zero-base /

Life Changing Education

Chapter 43. 컨볼루션 네트워크를 다시 들여다 보자

다시 MNIST using CNN

다시 MNIST 데이터를 가지고 오고~


```
import tensorflow as tf

mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
X_train, X_test = x_train/255, x_test/255

X_train = X_train.reshape((60000,28,28,1))
X_test = X_test.reshape((10000,28,28,1))
```

이번에는 구조를 간단히 가지고 오고~

```
from tensorflow.keras import layers, models

model = models.Sequential([
    layers.Conv2D(3, kernel_size=(3,3), strides=(1,1),
                  padding='same', activation='relu',
                  input_shape=(28,28,1)),
    layers.MaxPool2D(pool_size=(2,2), strides=(2,2)),
    layers.Dropout(0.25),
    layers.Flatten(),
    layers.Dense(1000, activation='relu'),
    layers.Dense(10, activation='softmax')
])
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 3)         30
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 3)         0
_________________________________________________________________
dropout (Dropout)            (None, 14, 14, 3)         0
_________________________________________________________________
flatten (Flatten)            (None, 588)               0
_________________________________________________________________
dense (Dense)                (None, 1000)              589000
_________________________________________________________________
dense_1 (Dense)              (None, 10)                10010
=================================================================
Total params: 599,040
Trainable params: 599,040
Non-trainable params: 0
_________________________________________________________________
```

내가 구성한 layers들은 이렇게 호출할 수도 있다


```
model.layers
```

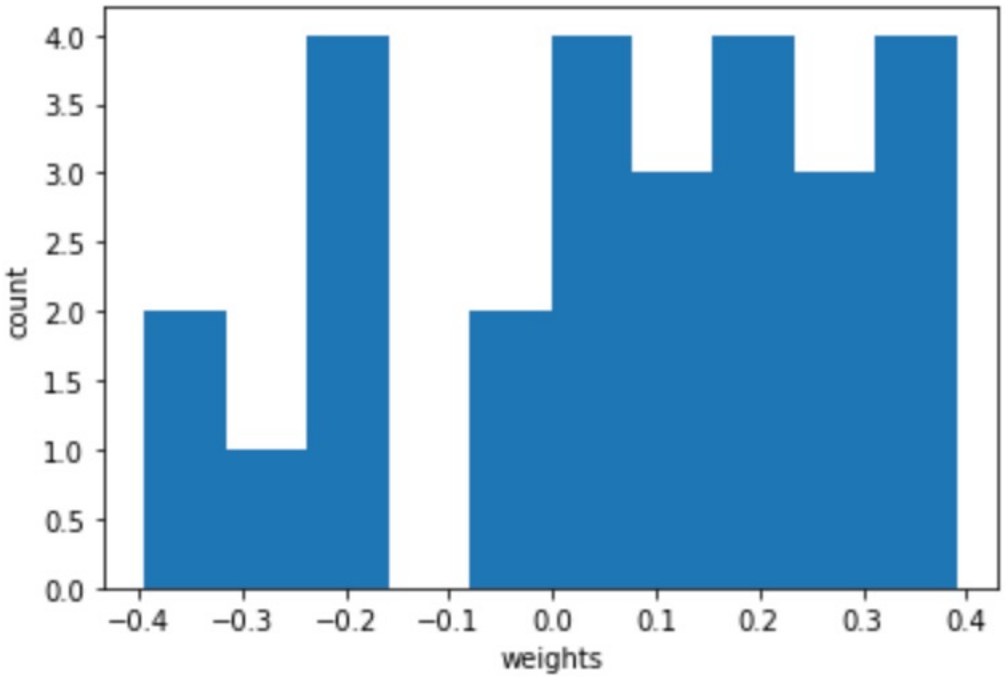
```
[<tensorflow.python.keras.layers.convolutional.Conv2D at 0x7f865077e250>,
 <tensorflow.python.keras.layers.pooling.MaxPooling2D at 0x7f865079efd0>,
 <tensorflow.python.keras.layers.core.Dropout at 0x7f868120b510>,
 <tensorflow.python.keras.layers.core.Flatten at 0x7f868120b790>,
 <tensorflow.python.keras.layers.core.Dense at 0x7f868120ba10>,
 <tensorflow.python.keras.layers.core.Dense at 0x7f868120be10>]
```

