딥러닝 6

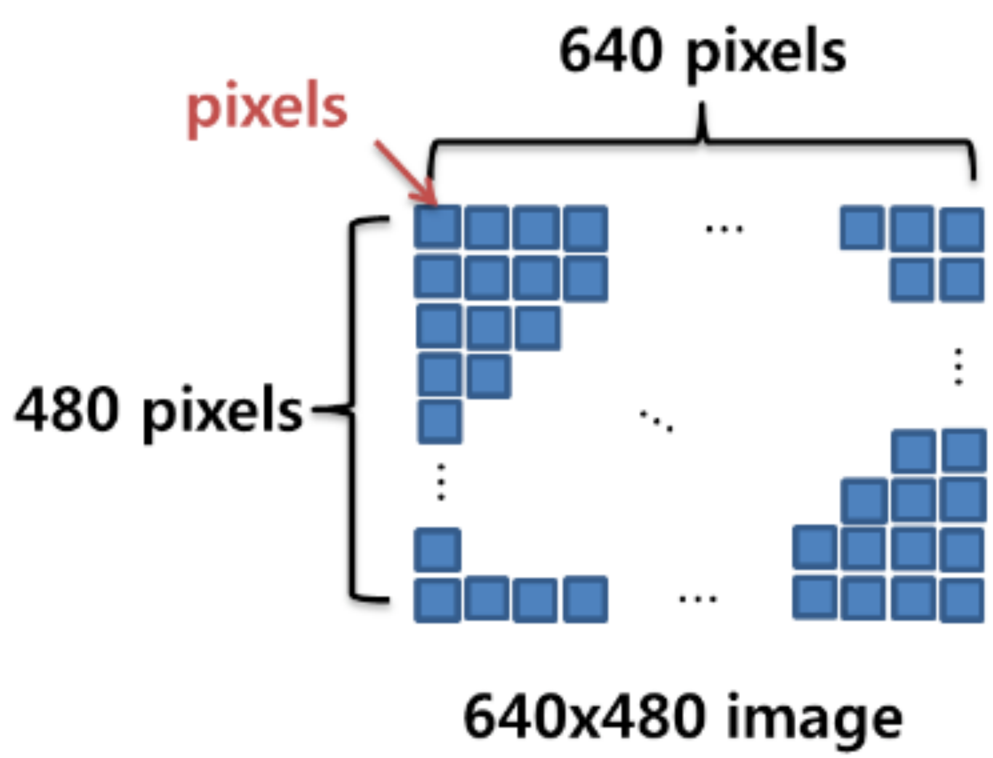
현운용

Goals

- 컴퓨터에서 이미지 처리
- 기초적인 이미지 분류 Computer vision
- Convolution neural network

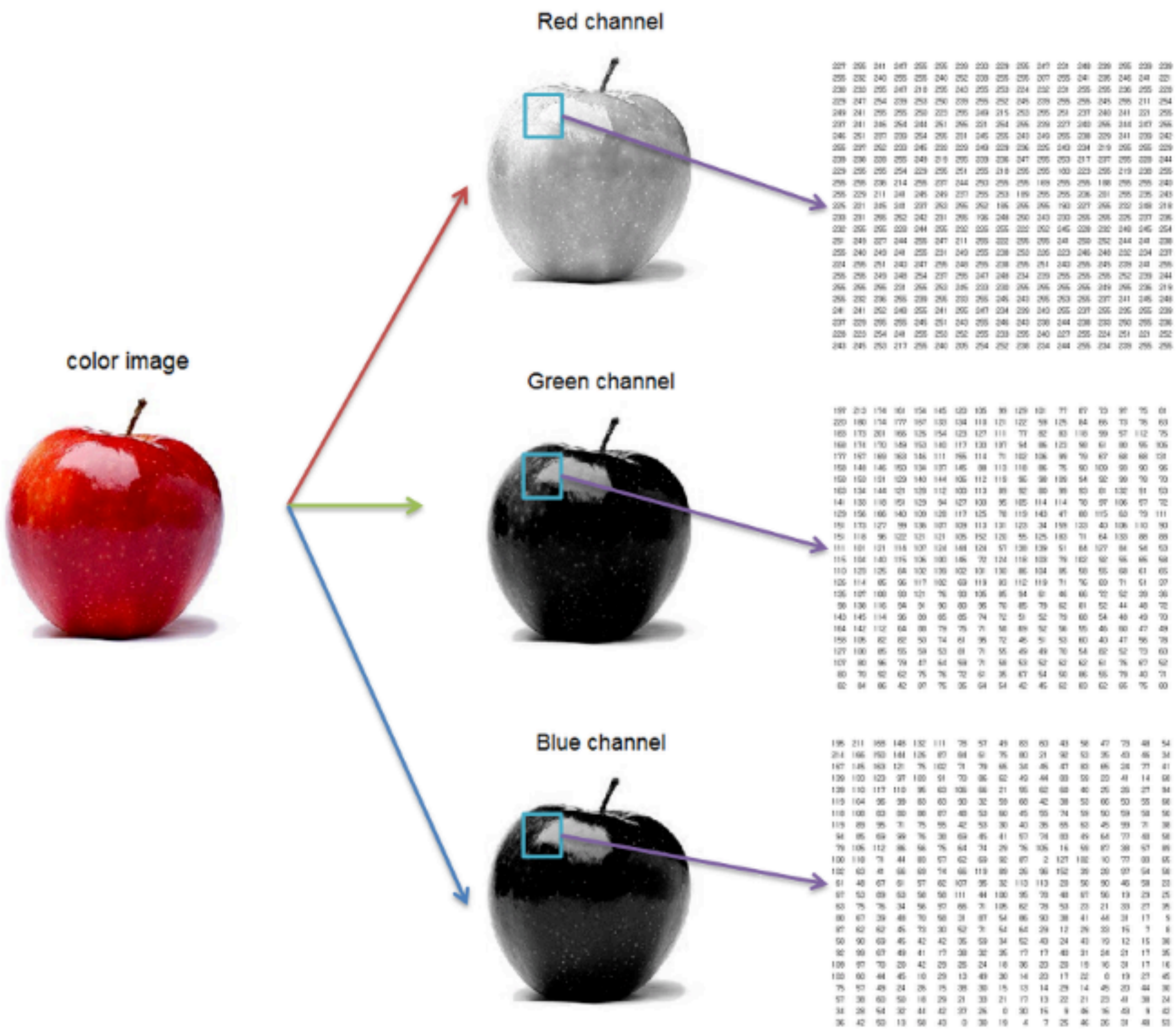
컴퓨터에서 이미지 처리

픽셀로 표현.

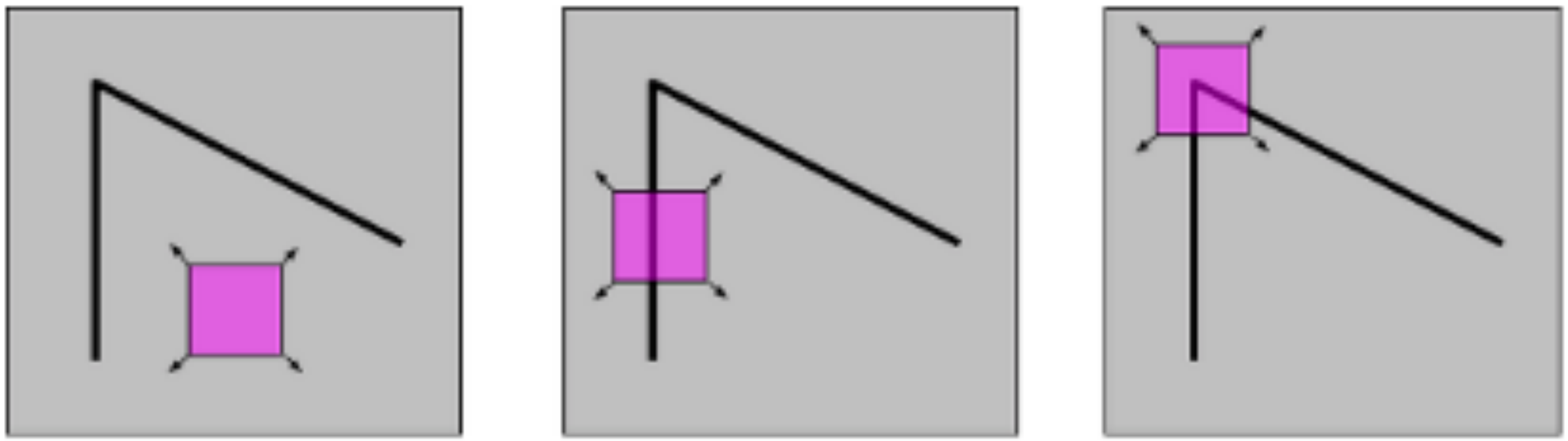


컴퓨터에서 이미지 처리

컴퓨터에서 이미지 처리



기초적인 이미지 분류



"flat" region:
no change in all
directions

"edge":
no change along the
edge direction

"corner":
significant change in
all directions

- 각 픽셀의 위치에 대해 윈도우를 수직, 수평, 좌대각선, 우대각선 이렇게 4개 방향으로 1픽셀씩 이동시켰을 때의 영상변화량(SSD) E를 계산한 후, E의 최소값을 해당 픽셀의 영상 변화량 값으로 설정…

