에 두개는 테스트용 데이터셋
(train_images, train_labels), (test_images, test_labels) =
fashion_mnist.load_data()

#훈련 이미지로 사용할 6만장의 이미지를 28x28x1크기의 3차원 배열로 변환
train_images = train_images.reshape((60000,28,28,1))

#테스트 이미지로 사용할 1만장의 이미지를 28x28x1크기의 3차원 배열로 변환
test_images = test_images.reshape((10000,28,28,1))

train_images = train_images / 255.0 #정규화
test_images = test_images / 255.0

model=models.Sequential() #스택모델 생성

#컨볼루션층
model.add(layers.Conv2D(32,(3,3), activation='relu',input_shape=(28,28,1)))
model.add(layers.MaxPooling2D((2,2))) #서브 샘플링(풀링)

model.add(layers.Conv2D(64,(3,3), activation='relu')) #컨볼루션층
model.add(layers.MaxPooling2D((2,2)))# 서브샘플링(풀링)

model.add(layers.Conv2D(64,(3,3),activation='relu'))#컨볼루션층
model.add(layers.Flatten())#3차원 배열을 1차원으로 변환

#노드 64개의 fully-connected 신경망, 활성화 함수 : ReLU
model.add(layers.Dense(64,activation='relu'))

#노드 10개의 출력층, 활성화 함수 : softmax
model.add(layers.Dense(10,activation='softmax'))

#model.summary() #모든 층 출력

#학습 방법 : 점진적 학습
#손실함수:다중분류 손실함수
#성능측정 지표 : 신경망이 올바르게 분류하는 이미지의 비율(정확도)
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
model.fit(train_images, train_labels, epochs=5)

test_loss, test_acc = model.evaluate(test_images, test_labels)
print('정확도 :', test_acc)
```

CNN : 컨볼루션 신경망 이용




```
Epoch 1/5
1875/1875 [==============================] - 23s 12ms/step - loss: 0.4873 - accuracy: 0.8215
Epoch 2/5
1875/1875 [==============================] - 21s 11ms/step - loss: 0.3210 - accuracy: 0.8833
Epoch 3/5
1875/1875 [==============================] - 21s 11ms/step - loss: 0.2781 - accuracy: 0.8982
Epoch 4/5
1875/1875 [==============================] - 21s 11ms/step - loss: 0.2470 - accuracy: 0.9083
Epoch 5/5
1875/1875 [==============================] - 21s 11ms/step - loss: 0.2226 - accuracy: 0.9176
313/313 [==============================] - 1s 3ms/step - loss: 0.2899 - accuracy: 0.8945
정확도 : 0.8945000171661377
```

THANK YOU