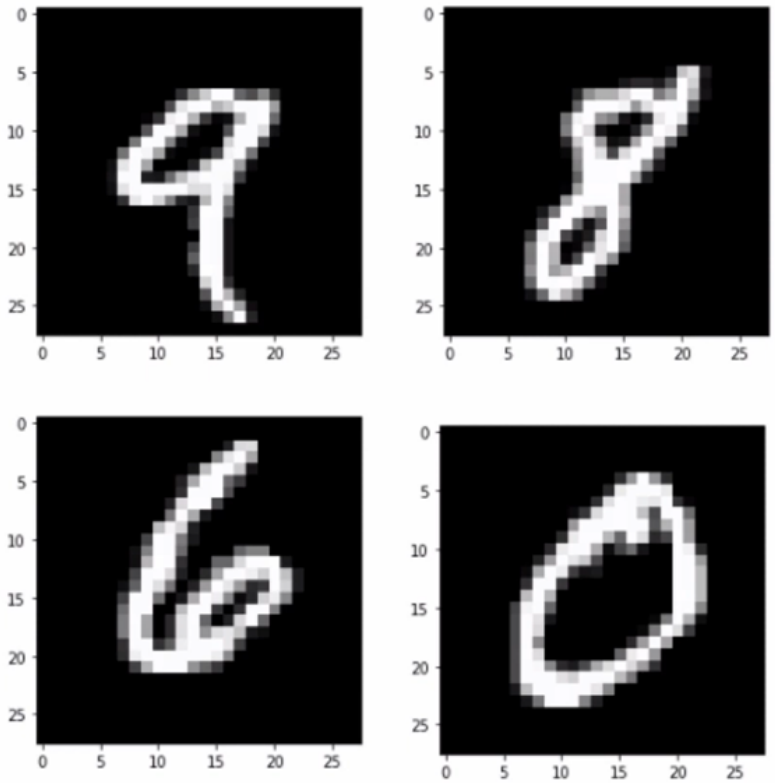
제4장: 세가지 기본 신경망II: CNN 기초

Soyoung Park

Pusan National University
Department of Statistics

soyoung@pusan.ac.kr

Convolution



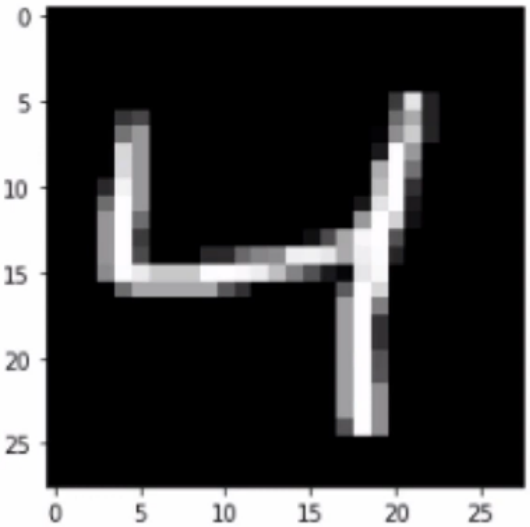
숫자	전체	위	아래
9	1	1	0
8	2	1	1
6	1	0	1
0	1	0	0

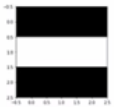
특정한 패턴의 특징이
어디서 나타나는지를 확인하는 도구

Convolution

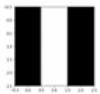
특정한 패턴의 특징이
어디서 나타나는지를 확인하는 도구

"Convolution"