기초적인 이미지 분류

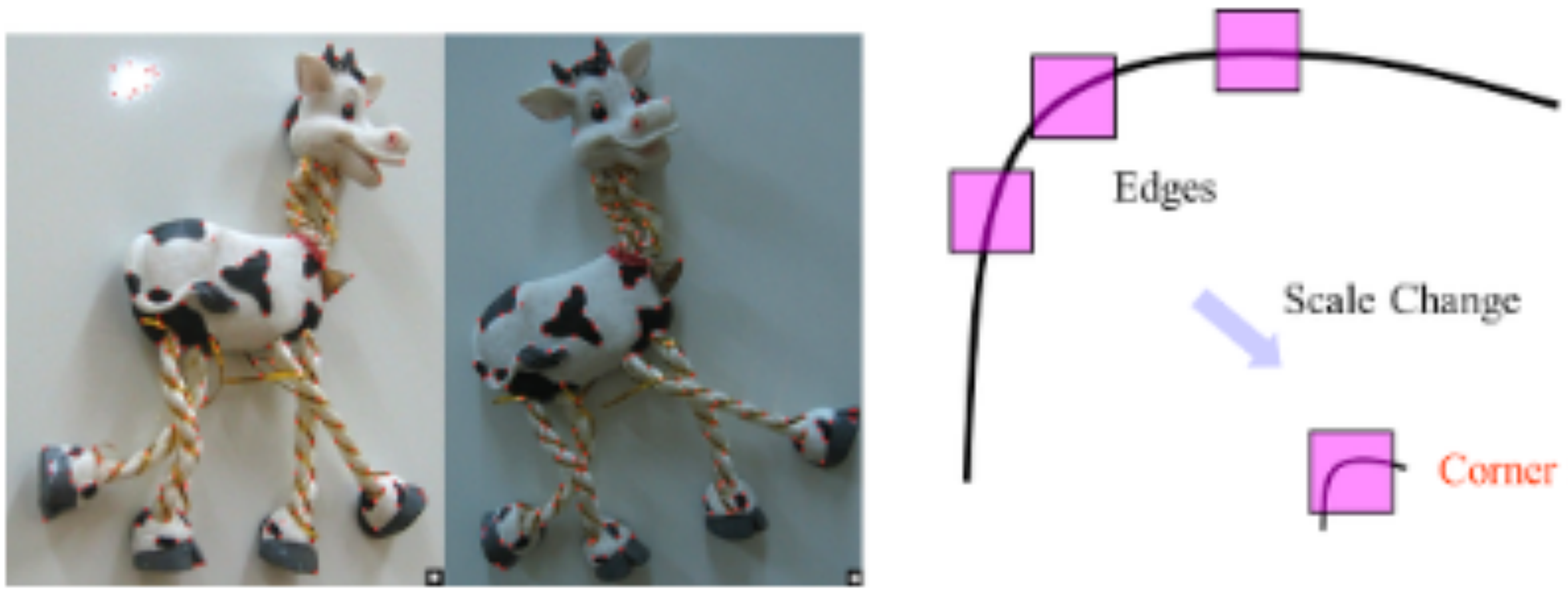
먼저, (△x, △y)만큼 윈도우를 이동시켰을 때 영상의 SSD(sum of squared difference) 변화량 E는 다음과 같습니다 (W: 로컬 윈도우).

$$E(\Delta x,\Delta y)=\sum_{W}[I(x_i+\Delta x,y_i+\Delta y)-I(x_i,y_i)]^2 \quad \text{--- (1)}$$

이 때, shift값 (△x,△y)이 매우 작다고 가정하고 그레디언트(gradient)를 이용하여 I를 선형 근사하면 (1차 테일러 근사),

$$\begin{aligned}
I(x_i+\Delta x,y_i+\Delta y) &\approx I(x_i,y_i)+\begin{bmatrix} I_x(x_i,y_i) & I_y(x_i,y_i) \end{bmatrix}\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \\
E(\Delta x,\Delta y) &= \sum_{W}[I(x_i+\Delta x,y_i+\Delta y)-I(x_i,y_i)]^2 \\
&\approx \sum_{W}\left[I(x_i,y_i)+\begin{bmatrix} I_x(x_i,y_i) & I_y(x_i,y_i) \end{bmatrix}\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}-I(x_i,y_i)\right]^2 \\
&= \begin{bmatrix} \Delta x & \Delta y \end{bmatrix}\begin{bmatrix} \sum_{W} I_x(x_i,y_i)^2 & \sum_{W} I_x(x_i,y_i)I_y(x_i,y_i) \\ \sum_{W} I_x(x_i,y_i)I_y(x_i,y_i) & \sum_{W} I_y(x_i,y_i)^2 \end{bmatrix}\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \\
&= \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} M \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \quad \text{--- (2)}
\end{aligned}$$

기초적인 이미지 분류



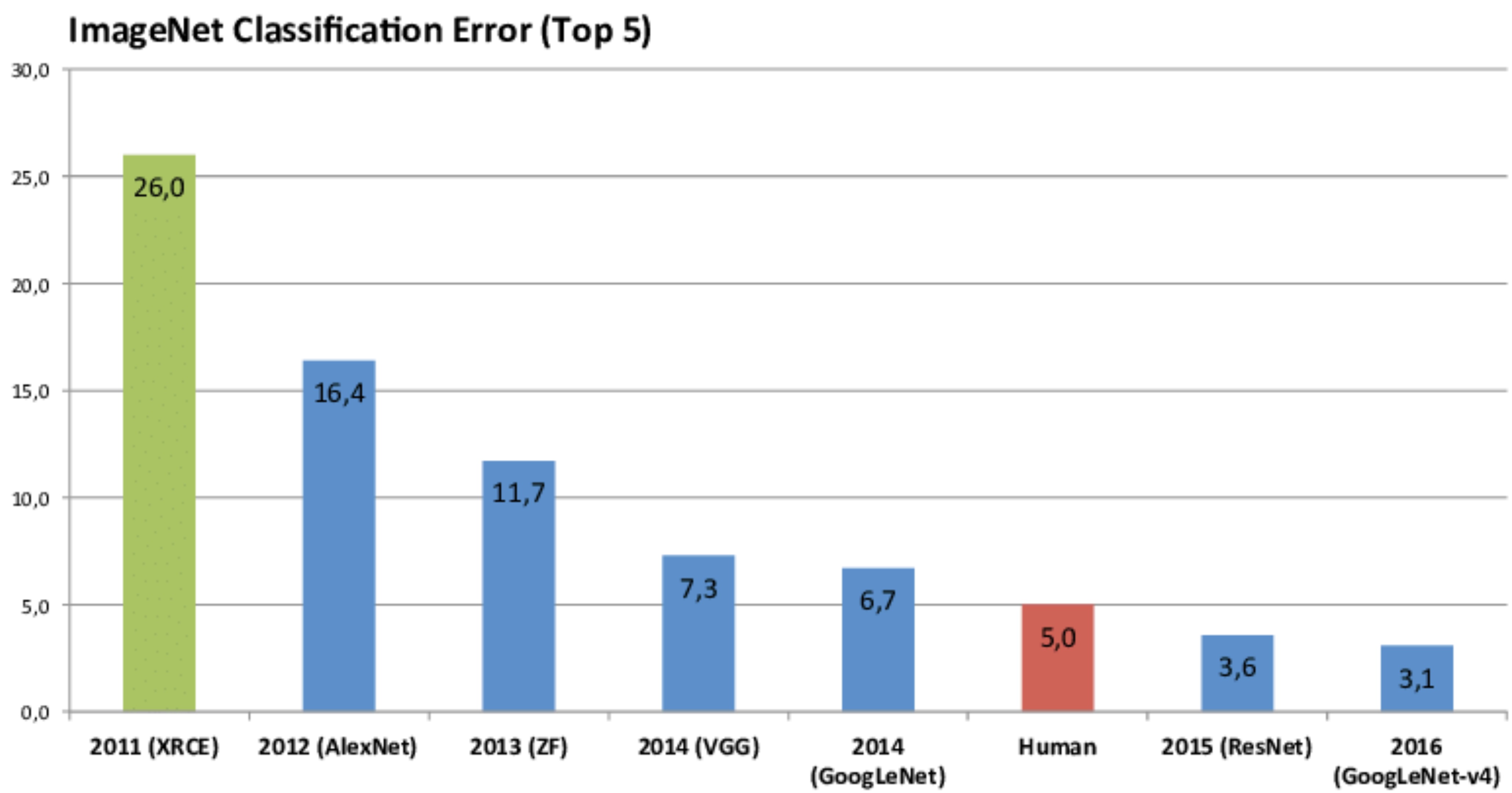
<그림 4> 출처: Matching with Invariant Features, Lecture Notes 2004

