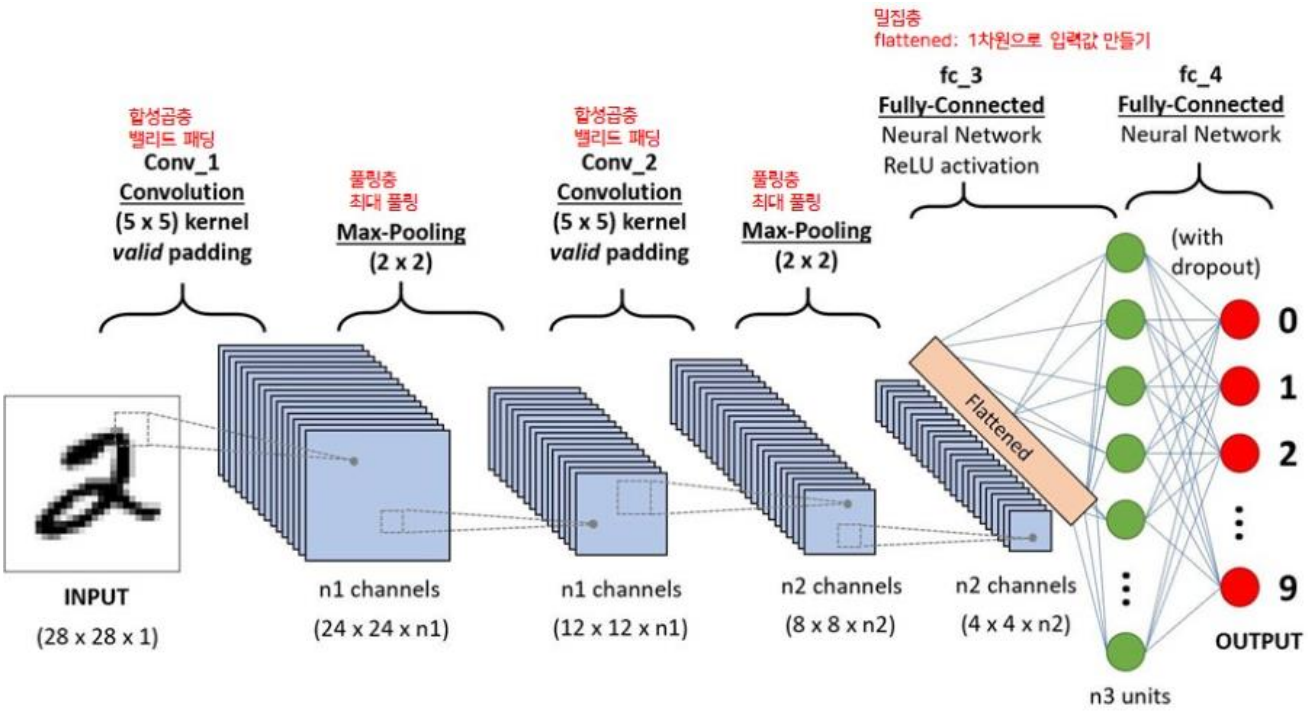
디지털 영상처리 연구실 연구보고서

김우헌

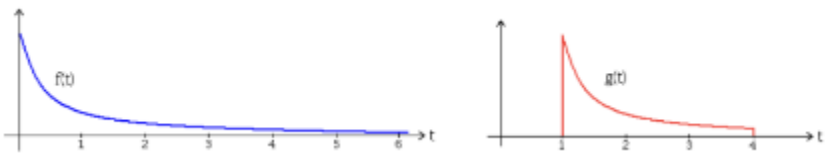
#CNN(convolutional neural network)