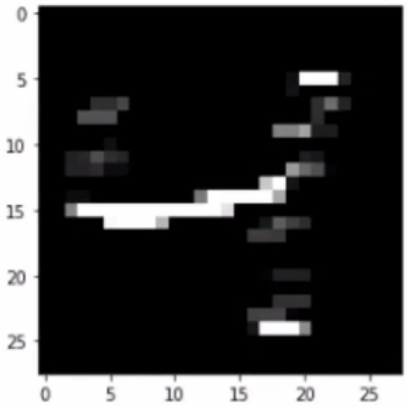


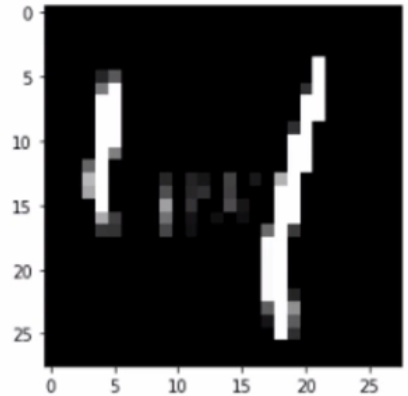


필터

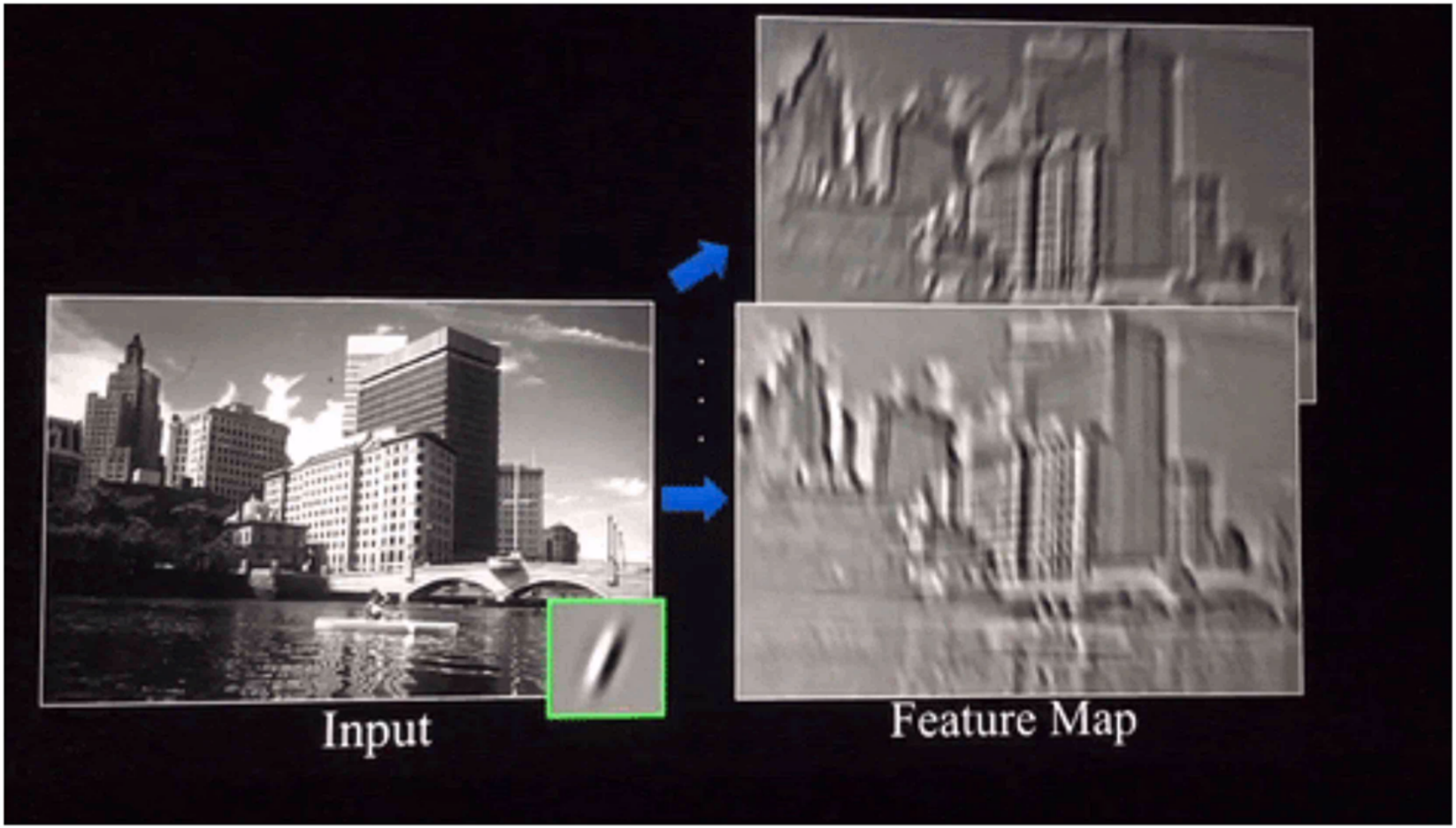


특징맵 feature map





Convolution



Convolution

Convolution Layer

```
[ ]  import tensorflow as tf
     import pandas as pd
```

```
[ ]  # 데이터를 준비하고
     (독립, 종속), _ = tf.keras.datasets.mnist.load_data()
     print(독립.shape, 종속.shape)

     독립 = 독립.reshape(60000, 28, 28, 1)
     종속 = pd.get_dummies(종속)
     print(독립.shape, 종속.shape)
```

```
[ ]  # 모델을 만들고
     X = tf.keras.layers.Input(shape=[28, 28, 1])
     H = tf.keras.layers.Conv2D(3, kernel_size=5, activation='swish')(X)
     H = tf.keras.layers.Conv2D(6, kernel_size=5, activation='swish')(H)
     H = tf.keras.layers.Flatten()(H)
     H = tf.keras.layers.Dense(84, activation='swish')(H)
     Y = tf.keras.layers.Dense(10, activation='softmax')(H)
     model = tf.keras.models.Model(X, Y)
     model.compile(loss='categorical_crossentropy', metrics='accuracy')
```

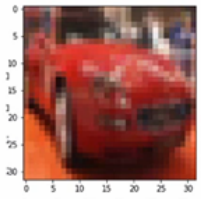
```
[ ]  # 모델을 학습하고
     model.fit(독립, 종속, epochs=10)
```

Filters

필터의 이해

1. 필터셋은 3차원 형태로 된 가중치의 모음
2. 필터셋 하나는 앞선 레이어의 결과인 "특징맵" 전체를 본다.
3. 필터셋 개수 만큼 특징맵을 만든다.

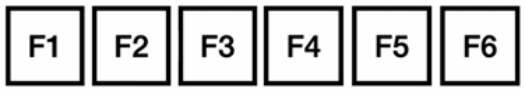
```
Conv2D(3, kernel_size=5, activation='swish')
Conv2D(6, kernel_size=5, activation='swish')
```



(32, 32, 3)

(5, 5, 3)