기초적인 이미지 분류

특징점을 수동으로 찾았음

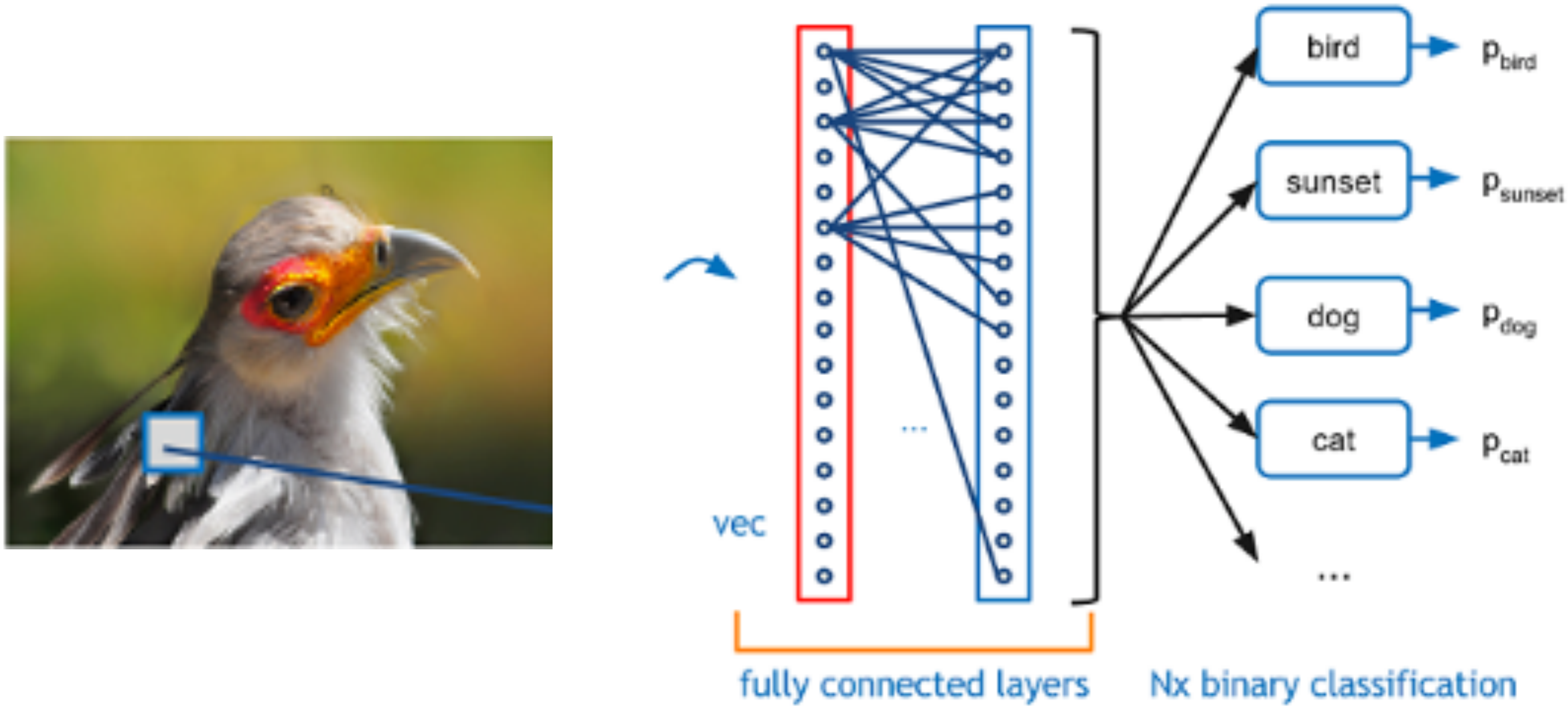
기초적인 이미지 분류

이런식의 분류가 오차율 26%



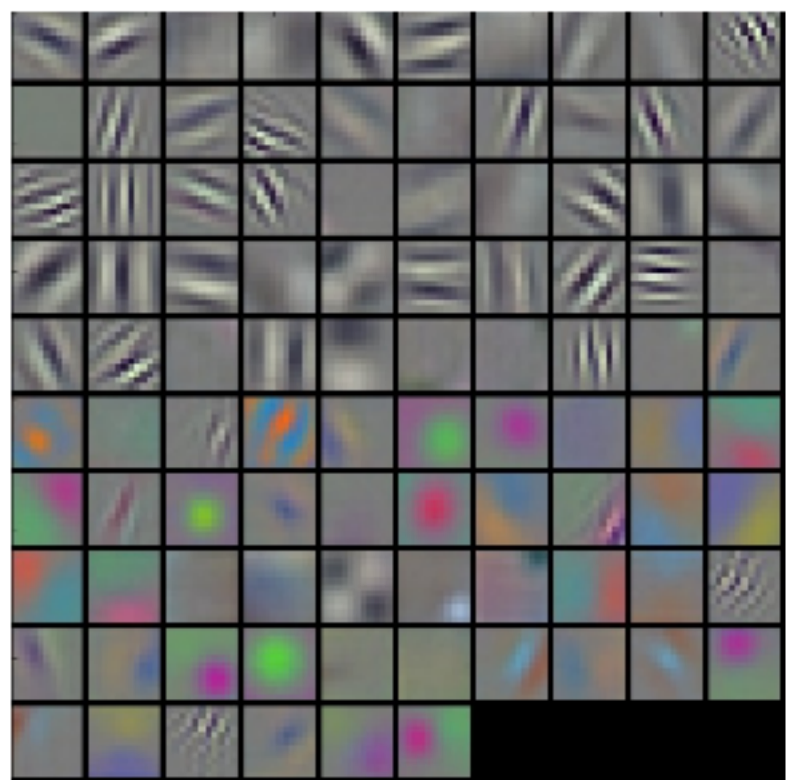
Convolution neural network

스스로 특징점을 찾는 프로그램 CNN



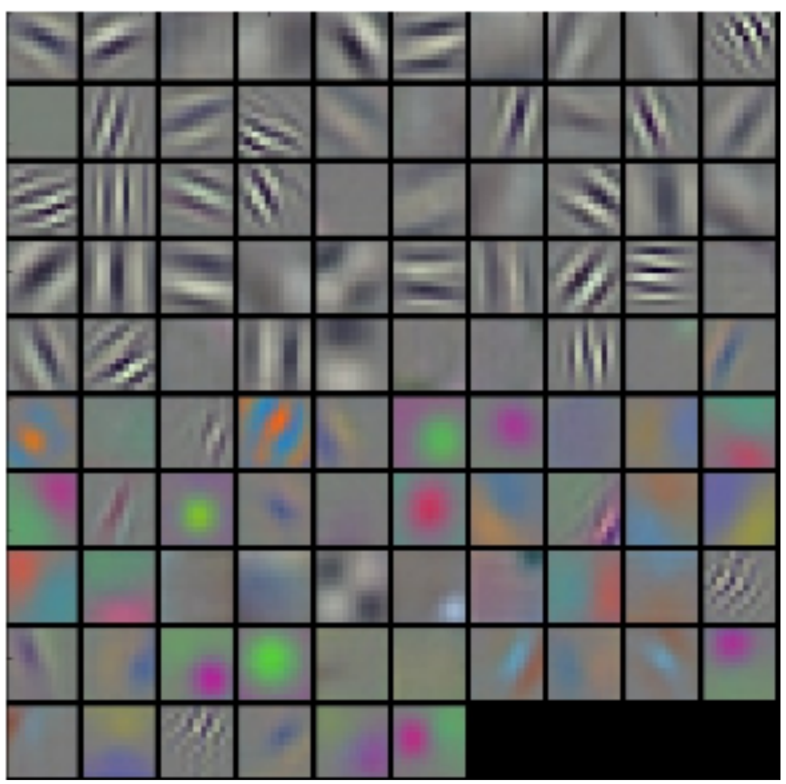
Convolution neural network

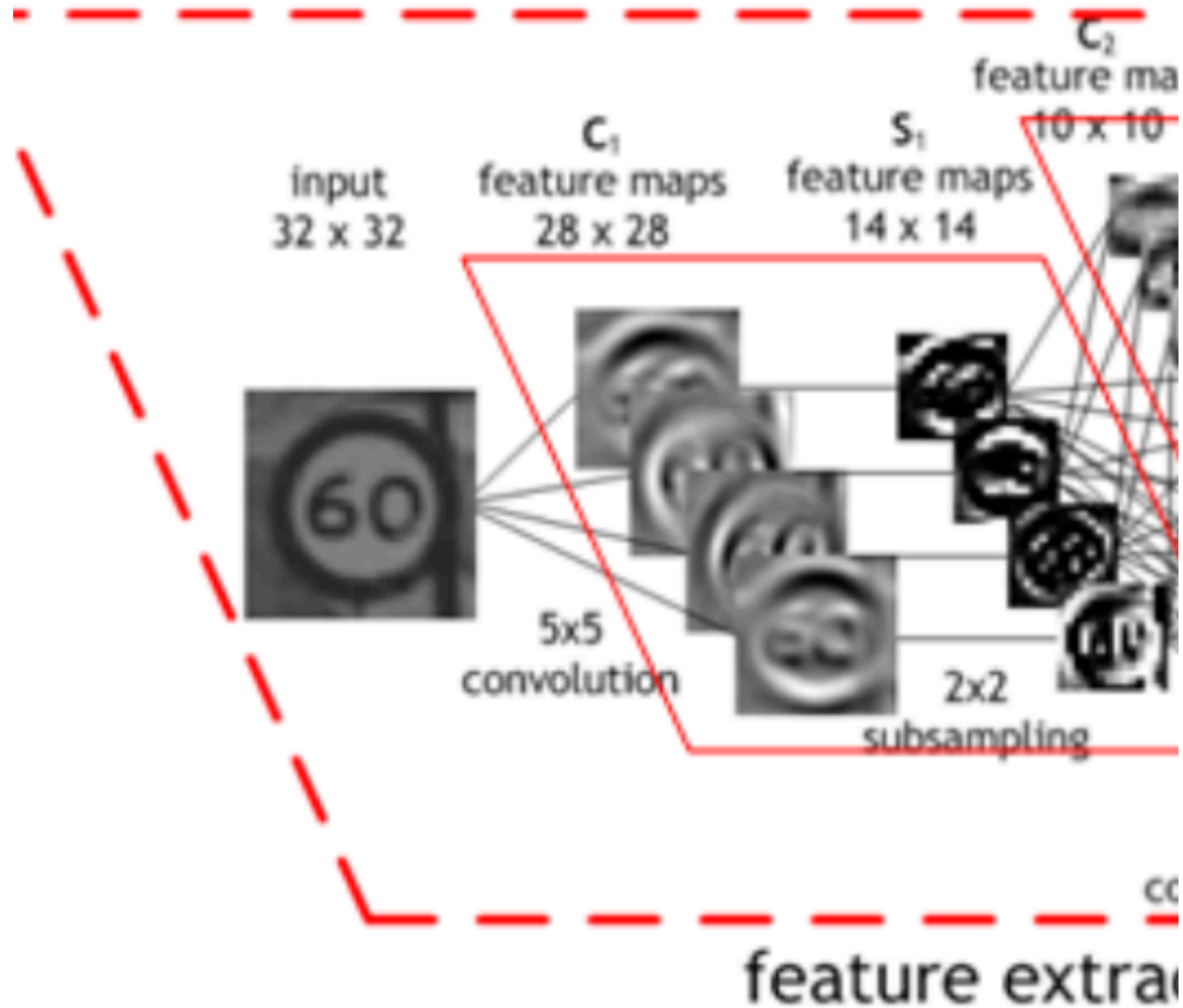
스스로 특징점을 찾는 프로그램 CNN



Convolution neural network

스스로 특징점을 찾는 프로그램 CNN



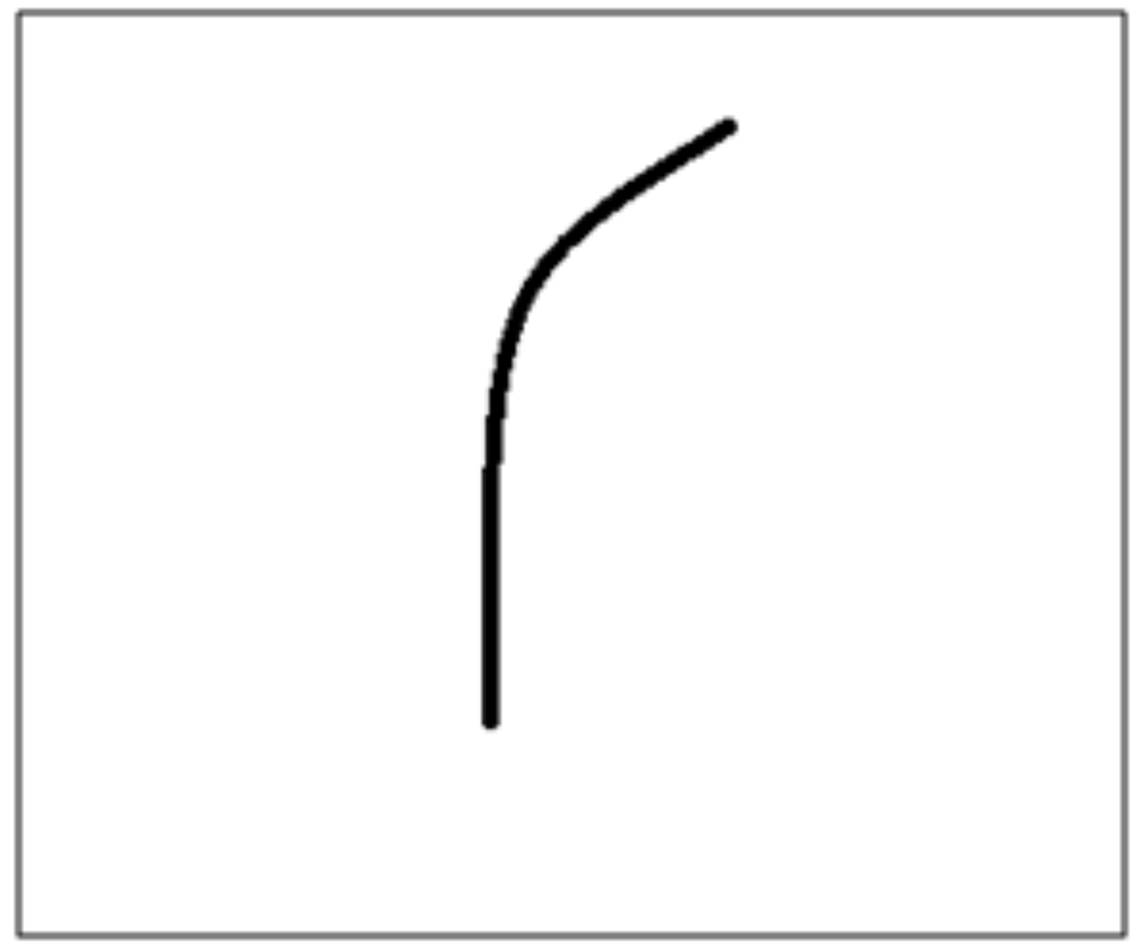


Convolution neural network

스스로 특징점을 찾는 프로그램 CNN

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