아직 학습하지 않은 conv 레이어의 웨이트의 평균


```
conv = model.layers[0]
conv_weights = conv.weights[0].numpy()
conv_weights.mean(), conv_weights.std()
```

```
(0.05563381, 0.22272052)
```

```
import matplotlib.pyplot as plt

plt.hist(conv_weights.reshape(-1,1))
plt.xlabel('weights')
plt.ylabel('count')
plt.show()
```



```
fig, ax = plt.subplots(1,3, figsize=(15,5))
for i in range(3):
    ax[i].imshow(conv_weights[:, :, 0, i], vmin=-0.5, vmax=0.5)
    ax[i].axis('off')

plt.show()
```

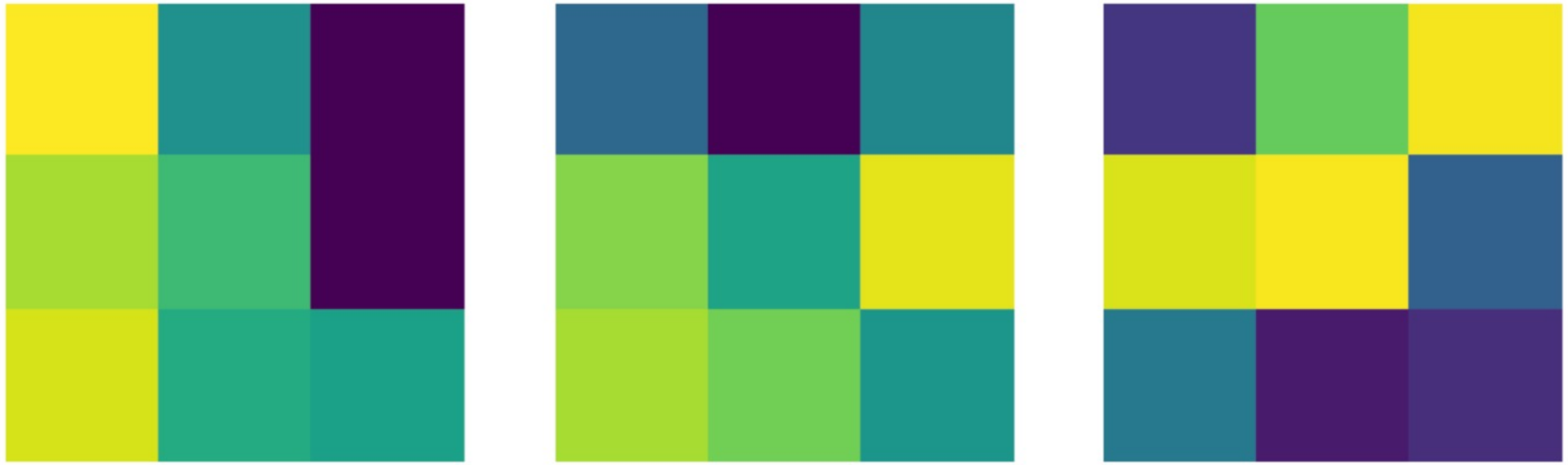
학습도 빨리 걸릴 것이고~


```
%%time
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
hist = model.fit(X_train, y_train, epochs=5, verbose=1,
                 validation_data = (X_test, y_test))
```

```
Epoch 1/5
1875/1875 [==============================] - 10s 5ms/step - loss: 0.
4349 - accuracy: 0.8703 - val_loss: 0.0828 - val_accuracy: 0.9739
```

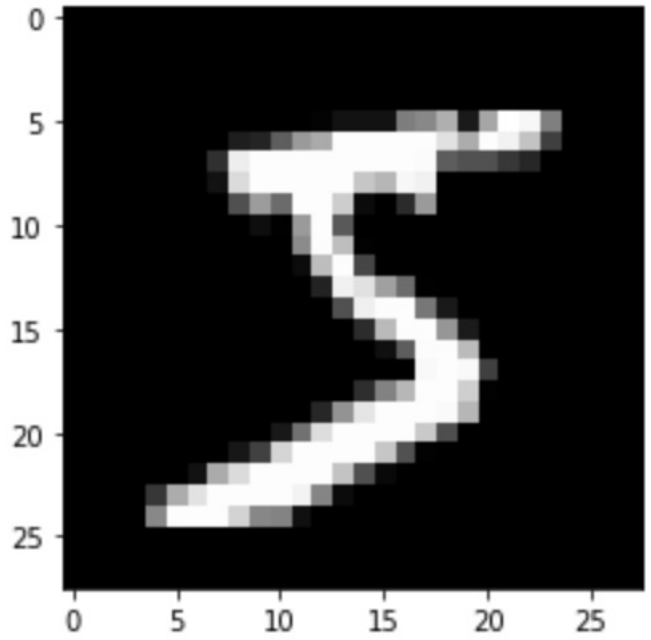
학습 후 conv filter의 변화


```
fig, ax = plt.subplots(1,3, figsize=(15,5))
for i in range(3):
    ax[i].imshow(conv_weights[:, :, 0, i], vmin=-0.5, vmax=0.5)
    ax[i].axis('off')

plt.show()
```