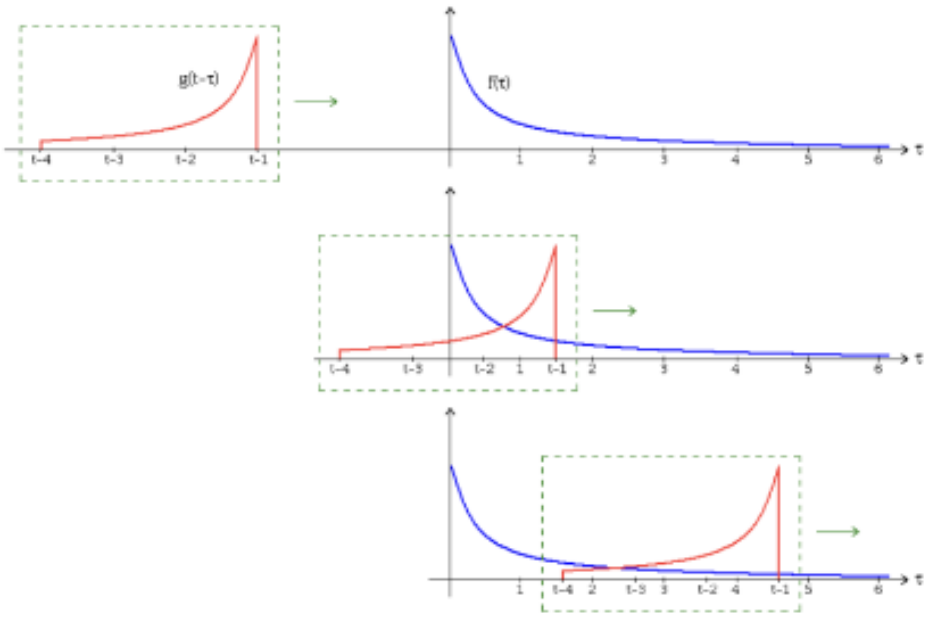


1. 입력층
2. 합성곱층
3. 풀링층
4. 완전연결층
5. 출력층

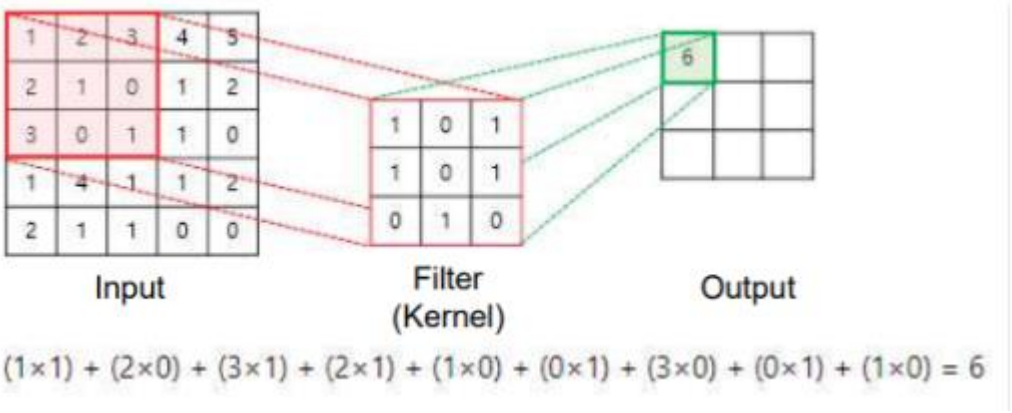
#합성곱연산

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$