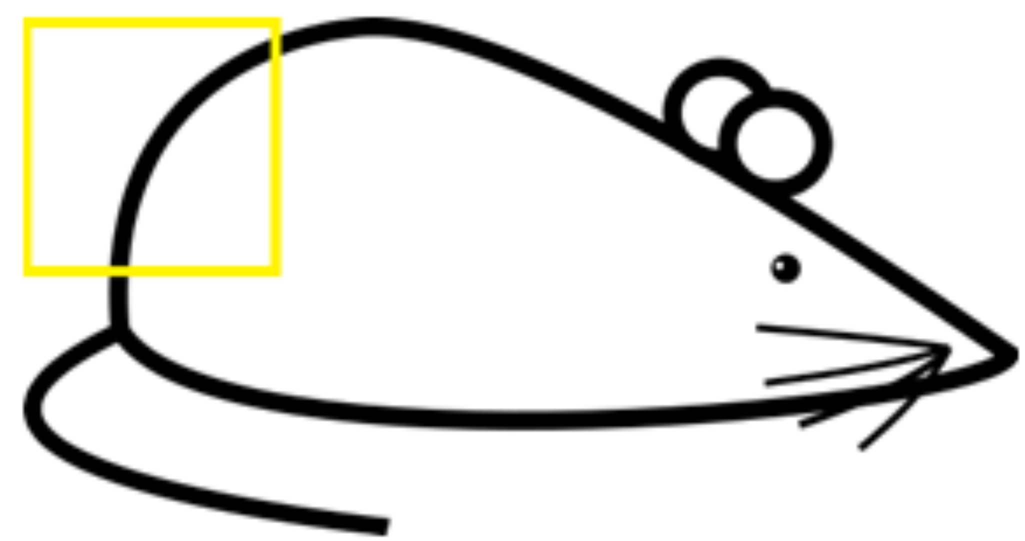


Visualization of a curve detector filter

Convolution neural network

스스로 특징점을 찾는 프로그램 CNN

Original image



Visualization of the filter on the image

Convolution neural network

스스로 특징점을 찾는 프로그램 CNN

Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

Convolution neural network

스스로 특징점을 찾는 프로그램 CNN

Visualization of the filter on the image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

Pixel representation of receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = 0