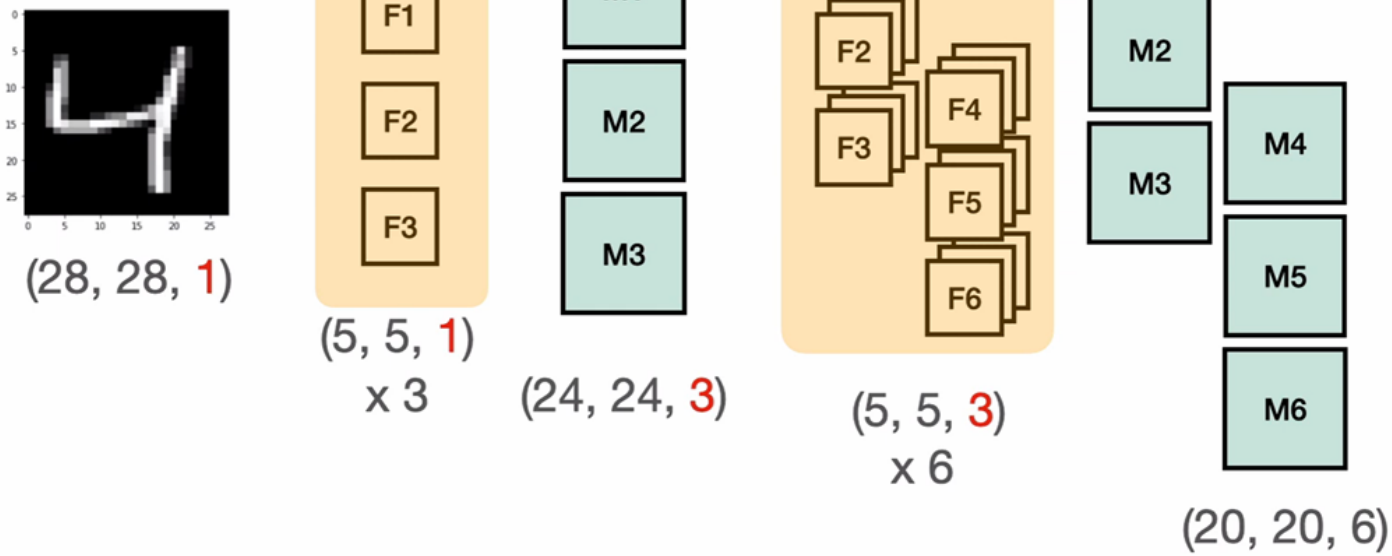
(3, 5, 5, ?)



(6, 5, 5, ?)

Filters

```
Conv2D(3, kernel_size=5, activation='swish')
Conv2D(6, kernel_size=5, activation='swish')
```



Convolution 연산

(8, 8, 1)

x1	x2	x3	x4	x5	x6	x7	x8
x9	x10	x11	x12	x13	x14	x15	x16
x17	x18	x19	...				

(3, 3, 1)

w1	w2	w3
w4	w5	w6
w7	w8	W9

필터

(6, 6)

y1	y2	y3	y4	y5	y6
y7	y8	y9	y10	...	

```
y1 = x1*w1 + x2*w2 + x3*w3 +
     x9*w4 + x10*w5 + x11*w6 +
     x17*w7 + x18*w8 + x19*w9
```

Convolution 연산

(8, 8, 1)

0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	0
0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0

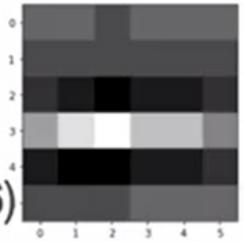
(3, 3, 1)

-1	-1	-1
2	2	2
-1	-1	-1

필터

y = * -1 + * -1 + * -1 +
* 2 + * 2 + * 2 +
* -1 + * -1 + * -1

(6, 6)



1	1	0	1	1	1
0	0	0	0	0	0
-1	-2	-3	-2	-2	-1
3	5	6	4	4	2
-2	-3	-3	-2	-2	-1
0	0	0	1	1	1

Max-pooling

```
X = tf.keras.layers.Input(shape=[28, 28])
H = tf.keras.layers.Flatten()(X)
H = tf.keras.layers.Dense(84, activation='swish')(H)
Y = tf.keras.layers.Dense(10, activation='softmax')(H)
model = tf.keras.models.Model(X, Y)
model.compile(loss='scategorical_crossentropy', metrics='accuracy')
```

$84 * (784 + 1) = 65940$

$10 * (84 + 1) = 850$

`model.summary()`

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 28, 28)]          0
_________________________________________________________________
flatten (Flatten)            (None, 784)               0
_________________________________________________________________
dense (Dense)                (None, 84)                65940
_________________________________________________________________
dense_1 (Dense)              (None, 10)                850
=================================================================
Total params: 66,790
Trainable params: 66,790
Non-trainable params: 0
_________________________________________________________________
```

Max-pooling