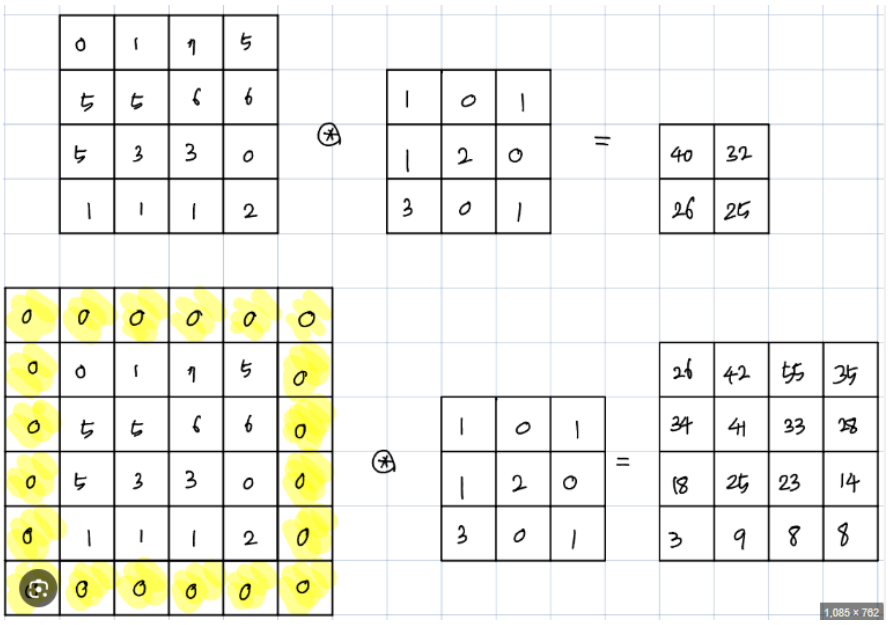


#2차원 배열의 합성곱

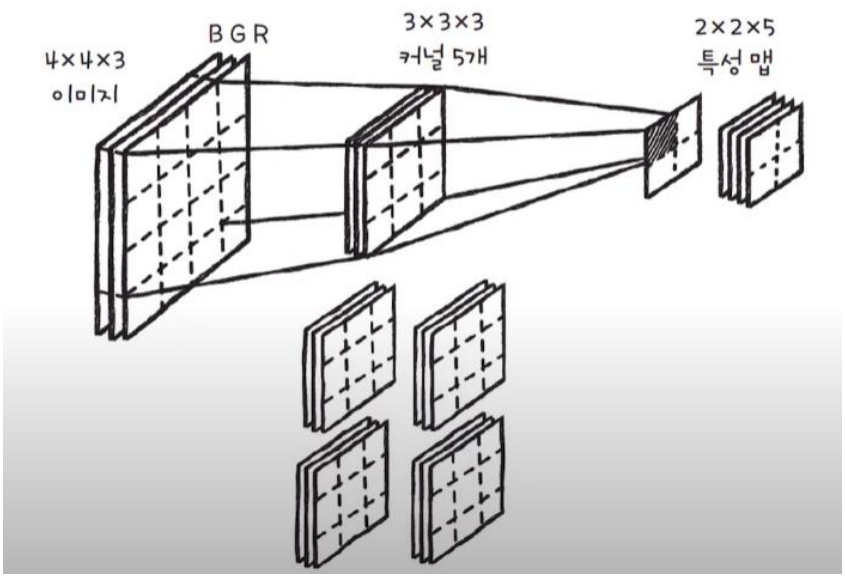


-> 스트라이드?

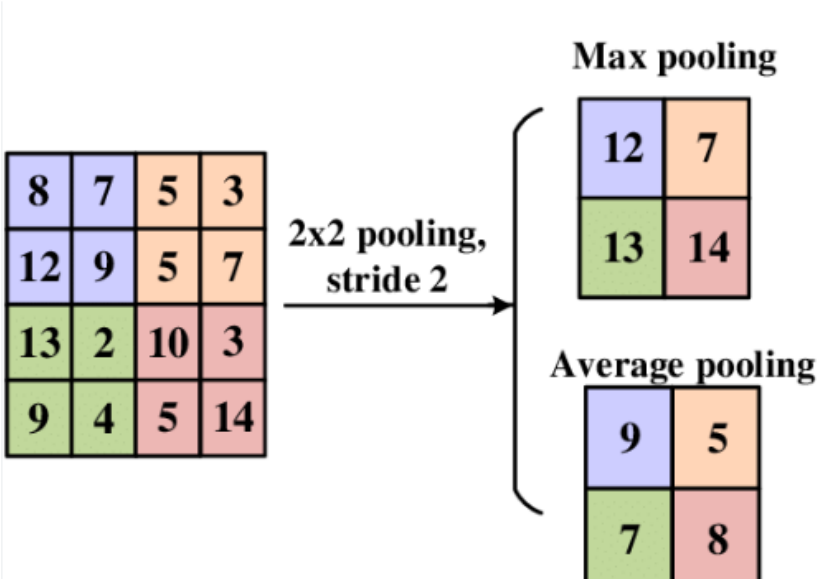
-> 패딩?