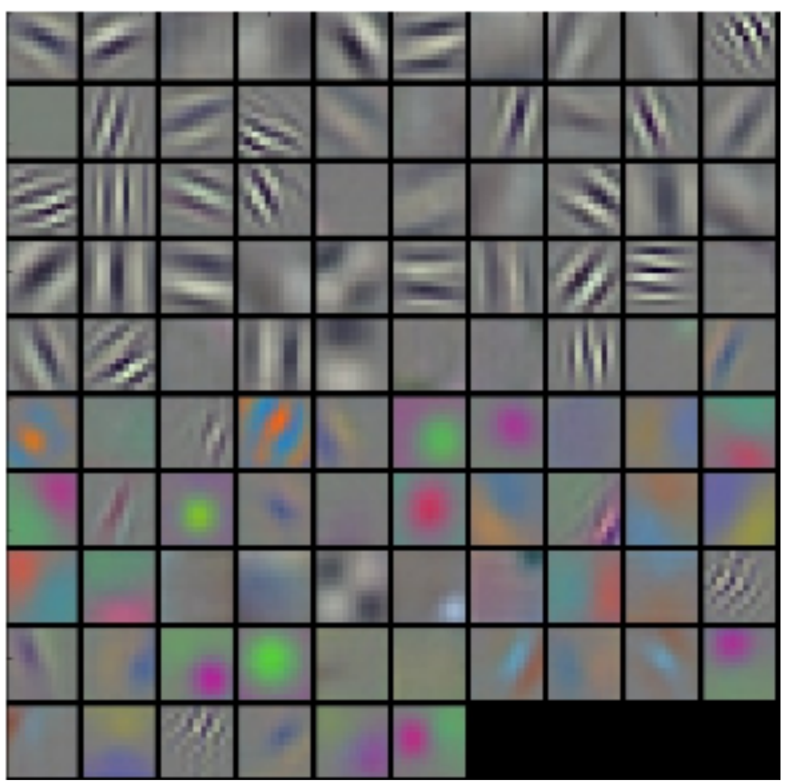
Convolution neural network

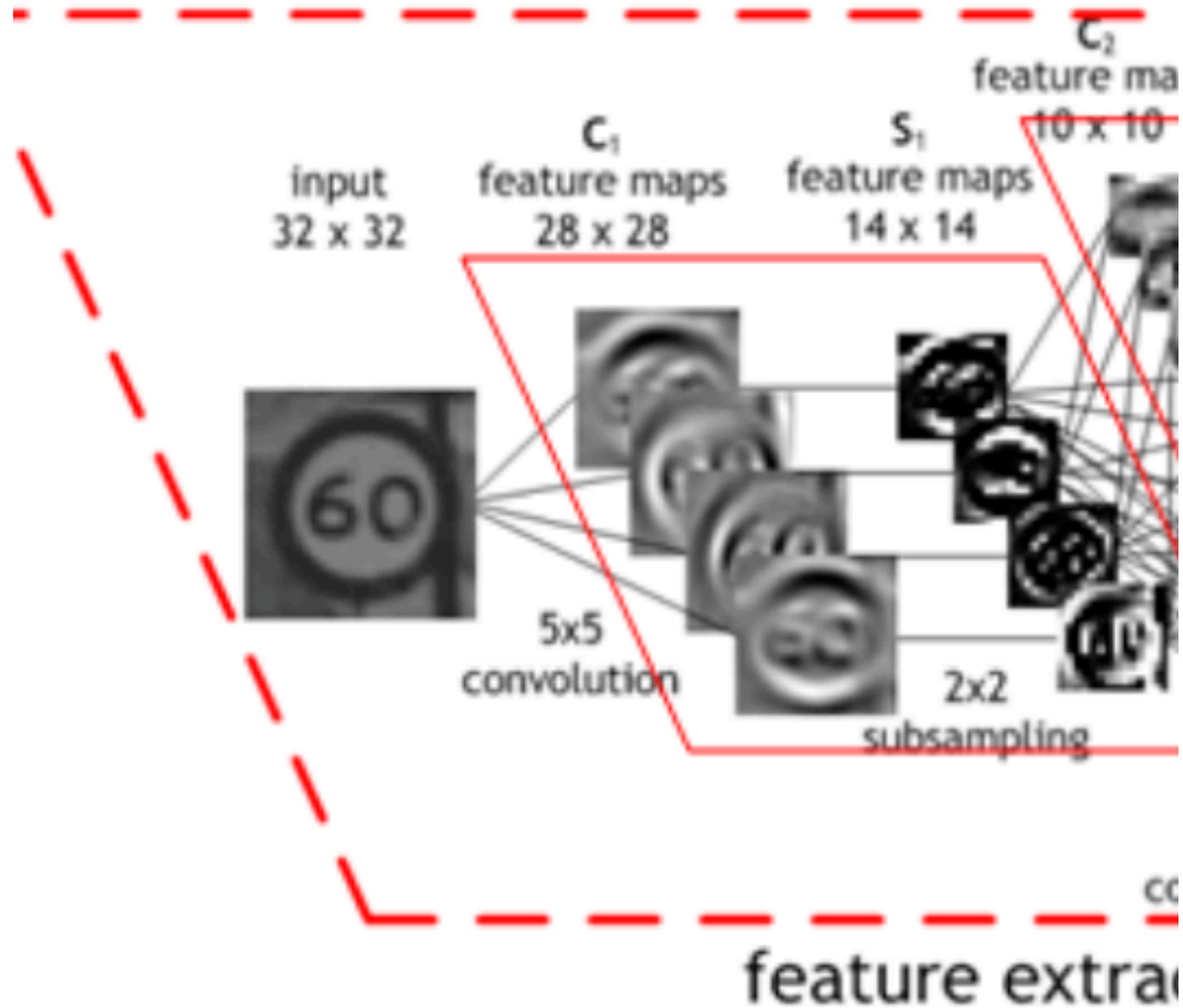
스스로 특징점을 찾는 프로그램 CNN

http://cs231n.github.io/convolutional-networks/

Convolution neural network

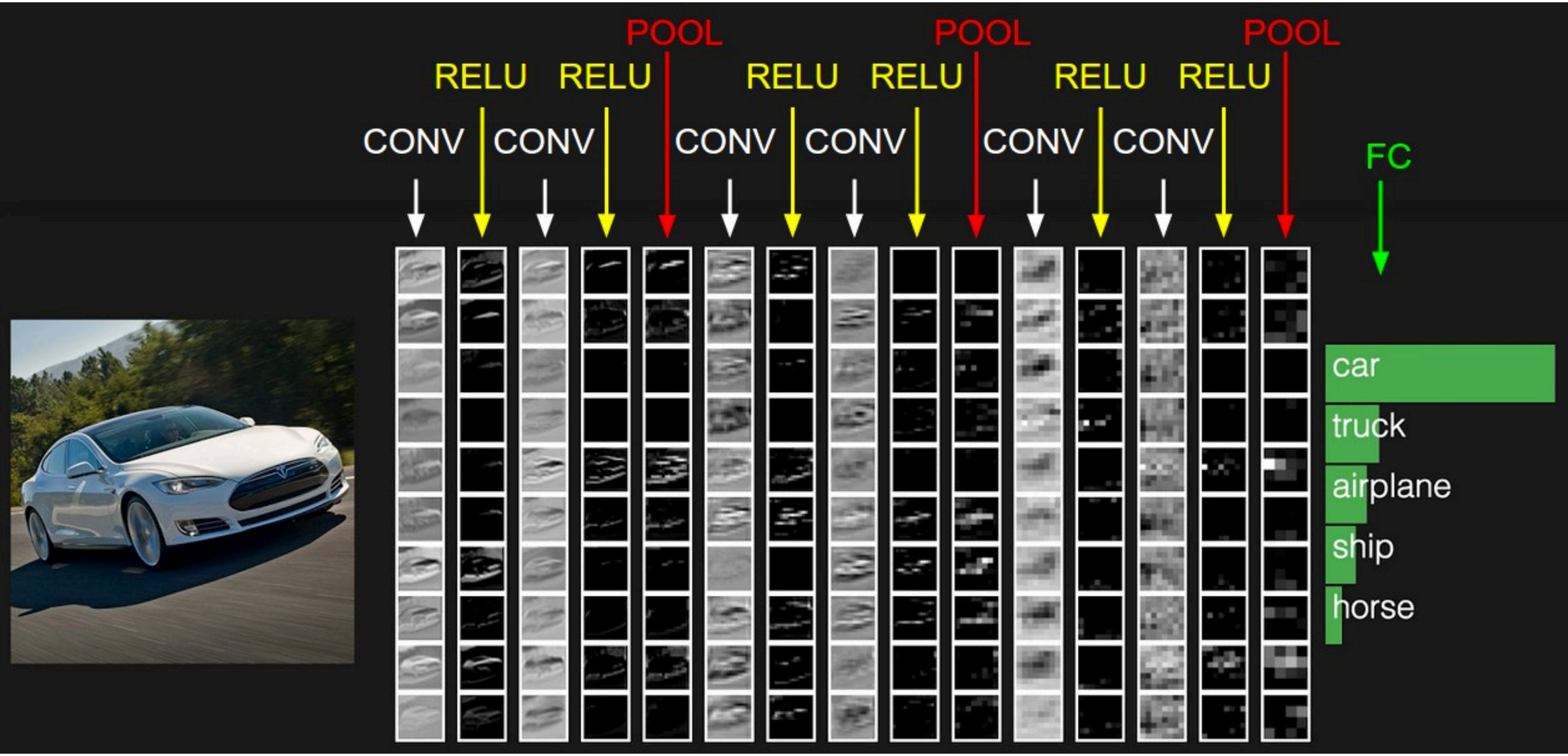
스스로 특징점을 찾는 프로그램 CNN





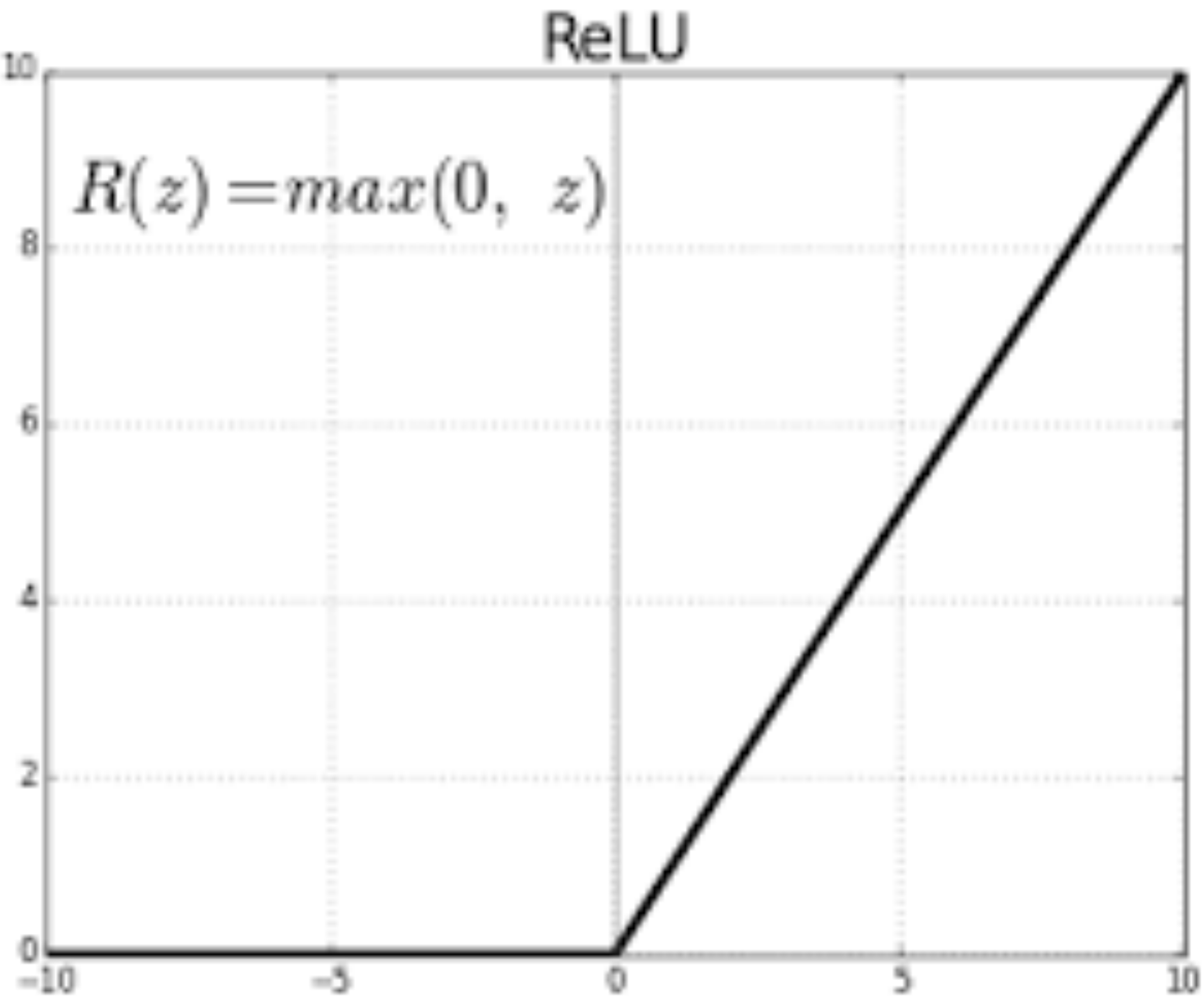
Convolution neural network

스스로 특징점을 찾는 프로그램 CNN