```
X = tf.keras.layers.Input(shape=[28, 28, 1])
H = tf.keras.layers.Conv2D(3, kernel_size=5, activation='swish')(X)
H = tf.keras.layers.Conv2D(6, kernel_size=5, activation='swish')(H)
H = tf.keras.layers.Flatten()(H)
H = tf.keras.layers.Dense(84, activation='swish')(H)
Y = tf.keras.layers.Dense(10, activation='softmax')(H)
model = tf.keras.models.Model(X, Y)
model.compile(loss='scategorical_cross
```

```
model.summary()
```

$84 * (2400 + 1) = 201{,}684$

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 28, 28, 1)]       0
_________________________________________________________________
conv2d (Conv2D)              (None, 24, 24, 3)         78
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 20, 20, 6)         456
_________________________________________________________________
flatten (Flatten)            (None, 2400)              0
_________________________________________________________________
dense (Dense)                (None, 84)                201684
_________________________________________________________________
dense_1 (Dense)              (None, 10)                850
=================================================================
Total params: 203,068
Trainable params: 203,068
Non-trainable params: 0
_________________________________________________________________
```

Pooling!!

Max-pooling

```
X = tf.keras.layers.Input(shape=[28, 28, 1])

H = tf.keras.layers.Conv2D(3, kernel_size=5, activation='swish')(X)
H = tf.keras.layers.MaxPool2D()(H)
H = tf.keras.layers.Conv2D(6, kernel_size=5, activation='swish')(H)
H = tf.keras.layers.MaxPool2D()(H)

H = tf.keras.layers.Flatten()(H)
H = tf.keras.layers.Dense(84, activation='swis
Y = tf.keras.layers.Dense(10, activation='soft
model = tf.keras.models.Model(X, Y)
model.compile(loss='scategorical_crossentropy'

model.summary()
```

$84 * (96 + 1) = 8{,}148$

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 28, 28, 1)]       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 24, 24, 3)         78
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 3)         0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 6)           456
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 4, 6)           0
_________________________________________________________________
flatten_1 (Flatten)          (None, 96)                0
_________________________________________________________________
dense_2 (Dense)              (None, 84)                8148
_________________________________________________________________
dense_3 (Dense)              (None, 10)                850
=================================================================
Total params: 9,532
Trainable params: 9,532
Non-trainable params: 0
_________________________________________________________________
```

Max-pooling

나는 특징맵

10	7	2	42	30	25
1	16	33	8	7	6
9	22	11	45	10	22
13	15	31	27	12	41
7	20	19	30	23	6
3	24	15	27	14	40

MaxPooling

16	42	30
22	45	41
24	30	40

값이 크다 = 필터로 찾으려는 특징이 많이 나타난 부분

Max-pooling

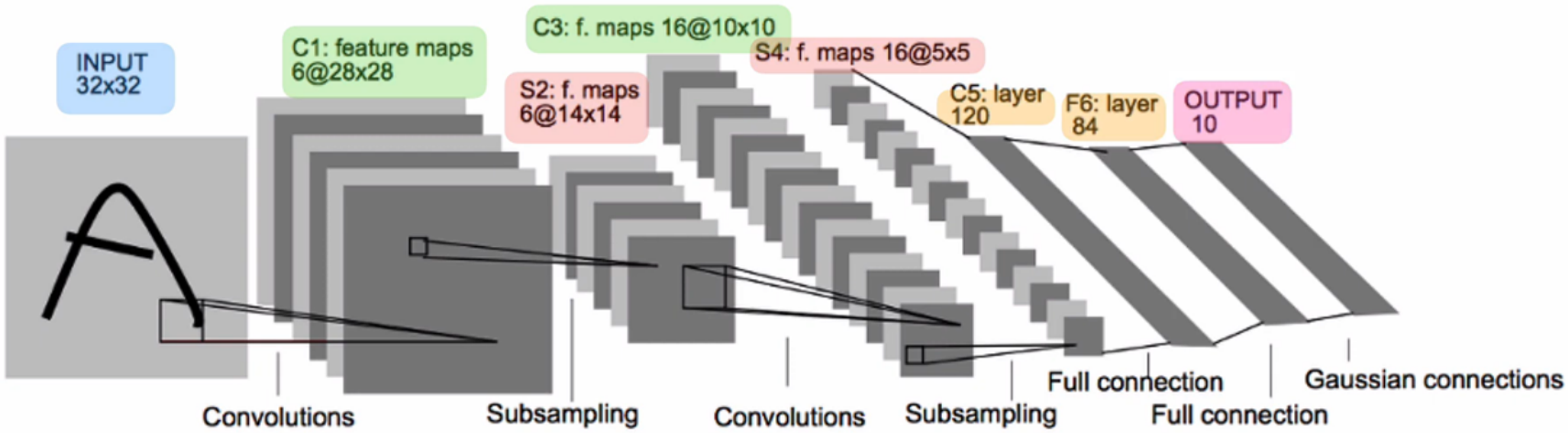
```
# 1.과거의 데이터를 준비합니다.
(독립, 종속), _ = tf.keras.datasets.mnist.load_data()
독립 = 독립.reshape(60000, 28, 28, 1)
종속 = pd.get_dummies(종속)
print(독립.shape, 종속.shape)

