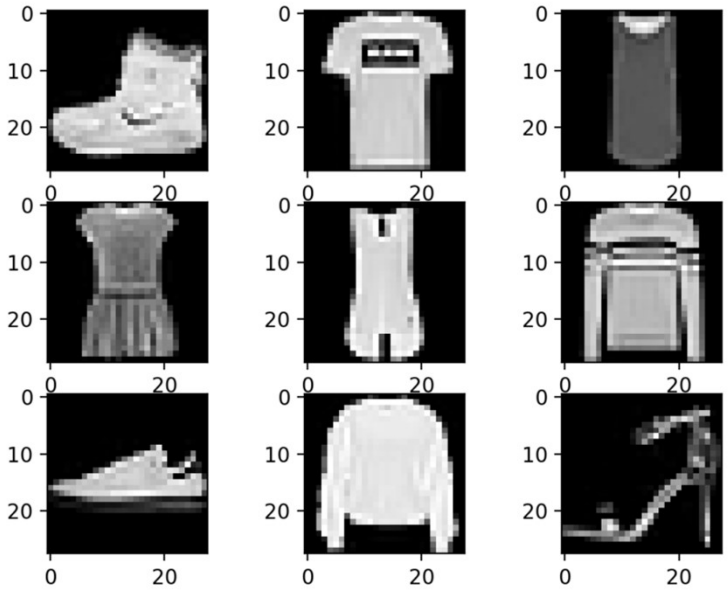
Image Classification

Byunghwan Jeon, Ph D

Fashion MNIST dataset

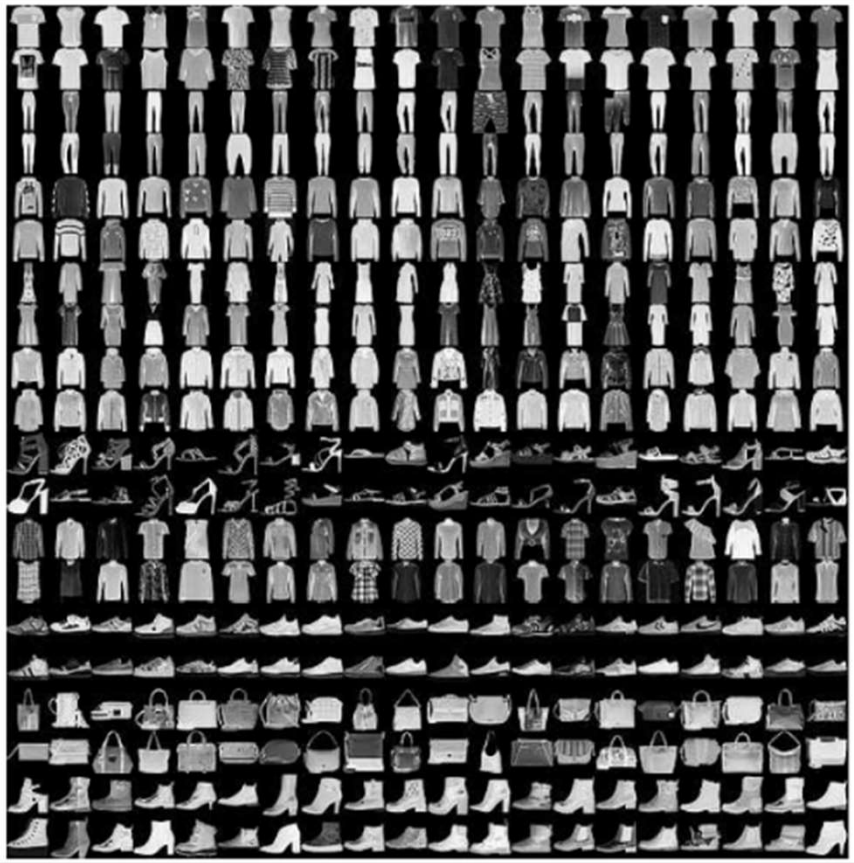
Fashion MNIST 데이터셋 소개

- 70000개의 이미지
- 10개의 클래스
- 이미지 크기 28x28



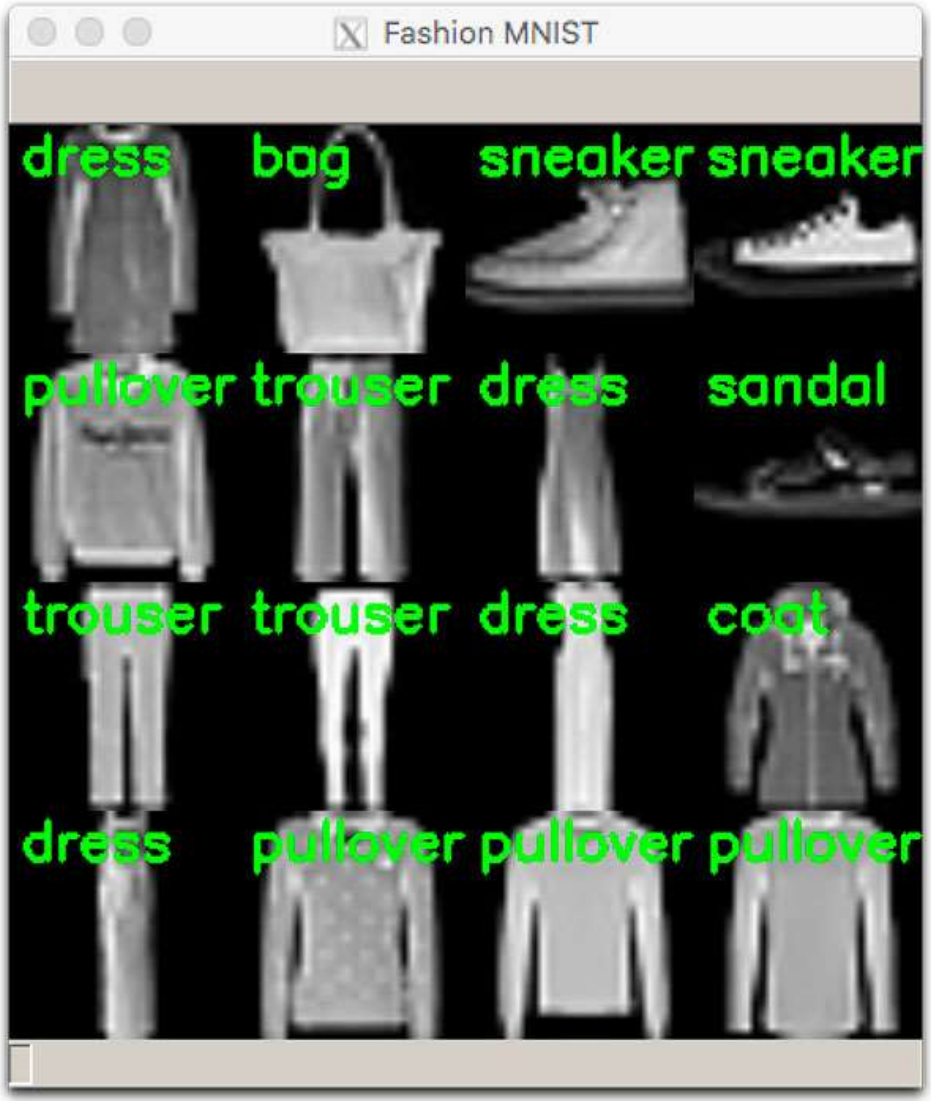
Fashion MNIST 데이터/정답 정보

Fashion-MNIST is a fashion product image dataset for benchmarking machine learning algorithms for computer vision. This dataset comprises 60,000 28×28 training images and 10,000 28×28 test images, including 10 categories of fashion products. Figure 1 shows all the labels and some images in **Fashion-MNIST**.

Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	

Fashion MNIST 데이터셋 소개

- 0-255의 그레이 스케일 값을 가지고 있음
- 정답 태깅이 되어 있음.
- Training, Test 데이터셋으로 분리되어 있음.