0번 데이터는 5

```
plt.imshow(X_train[0], cmap='gray');
```



Conv 레이어에서 출력을 뽑고~


```
inputs = X_train[0].reshape(-1, 28, 28, 1)
conv_layer_output = tf.keras.Model(model.input, model.layers[0].output)
conv_layer_output.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_input (InputLayer)    [(None, 28, 28, 1)]       0
_________________________________________________________________
conv2d (Conv2D)              (None, 28, 28, 3)         30
=================================================================
Total params: 30
Trainable params: 30
Non-trainable params: 0
_________________________________________________________________
```

입력에 대한 feature map을 뽑고~


```
feature_maps = conv_layer_output.predict(inputs)
feature_maps.shape
```

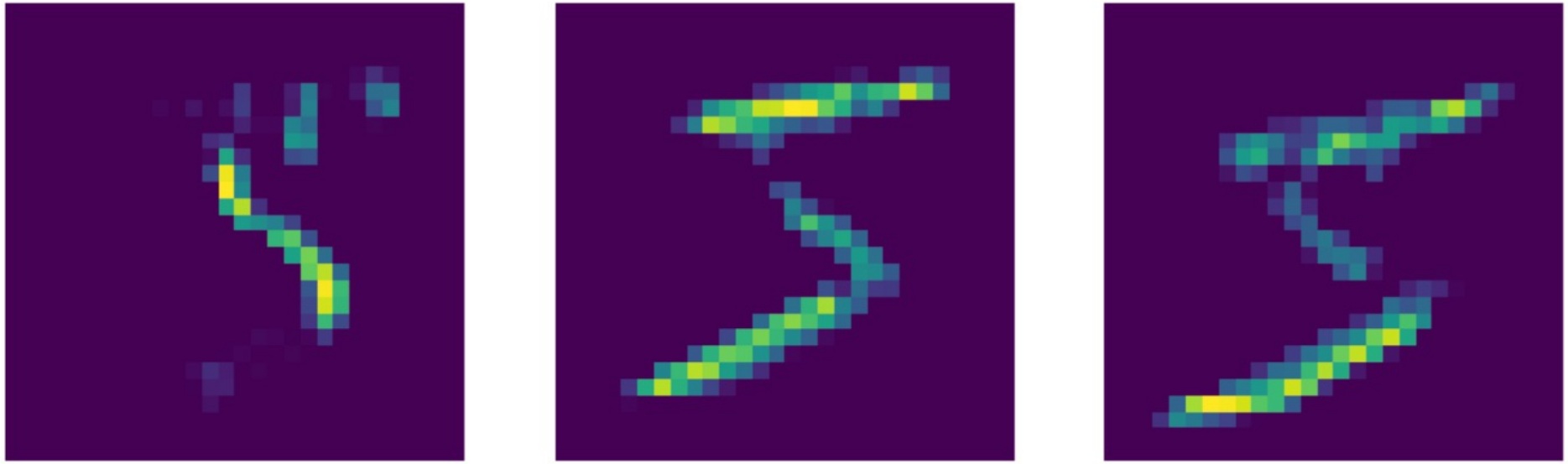
```
(1, 28, 28, 3)
```

```
feature_maps[0, :, :, 0].shape
```

```
(28, 28)
```

Feature map이 본 숫자 5


```
fig, ax = plt.subplots(1,3, figsize=(15,5))
for i in range(3):
    ax[i].imshow(feature_maps[0, :, :, i])
    ax[i].axis('off')

plt.show()
```