# 2. 모델의 구조를 만듭니다
X = tf.keras.layers.Input(shape=[28, 28, 1])

H = tf.keras.layers.Conv2D(3, kernel_size=5, activation='swish')(X)
H = tf.keras.layers.MaxPool2D()(H)
H = tf.keras.layers.Conv2D(6, kernel_size=5, activation='swish')(H)
H = tf.keras.layers.MaxPool2D()(H)

H = tf.keras.layers.Flatten()(H)
H = tf.keras.layers.Dense(84, activation='swish')(H)
Y = tf.keras.layers.Dense(10, activation='softmax')(H)
model = tf.keras.models.Model(X, Y)
model.compile(loss='scategorical_crossentropy', metrics='accuracy')
```

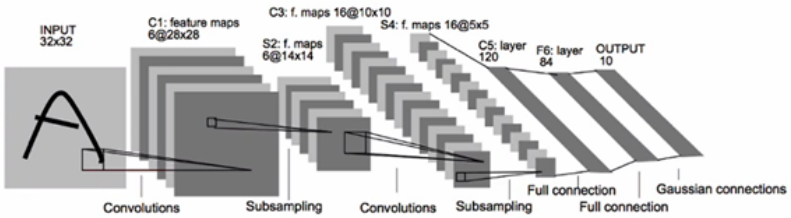
LeNet 5



CNN called LeNet by Yann LeCun (1998)

LeNet 5

```
# 1.과거의 데이터를 준비합니다.
(독립, 종속), _ = tf.keras.datasets.mnist.load_data()
독립 = 독립.reshape(60000, 28, 28, 1)
종속 = pd.get_dummies(종속)
print(독립.shape, 종속.shape)
```

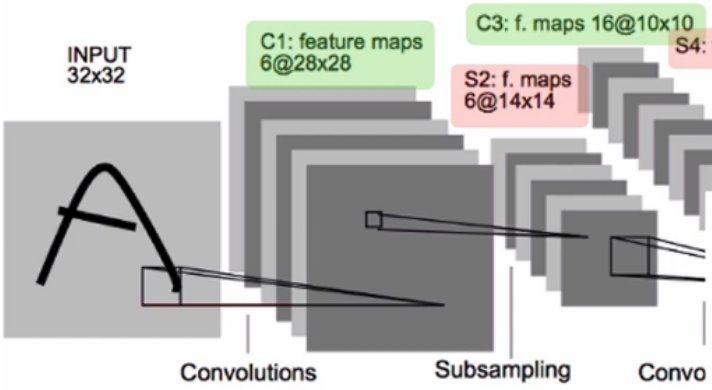


CNN called LeNet by Yann LeCun (1998)