Convolution neural network

스스로 특징점을 찾는 프로그램 CNN



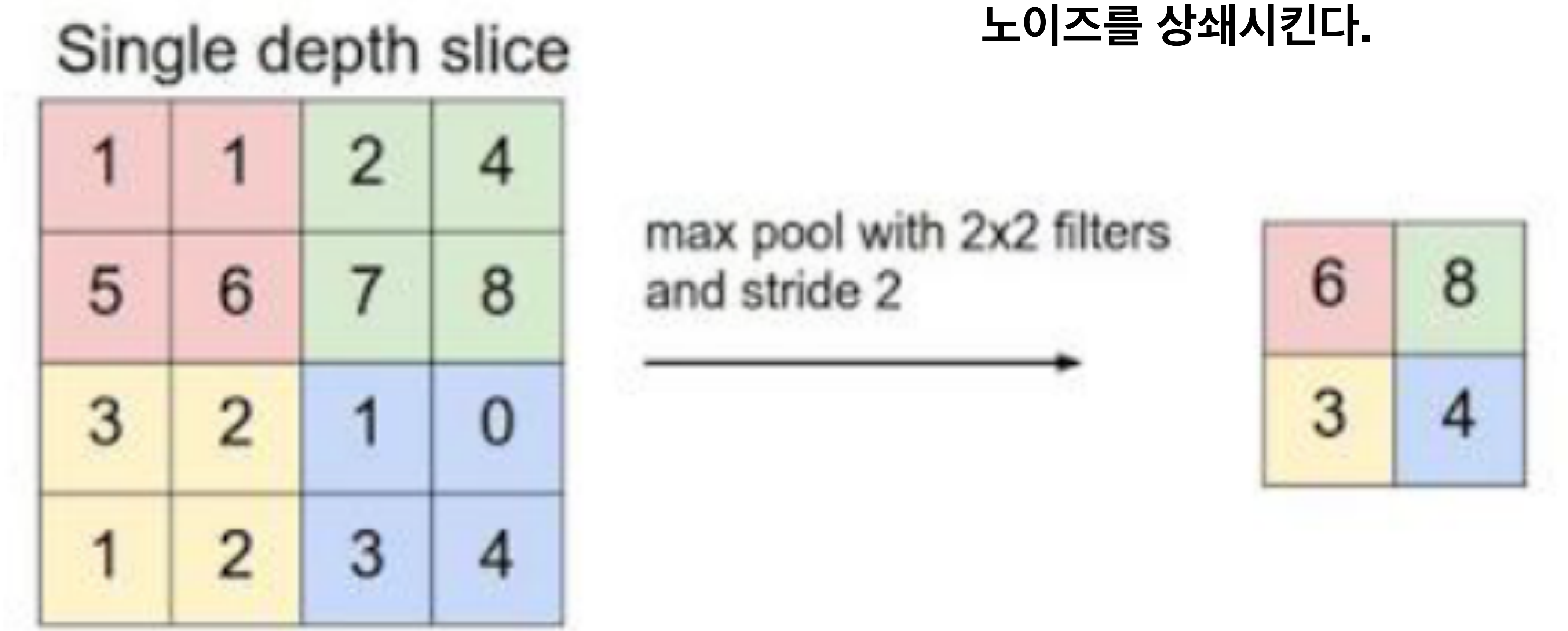
Convolution neural network

- CNN의 핵심 Convolution Layer 였습니다.
- CNN에는 Convolution Layer 말고도 사용되는 계층
  1. Max Pooling Layer
  2. Flatten Layer

Convolution neural network

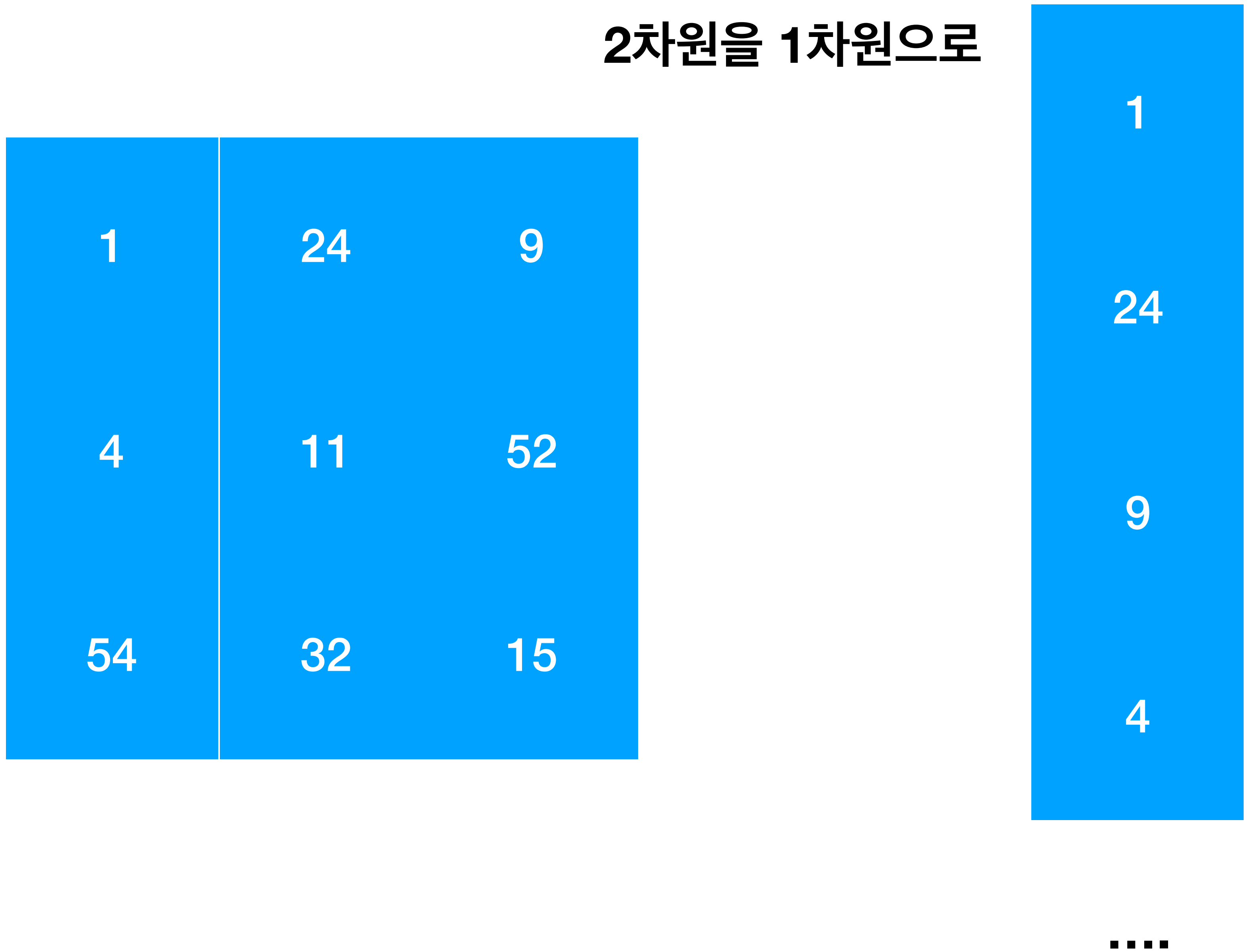
Max Poolin Layer

단순히 데이터의 사이즈를 줄여주는 것 뿐만 아니라,
노이즈를 상쇄시킨다.