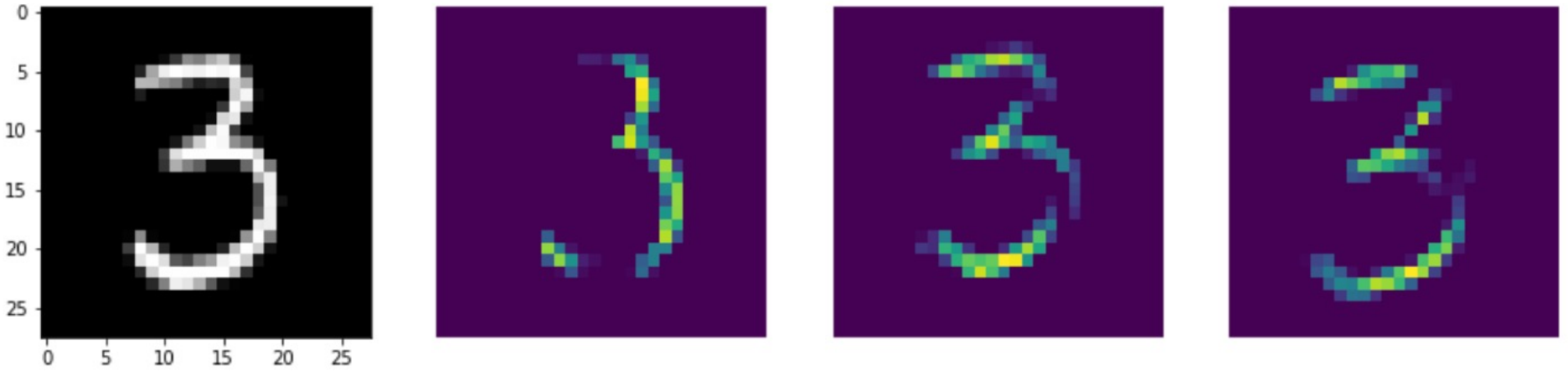


방금 전 과정을 함수로 만들고~

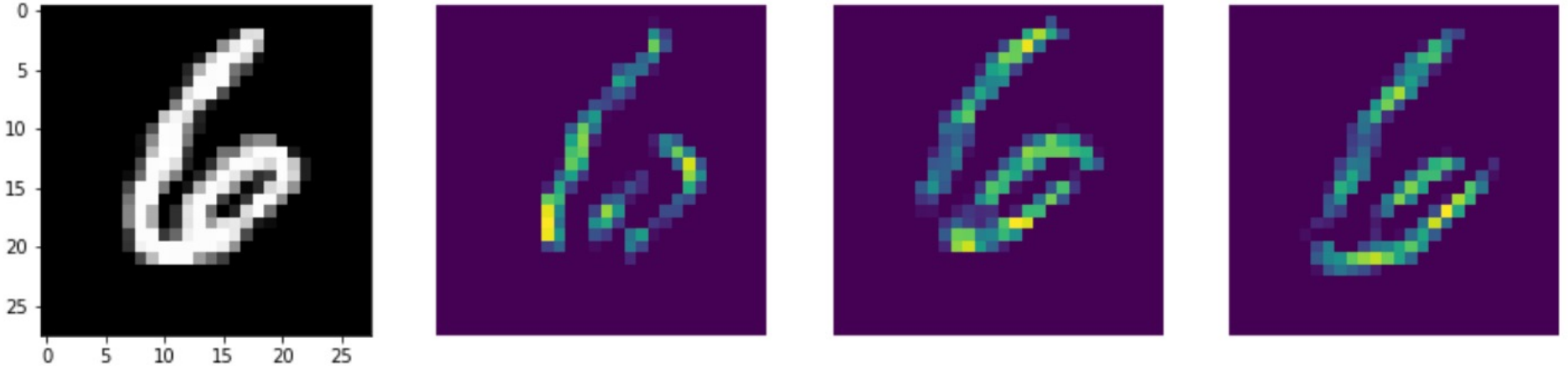

```
def draw_feature_maps(n):
    inputs = X_train[n].reshape(-1, 28, 28, 1)
    feature_maps = conv_layer_output.predict(inputs)

    fig, ax = plt.subplots(1,4, figsize=(15,5))

    ax[0].imshow(inputs[0, :, :, 0], cmap='gray');
    for i in range(1, 4):
        ax[i].imshow(feature_maps[0, :, :, i-1])
        ax[i].axis('off')

    plt.show()
```

