CNN(Convolution Neural Network) - 합성곱 신경망

학습 내용

- CNN의 기본 이해
- CNN 실습해보기

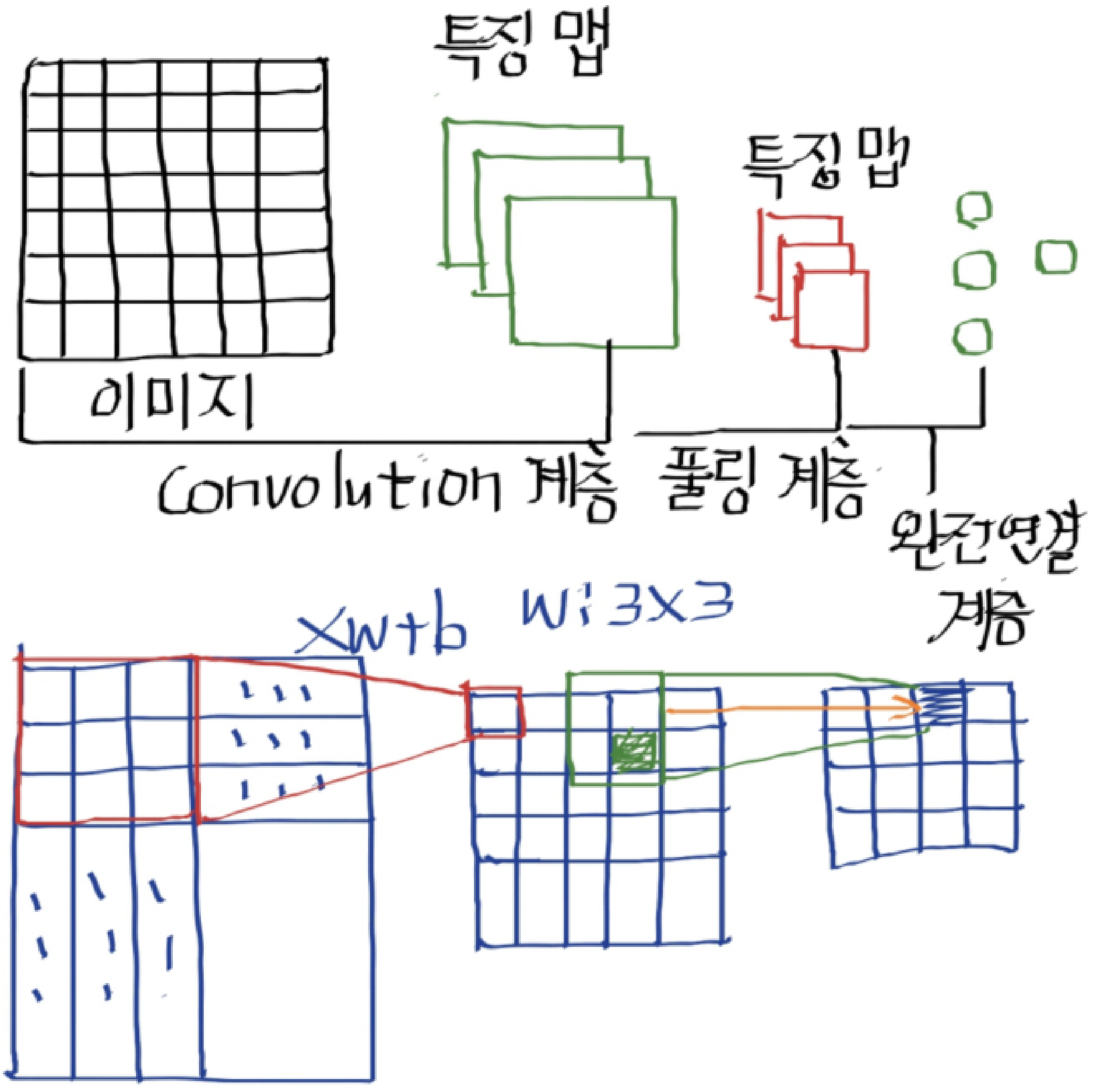
In [12]:

```
from IPython.display import display, Image
import os, warnings

warnings.filterwarnings(action='ignore')
```

In [2]:

```
display(Image(filename="img/cnn.png"))
```



01 합성망 신경망 알아보기

In [13]:

```
from keras import layers
from keras import models
```

- Conv :3x3 필터, 32개의 필터개수, 입력 이미지 (28, 28, 1)
- Maxpooling (2,2)
- 3x3 필터, 64개의 필터개수
- Maxpooling (2,2)
- 3x3 필터, 64개의 필터개수

In [14]:

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu',
                        input_shape=(28, 28, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```

컨브넷 구조 알아보기

In [15]:

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_6 (Conv2D)            (None, 26, 26, 32)        320       
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 13, 13, 32)        0         
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 11, 11, 64)        18496     
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 5, 5, 64)          0         
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 3, 3, 64)          36928     
=================================================================
Total params: 55,744
Trainable params: 55,744
Non-trainable params: 0
_________________________________________________________________
```

- (height, width, channels)크기의 3D텐서
- 높이와 넓이 차원은 네트워크가 깊어질수록 작아지는 경향이 있다.
- 채널의 수는 Conv2D층에 전달된 첫번째 매개변수에 의해 조절된다.
- (3,3,64)를 완전 연결 네트워크에 펼쳐 연결한다.

분류기 추가

In [16]:

```
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))
```

In [17]:

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_6 (Conv2D)            (None, 26, 26, 32)        320       
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 13, 13, 32)        0         
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 11, 11, 64)        18496     
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 5, 5, 64)          0         
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 3, 3, 64)          36928     
_________________________________________________________________
flatten_1 (Flatten)          (None, 576)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 64)                36928     
_________________________________________________________________
dense_3 (Dense)              (None, 10)                650       
=================================================================
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0
_________________________________________________________________
```

02 데이터 준비하기

- MNIST 데이터 셋 준비

In [18]:

```
from keras.datasets import mnist
from keras.utils import to_categorical

(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
```

In [19]:

```
train_images = train_images.reshape((60000, 28, 28, 1))
train_images = train_images.astype('float32') / 255

test_images = test_images.reshape((10000, 28, 28, 1))
test_images = test_images.astype('float32') / 255

train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)
```

02 CNN 계층 구성

비용함수, 최적화 함수 구성

- 비용함수와 최적화 함수 지정
- 비용함수 : categorical_crossentropy
- 최적화 함수 : rmsprop

In [20]:

```
%%time

model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
model.fit(train_images, train_labels, epochs=5, batch_size=64)
```

```
Epoch 1/5
938/938 [==============================] - 28s 29ms/step - loss: 0.3926 - accuracy:
0.8756
Epoch 2/5
938/938 [==============================] - 29s 31ms/step - loss: 0.0510 - accuracy:
0.9836
Epoch 3/5
938/938 [==============================] - 26s 28ms/step - loss: 0.0331 - accuracy:
0.9895
Epoch 4/5
938/938 [==============================] - 26s 28ms/step - loss: 0.0252 - accuracy:
0.9918
Epoch 5/5
938/938 [==============================] - 27s 28ms/step - loss: 0.0189 - accuracy:
0.9943
Wall time: 2min 15s
```

Out[20]:

```
<tensorflow.python.keras.callbacks.History at 0x18fd9943970>
```

In [21]:

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

```
313/313 [==============================] - 2s 5ms/step - loss: 0.0281 - accuracy: 0.
9920
0.9919999837875366
```

실습 01

- 10epochs를 돌려보기
- conv를 하나 삭제 해보기
- conv를 하나 추가 해보기