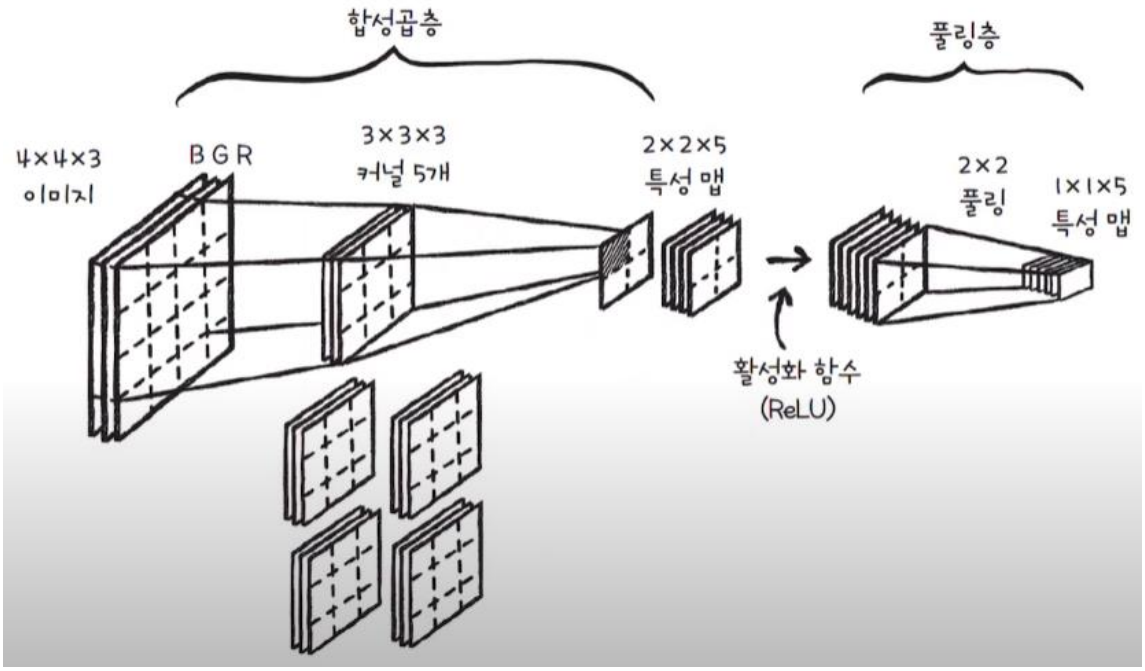
#합성곱층


4×4×3
이미지
BGR
3×3×3
커널 5개
2×2×5
특성 맵

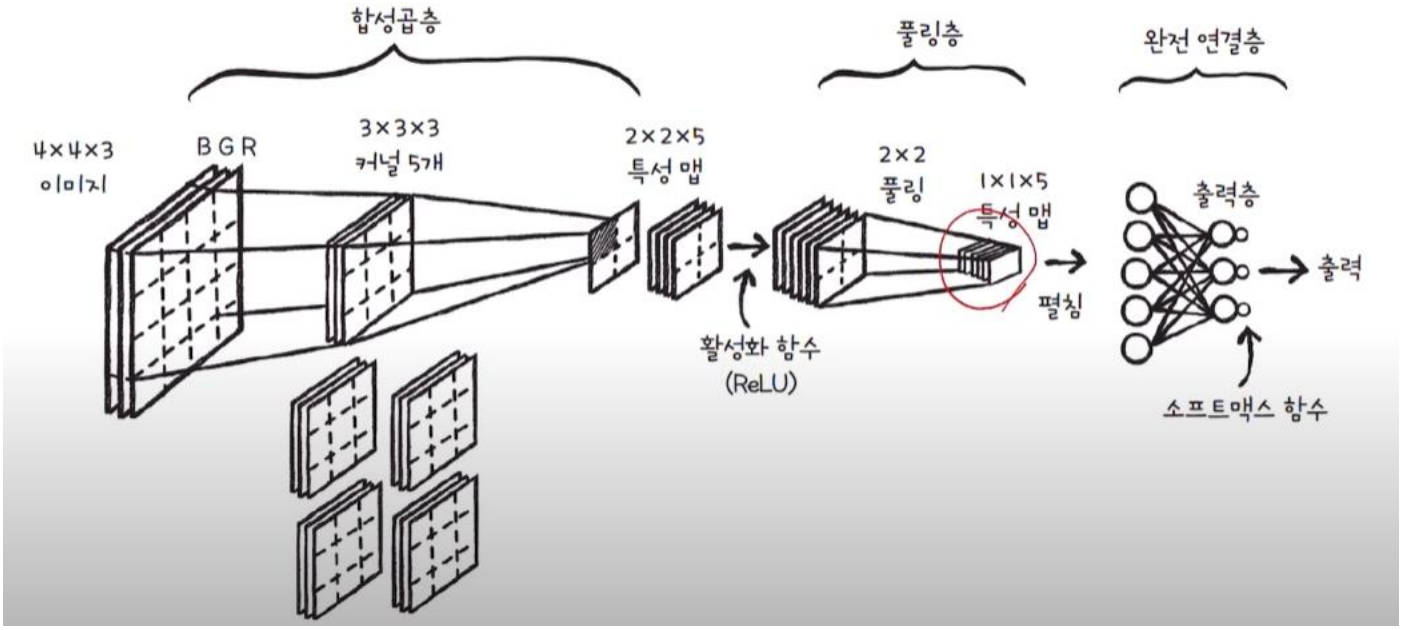
#풀링층


8 7 5 3
12 9 5 7
13 2 10 3
9 4 5 14
2x2 pooling,
stride 2
Max pooling
12 7
13 14
Average pooling
9 5
7 8