Ch

```
draw_feature_maps(50)
```

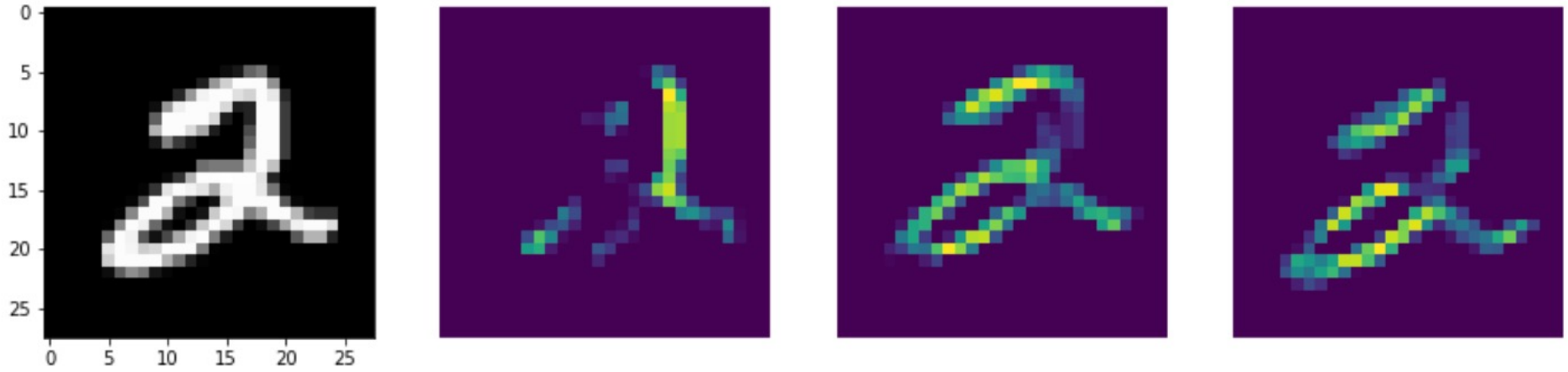


```
draw_feature_maps(13)
```

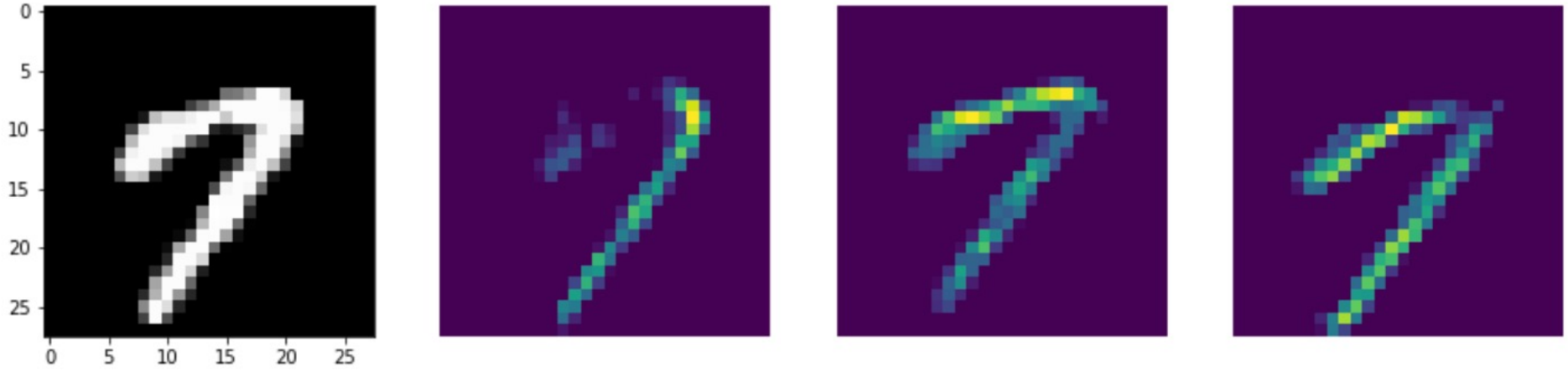


Ch

```
draw_feature_maps(5)
```




```
draw_feature_maps(15)
```



이번에는 모델 채널을 증가시키고

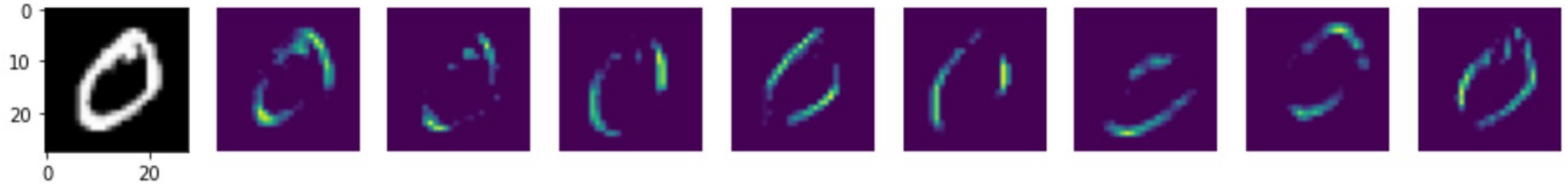

```
model1 = models.Sequential([
    layers.Conv2D(8, kernel_size=(3,3), strides=(1,1), padding='same',
                  activation='relu', input_shape=(28,28,1)),
    layers.MaxPool2D(pool_size=(2,2), strides=(2,2)),
    layers.Dropout(0.25),
    layers.Flatten(),
    layers.Dense(1000, activation='relu'),
    layers.Dense(10, activation='softmax')
])
```

```
%%time
model1.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
               metrics=['accuracy'])
hist = model1.fit(X_train, y_train, epochs=5, verbose=1,
                  validation_data = (X_test, y_test))
```