```
# 2. 모델의 구조를 만듭니다
X = tf.keras.layers.Input(shape=[28, 28, 1])
H = tf.keras.layers.Conv2D(6, kernel_size=5, padding='same', activation='swish')(X)
H = tf.keras.layers.MaxPool2D()(H)
H = tf.keras.layers.Conv2D(16, kernel_size=5, activation='swish')(H)
H = tf.keras.layers.MaxPool2D()(H)

H = tf.keras.layers.Flatten()(H)
H = tf.keras.layers.Dense(120, activation='swish')(H)
H = tf.keras.layers.Dense(84, activation='swish')(H)
Y = tf.keras.layers.Dense(10, activation='softmax')(H)
model = tf.keras.models.Model(X, Y)
model.compile(loss='scategorical_crossentropy', metrics='accuracy')
```

LeNet 5



CNN called LeNet by

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 28, 28, 1)]	0
conv2d (Conv2D)	(None, 28, 28, 6)	156
max_pooling2d (MaxPooling2D)	(None, 14, 14, 6)	0
conv2d_1 (Conv2D)	(None, 10, 10, 16)	2416
max_pooling2d_1 (MaxPooling2	(None, 5, 5, 16)	0
flatten (Flatten)	(None, 400)	0
dense (Dense)	(None, 120)	48120
dense_1 (Dense)	(None, 84)	10164
dense_2 (Dense)	(None, 10)	850

Total params: 61,706
Trainable params: 61,706
Non-trainable params: 0

Python 실습

▾ 내 이미지 사용

```
!wget -q https://raw.githubusercontent.com/blackdew/tensorflow1/master/csv/notMNIST_small.tar.gz
!tar -xzf notMNIST_small.tar.gz
```

```
# 이미지 읽어서 데이터 준비하기
import glob
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

paths = glob.glob('./notMNIST_small/*/*.png')
paths = np.random.permutation(paths)
독립 = np.array([plt.imread(paths[i]) for i in range(len(paths))])
종속 = np.array([paths[i].split('/')[-2] for i in range(len(paths))])
print(독립.shape, 종속.shape)
```

Python 실습

```
[ ] 종속[0:10]
```

```
[ ] plt.imshow(독립[3], cmap='gray')
```

```
[ ] 독립 = 독립.reshape(18724, 28, 28, 1)
    종속 = pd.get_dummies(종속)
    print(독립.shape, 종속.shape)
```

```
import tensorflow as tf

# 모델을 완성합니다.
X = tf.keras.layers.Input(shape=[28, 28, 1])

H = tf.keras.layers.Conv2D(6, kernel_size=5, padding='same', activation='swish')(X)
H = tf.keras.layers.MaxPool2D()(H)

H = tf.keras.layers.Conv2D(16, kernel_size=5, activation='swish')(H)
H = tf.keras.layers.MaxPool2D()(H)

H = tf.keras.layers.Flatten()(H)
H = tf.keras.layers.Dense(120, activation='swish')(H)
H = tf.keras.layers.Dense(84, activation='swish')(H)
Y = tf.keras.layers.Dense(10, activation='softmax')(H)

model = tf.keras.models.Model(X, Y)
model.compile(loss='categorical_crossentropy', metrics='accuracy')
```

```
[ ] model.fit(독립, 종속, epochs=10)
```

Reference

Tensorflow 102 - 이미지 분류(CNN)

Tensorflow 102 - 이미지 분류(CNN)

공동공부 (81명)

커버 페이지

토픽 목록

- 오리엔테이션
- 데이터와 차원
- 이미지 데이터 이해
- 다섯번째 딥러닝 1 - Flatten
- 다섯번째 딥러닝 2 - Conv2D
- 다섯번째 딥러닝 3 - MaxPool2D
- 다섯번째 딥러닝 완성 - LeNet
- 내 이미지 사용하기
- 수업을 마치며

생산자

이선비

토픽 22 / 봤어요 0