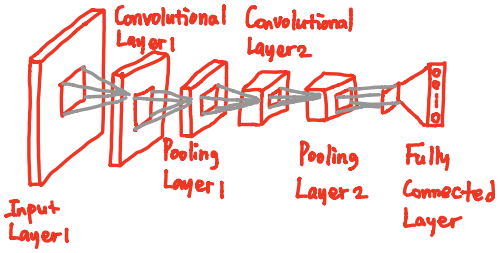
```
# Lab 11 MNIST and Convolutional Neural Network
import numpy as np
import tensorflow as tf
import random

mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_test = x_test / 255
x_train = x_train / 255
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)

# one hot encode y data
y_train = tf.keras.utils.to_categorical(y_train, 10)
y_test = tf.keras.utils.to_categorical(y_test, 10)

# hyper parameters
learning_rate = 0.001
training_epochs = 12
batch_size = 128

tf.model = tf.keras.Sequential()
# L1
tf.model.add(tf.keras.layers.Conv2D(filters=16, kernel_size=(3, 3), input_shape=(28, 28, 1),
activation='relu'))
tf.model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))

# L2
tf.model.add(tf.keras.layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu'))
tf.model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))

# L3 fully connected
tf.model.add(tf.keras.layers.Flatten())
tf.model.add(tf.keras.layers.Dense(units=10, kernel_initializer='glorot_normal', activation='softmax'))

tf.model.compile(loss='categorical_crossentropy', optimizer=tf.keras.optimizers.Adam(lr=learning_rate),
metrics=['accuracy'])
tf.model.summary()

tf.model.fit(x_train, y_train, batch_size=batch_size, epochs=training_epochs)

# predict 10 random hand-writing data
y_predicted = tf.model.predict(x_test)
for x in range(0, 10):
    random_index = random.randint(0, x_test.shape[0]-1)
    print("index: ", random_index,
          "actual y: ", np.argmax(y_test[random_index]),
          "predicted y: ", np.argmax(y_predicted[random_index]))

evaluation = tf.model.evaluate(x_test, y_test)
print('loss: ', evaluation[0])
print('accuracy', evaluation[1])
```

* L1 → Conv : (28,28,16)
  Pool : (14,14,16)
  L2 → Conv : (14,14,32)
  Pool : (7,7,32)



필터의 개수: 16 / 필터의 사이즈: 3x3 / 28x28 사이즈의 입력 데이터 / 색의 색깔

28x28 사이즈를 stride 2로 움직이면 → output : 14x14

stride : 2x2 → 2칸씩 움직이겠다.

L1의 output이 L2의 input이 되므로
input : 14x14 이고, 14x14 사이즈를 stride 2로 움직이면 → output : 7x7

↳ 값을 주 펼쳐줌 (7x7x32 = 1568)

* Layer의 개수를 늘리면, 더 높은 정확도를 얻을 수 있습니다.
  ex) L1~L4 : accuracy = 0.9938 ...