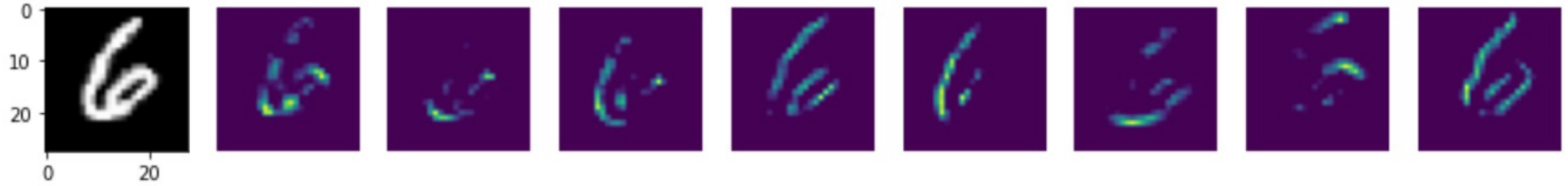
```
Epoch 1/5
1875/1875 [==============================] - 28s 15ms/step - loss: 0.3007
- accuracy: 0.9083 - val_loss: 0.0661 - val_accuracy: 0.9789
Epoch 2/5
1875/1875 [==============================] - 28s 15ms/step - loss: 0.0697
```

```
conv_layer_output = tf.keras.Model(model1.input, model1.layers[0].output)

def draw_feature_maps(n):
    inputs = X_train[n].reshape(-1, 28, 28, 1)
    feature_maps = conv_layer_output.predict(inputs)

    fig, ax = plt.subplots(1,9, figsize=(15,5))

    ax[0].imshow(inputs[0, :, :, 0], cmap='gray');
    for i in range(1, 9):
        ax[i].imshow(feature_maps[0, :, :, i-1])
        ax[i].axis('off')

    plt.show()
```

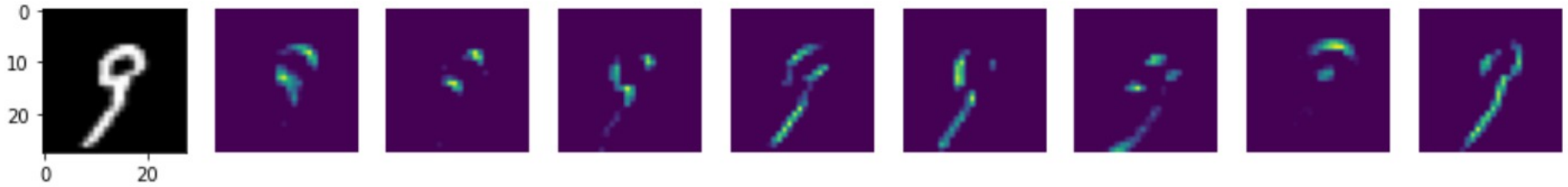
```
draw_feature_maps(1)
```




```
draw_feature_maps(13)
```



```
draw_feature_maps(19)
```




```
draw_feature_maps(25)
```