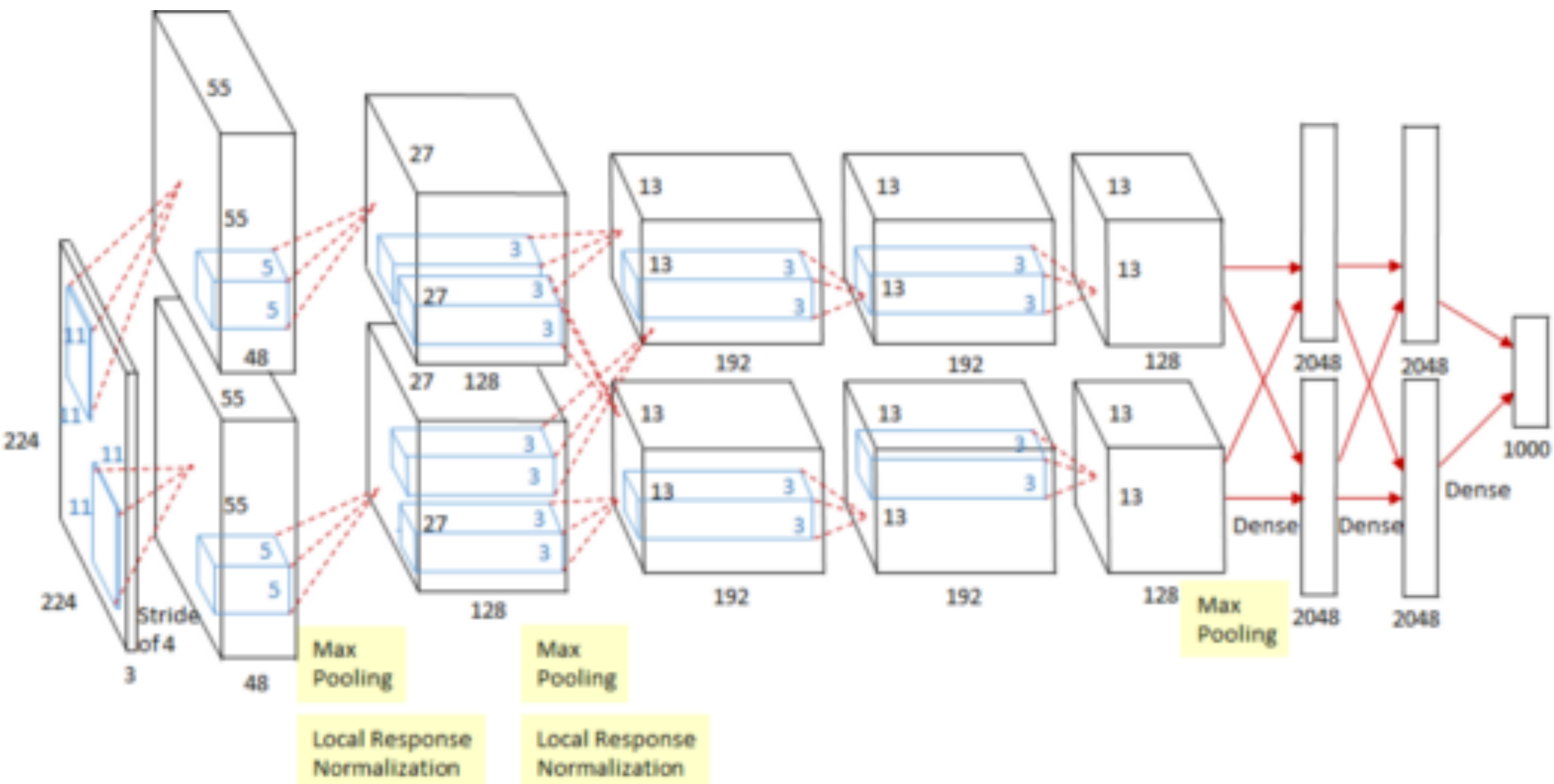


Convolution neural network

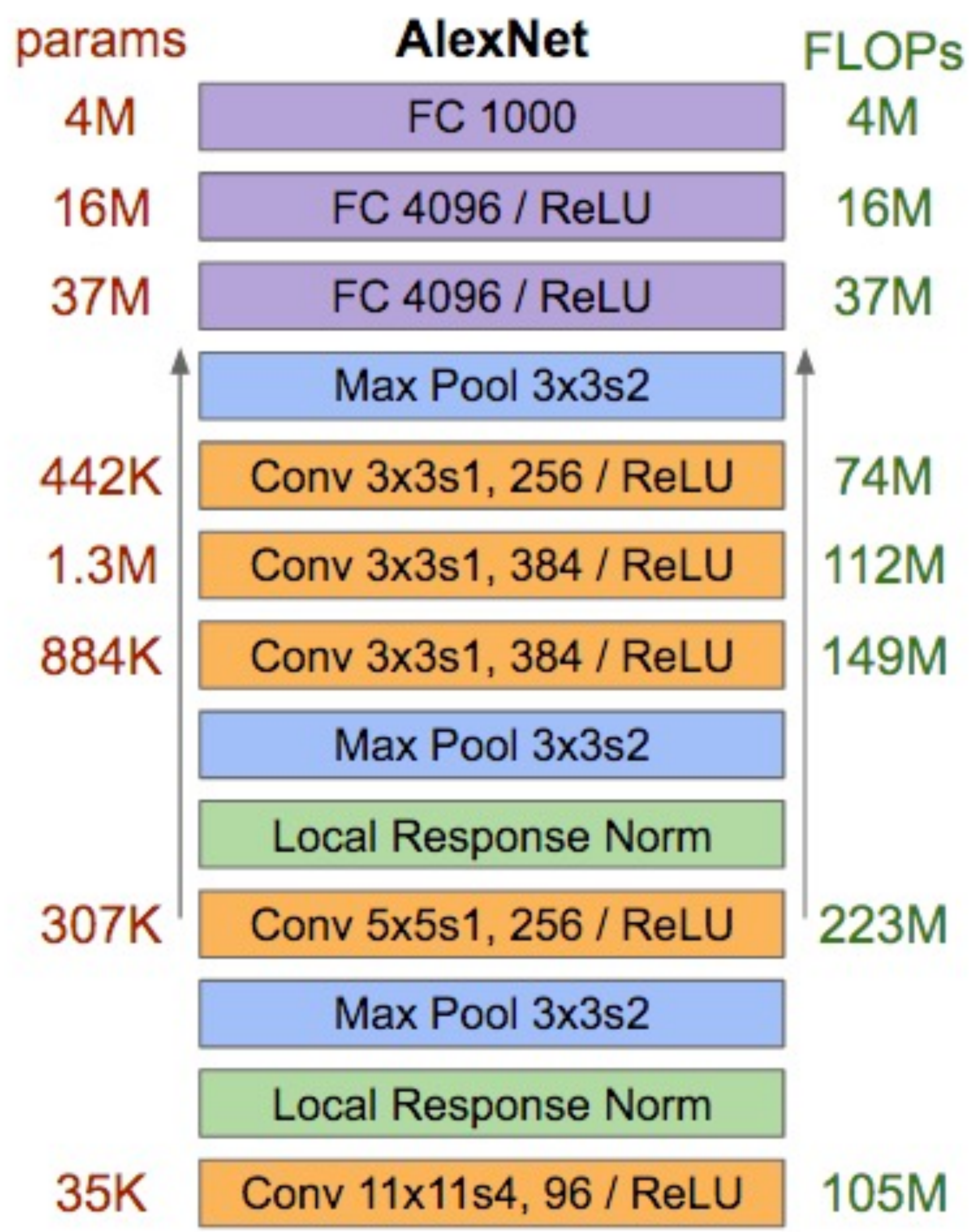
Flatton Layer



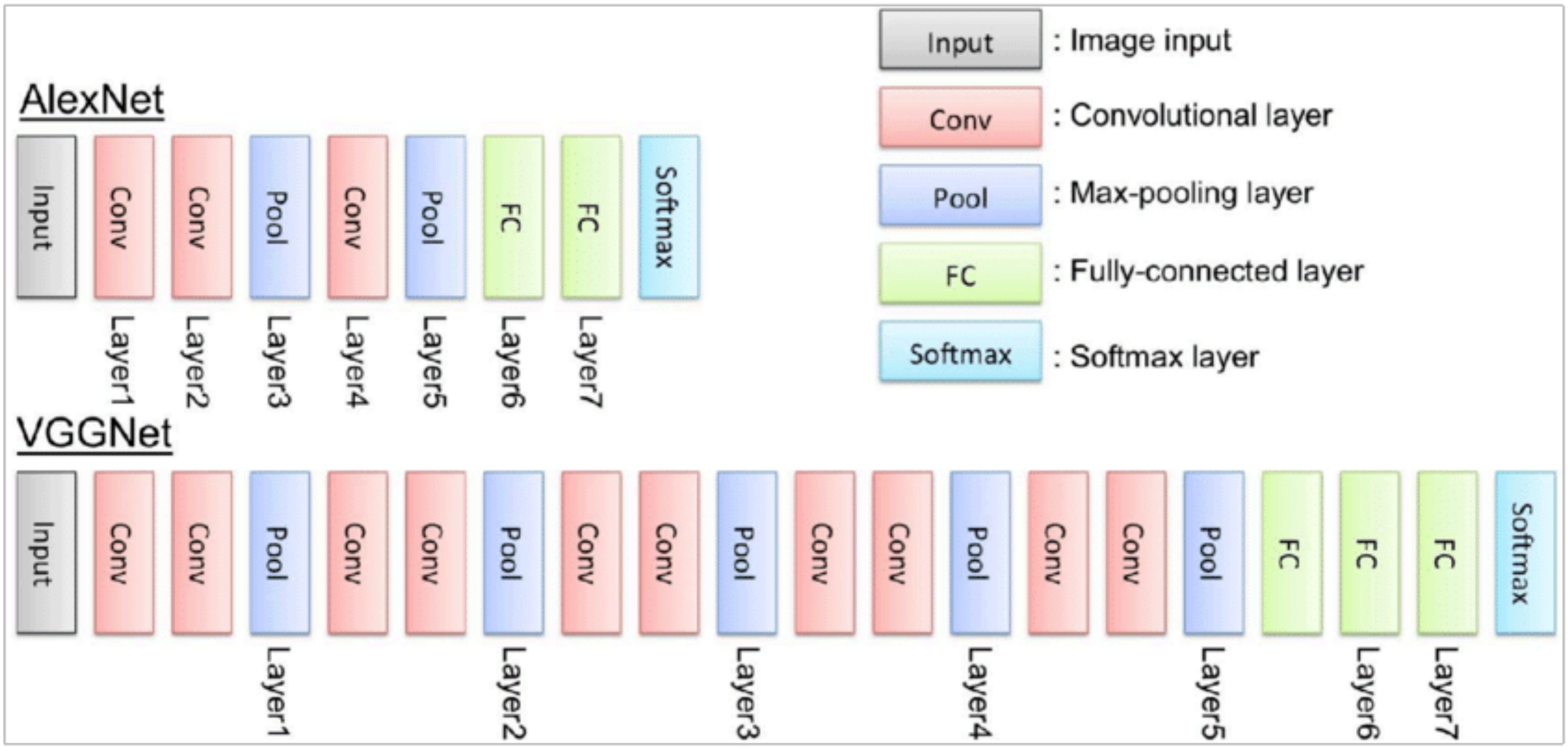
Convolution neural network



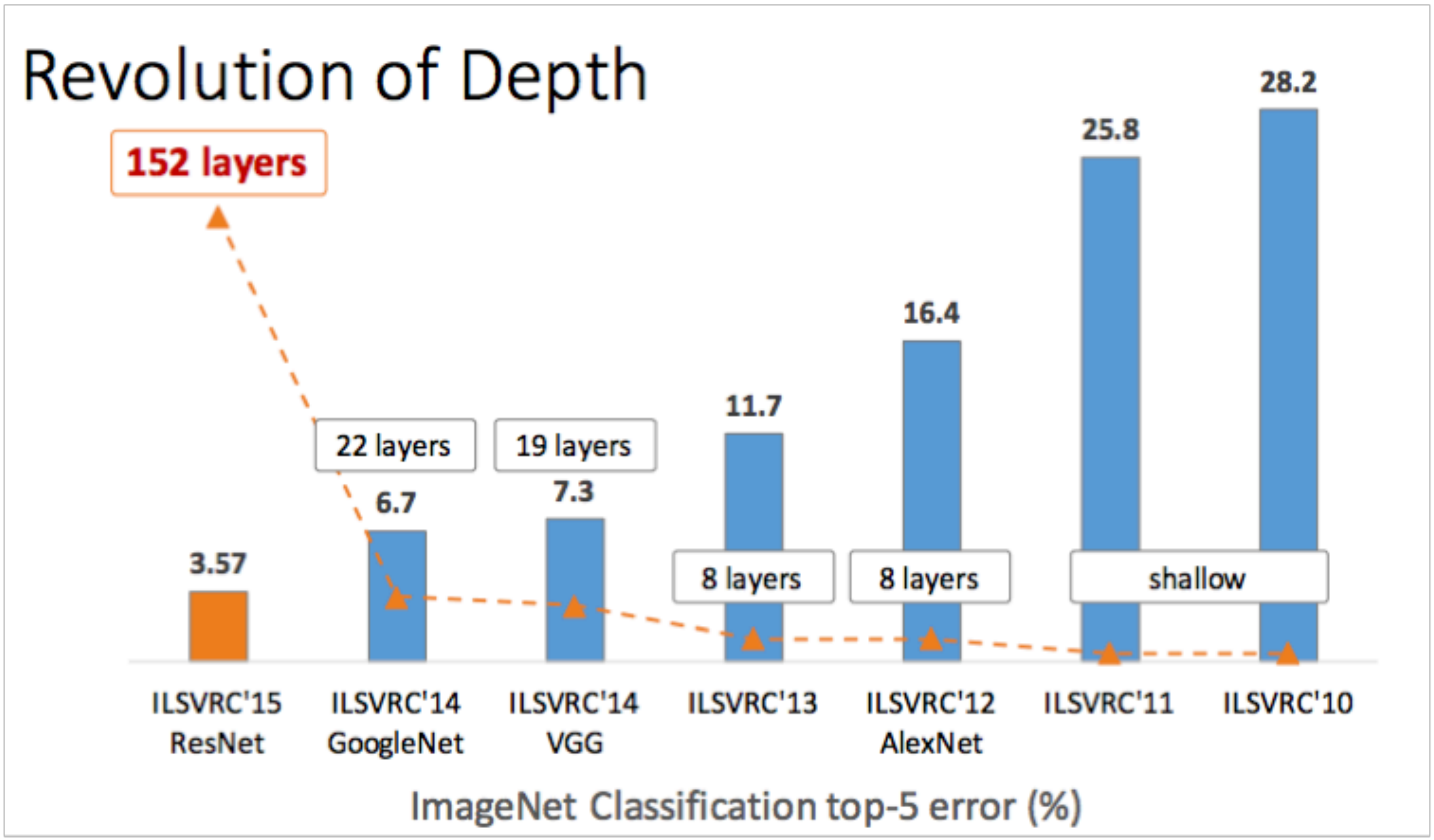
Convolution neural network



Convolution neural network



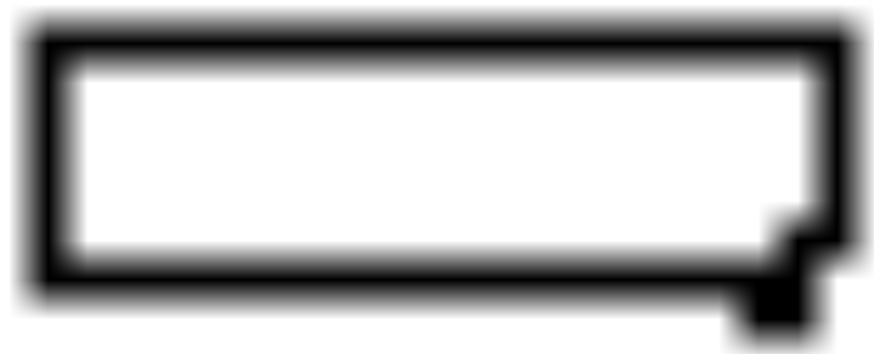
Convolution neural network



Convolution neural network

Triangle002.png

rectangle005.png



실습

```
conda install -c anaconda pillow
```

실습

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator
```

실습

```
train_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        './dataset/shape/train',
        target_size=(24, 24),
        batch_size=3,
        class_mode='categorical')

test_datagen = ImageDataGenerator(rescale=1./255)

test_generator = test_datagen.flow_from_directory(
        './dataset/shape/test',
        target_size=(24, 24),
        batch_size=3,
        class_mode='categorical')
```

실습

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=(24,24,3)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(3, activation='softmax'))
```

실습

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

실습

```
model.fit_generator(
        train_generator,
        steps_per_epoch=15,
        epochs=50,
        validation_data=test_generator,
        validation_steps=5)
```

실습

```
print("-- Evaluate --")
scores = model.evaluate_generator(test_generator, steps=5)
print("%s: %.2f%%" %(model.metrics_names[1], scores[1]*100))
```

실습

```
print("-- Predict --")
output = model.predict_generator(test_generator, steps=5)
np.set_printoptions(formatter={'float': lambda x: "{0:0.3f}".format(x)})
print(test_generator.class_indices)
print(output)
```

실습

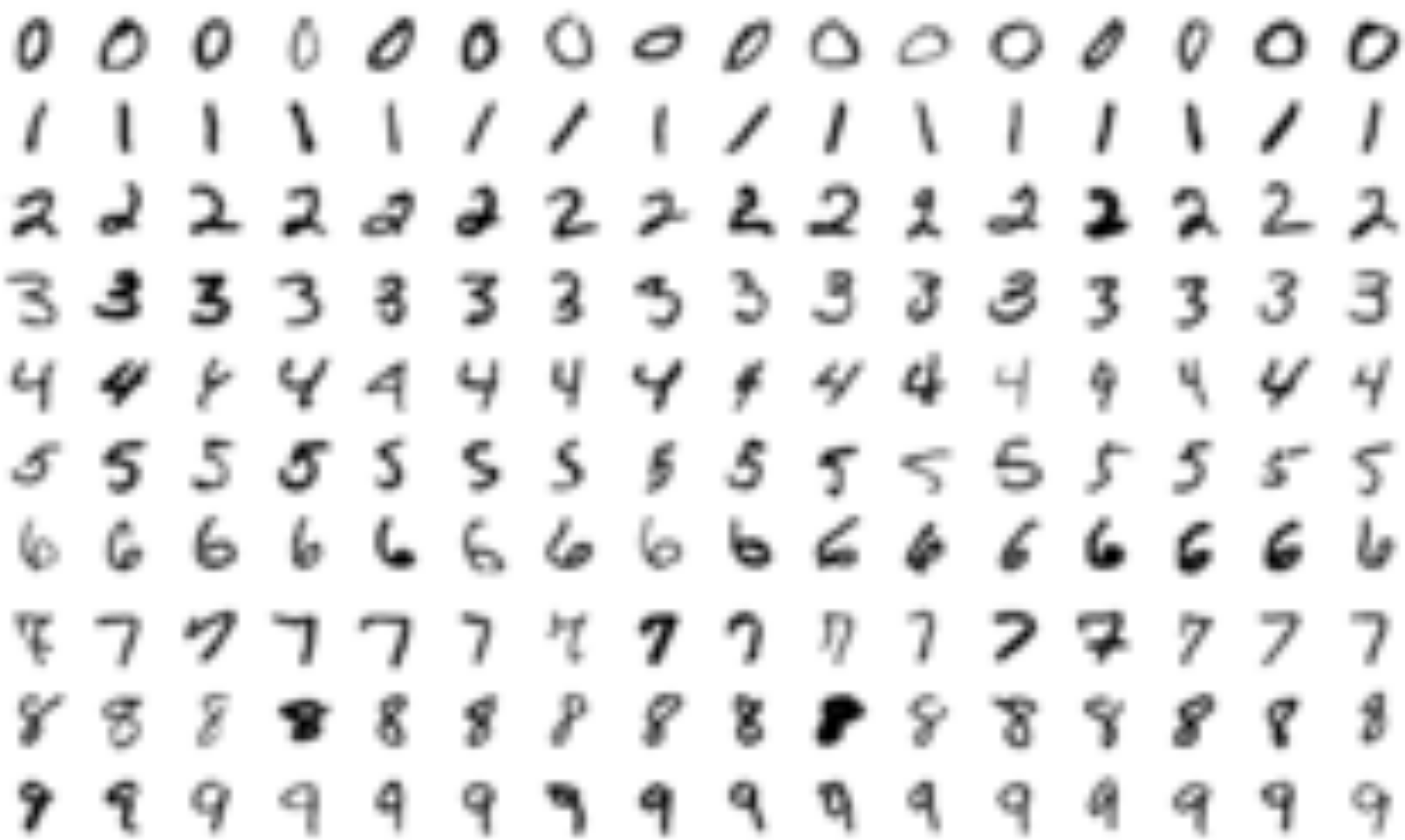
자기가 그린 도형과 좀 더 다르게 그림을 그려서 테스트한다면?

실습

과적합.

실습

MNist



실습

```python
# 영상 => 다중분류 컨볼루션
# 0. 사용할 패키지 불러오기
import numpy as np
from keras.utils import np_utils
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Activation
from keras.layers import Conv2D, MaxPooling2D, Flatten

width = 28
height = 28
```

실습

```
# 1. 데이터셋 생성하기

# 훈련셋과 시험셋 불러오기
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train = x_train.reshape(60000, width, height, 1).astype('float32') / 255.0