#합성곱층+풀링층



합성곱층+풀링층+완전연결층+출력층

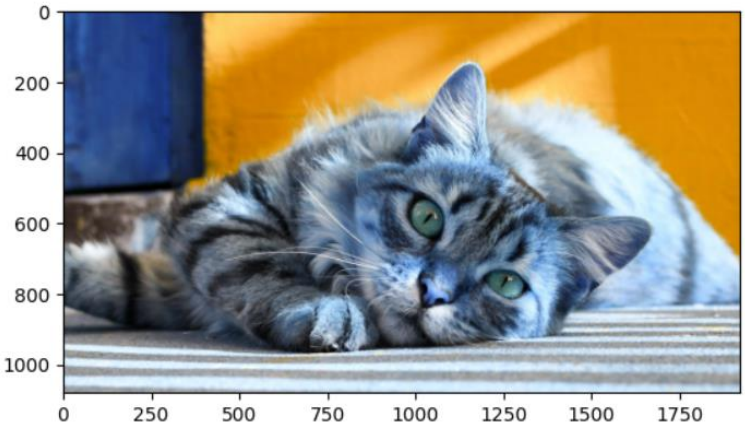


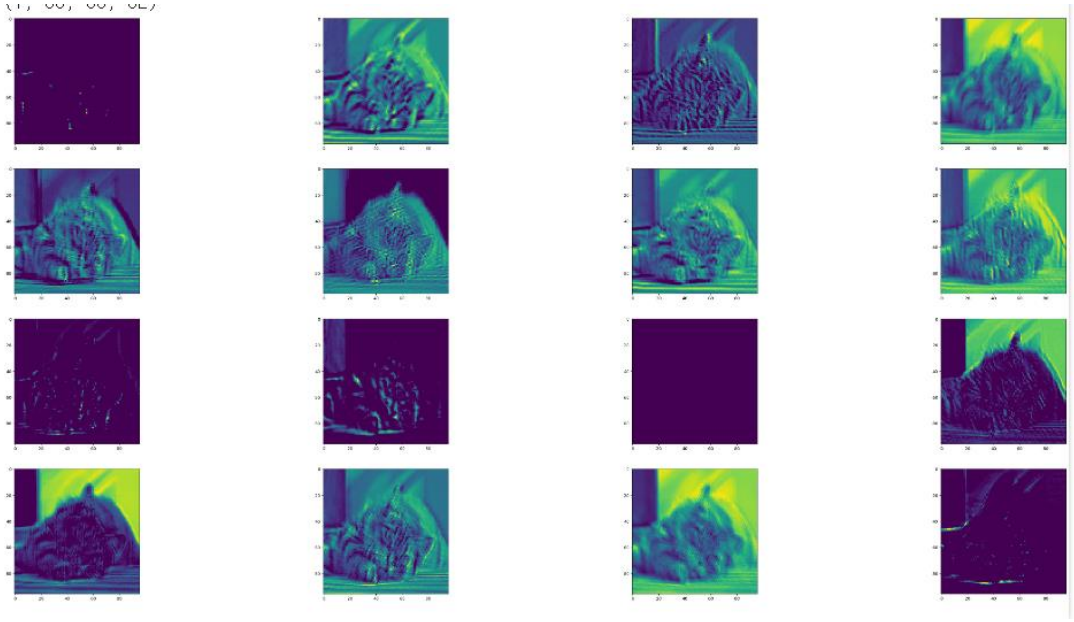
#텐서플로를 통한 cnn모델링

```
model= tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(input_shape=(100,100,3), activation='relu', kernel_size=(5,5), filters=32),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(activation='relu', kernel_size=(5,5), filters=64),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(activation='relu', kernel_size=(5,5), filters=64),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(activation='relu', kernel_size=(5,5), filters=64),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(2, activation='softmax')
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 96, 96, 32)	2432
max_pooling2d (MaxPooling2D)	(None, 48, 48, 32)	0
conv2d_1 (Conv2D)	(None, 44, 44, 64)	51264
max_pooling2d_1 (MaxPooling2D)	(None, 22, 22, 64)	0
conv2d_2 (Conv2D)	(None, 18, 18, 64)	102464
max_pooling2d_2 (MaxPooling2D)	(None, 9, 9, 64)	0
conv2d_3 (Conv2D)	(None, 5, 5, 64)	102464
max_pooling2d_3 (MaxPooling2D)	(None, 2, 2, 64)	0
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 128)	32896
dense_1 (Dense)	(None, 64)	8256
dense_2 (Dense)	(None, 32)	2080
dense_3 (Dense)	(None, 2)	66

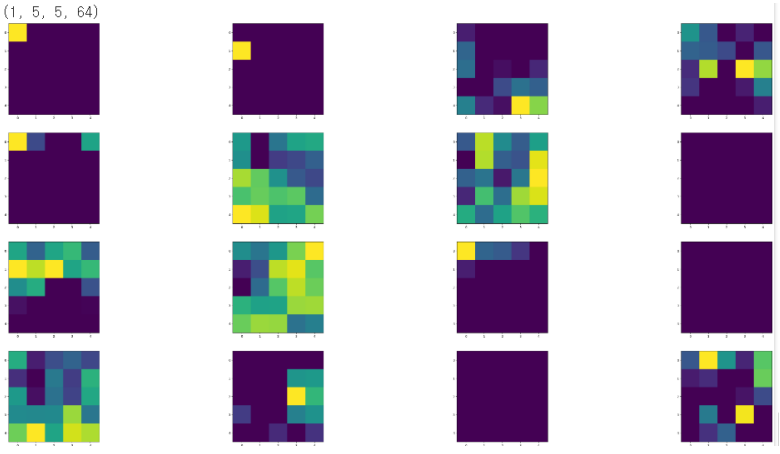
#Feature map





conv2d_input (InputLayer)	[(None, 100, 100, 3)]	0
conv2d (Conv2D)	(None, 96, 96, 32)	2432

->96*96의 feature map 32개



->5*5의 feature map 64개 (6번째 계층)