x_test = x_test.reshape(10000, width, height, 1).astype('float32') / 255.0

# 훈련셋과 검증셋 분리
x_val = x_train[50000:]
y_val = y_train[50000:]
x_train = x_train[:50000]
y_train = y_train[:50000]

# 데이터셋 전처리 : one-hot 인코딩
y_train = np_utils.to_categorical(y_train)
y_val = np_utils.to_categorical(y_val)
y_test = np_utils.to_categorical(y_test)
```

실습

```
# 2. 모델 구성하기
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(width, height, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dense(10, activation='softmax'))
```

실습

```
# 3. 모델 학습과정 설정하기
model.compile(loss='categorical_crossentropy', optimizer='sgd', metrics=['accuracy'])

# 4. 모델 학습시키기
hist = model.fit(x_train, y_train, epochs=3, batch_size=32, validation_data=(x_val, y_val))
```

```python
# 5. 학습과정 살펴보기
%matplotlib inline
import matplotlib.pyplot as plt

fig, loss_ax = plt.subplots()

acc_ax = loss_ax.twinx()

loss_ax.plot(hist.history['loss'], 'y', label='train loss')
loss_ax.plot(hist.history['val_loss'], 'r', label='val loss')
loss_ax.set_ylim([0.0, 0.5])

acc_ax.plot(hist.history['acc'], 'b', label='train acc')
acc_ax.plot(hist.history['val_acc'], 'g', label='val acc')
acc_ax.set_ylim([0.8, 1.0])

loss_ax.set_xlabel('epoch')
loss_ax.set_ylabel('loss')
acc_ax.set_ylabel('accuray')

loss_ax.legend(loc='upper left')
acc_ax.legend(loc='lower left')

plt.show()
```

실습

```
# 6. 모델 평가하기
loss_and_metrics = model.evaluate(x_test, y_test, batch_size=32)
print('## evaluation loss and_metrics ##')
print(loss_and_metrics)

# 7. 모델 사용하기
yhat_test = model.predict(x_test, batch_size=32)
```

```python
# 7. 모델 사용하기
yhat_test = model.predict(x_test, batch_size=32)

%matplotlib inline
import matplotlib.pyplot as plt

plt_row = 5
plt_col = 5

plt.rcParams["figure.figsize"] = (10,10)

f, axarr = plt.subplots(plt_row, plt_col)

cnt = 0
i = 0

while cnt < (plt_row*plt_col):

    if np.argmax(y_test[i]) == np.argmax(yhat_test[i]):
        i += 1
        continue

    sub_plt = axarr[int(cnt/plt_row), int(cnt%plt_col)]
    sub_plt.axis('off')
    sub_plt.imshow(x_test[i].reshape(width, height))
    sub_plt_title = 'R: ' + str(np.argmax(y_test[i])) + ' P: ' + str(np.argmax(yhat_test[i]))
    sub_plt.set_title(sub_plt_title)

    i += 1
    cnt += 1

plt.show()
```