API를 통하여 데이터 가져오기

60000장

10000장

훈련데이터
Training data

시험데이터
(test data)

API를 통하여 데이터 가져오기


```
fashion_mnist = tf.keras.datasets.fashion_mnist
```
데이터 가져오기

```
(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
```
훈련용 (x_train, y_train) 시험용 (x_test, y_test)

데이터 정답 데이터 정답 데이터 불러오기

이미지는 그저 숫자 덩어리일 뿐

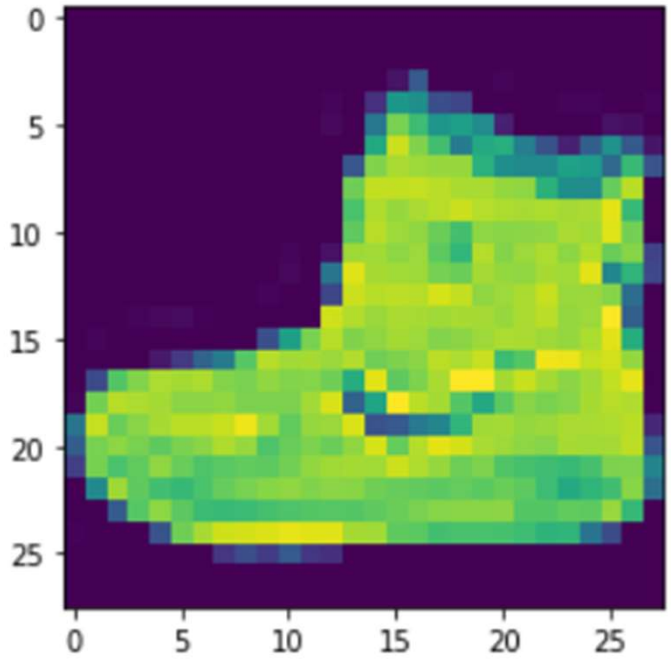
한국외국어대학교 HANKUK UNIVERSITY OF FOREIGN STUDIES

Print (x_train[0]) 60000장의 훈련데이터 중 0번째 데이터의 픽셀들

```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0  13  73   0   0   1   4   0   0   0   0   1   1   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   3   0  36 136 127  62  54   0   0   0   1   3   4   0   0   3]
 [  0   0   0   0   0   0   0   0   0   0   0   0   6   0 102 204 176 134 144 123  23   0   0   0   0  12  10   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 155 236 207 178 107 156 161 109  64  23  77 130  72  15]
 [  0   0   0   0   0   0   0   0   0   0   0   1   0  69 207 223 218 216 216 163 127 121 122 146 141  88 172  66]
 [  0   0   0   0   0   0   0   0   0   1   1   1   0 200 232 232 233 229 223 223 215 213 164 127 123 196 229   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 183 225 216 223 228 235 227 224 222 224 221 223 245 173   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 193 228 218 213 198 180 212 210 211 213 223 220 243 202   0]
 [  0   0   0   0   0   0   0   0   0   1   3   0  12 219 220 212 218 192 169 227 208 218 224 212 226 197 209  52]
 [  0   0   0   0   0   0   0   0   0   0   6   0  99 244 222 220 218 203 198 221 215 213 222 220 245 119 167  56]
 [  0   0   0   0   0   0   0   0   0   4   0   0  55 236 228 230 228 240 232 213 218 223 234 217 217 209  92   0]
 [  0   0   1   4   6   7   2   0   0   0   0   0 237 226 217 223 222 219 222 221 216 223 229 215 218 255  77   0]
 [  0   3   0   0   0   0   0   0   0  62 145 204 228 207 213 221 218 208 211 218 224 223 219 215 224 244 159   0]
 [  0   0   0   0  18  44  82 107 189 228 220 222 217 226 200 205 211 230 224 234 176 188 250 248 233 238 215   0]
 [  0  57 187 208 224 221 224 208 204 214 208 209 200 159 245 193 206 223 255 255 221 234 221 211 220 232 246   0]
 [  3 202 228 224 221 211 211 214 205 205 205 220 240  80 150 255 229 221 188 154 191 210 204 209 222 228 225   0]
 [ 98 233 198 210 222 229 229 234 249 220 194 215 217 241  65  73 106 117 168 219 221 215 217 223 223 224 229  29]
 [ 75 204 212 204 193 205 211 225 216 185 197 206 198 213 240 195 227 245 239 223 218 212 209 222 220 221 230  67]
 [ 48 203 183 194 213 197 185 190 194 192 202 214 219 221 220 236 225 216 199 206 186 181 177 172 181 205 206 115]
 [  0 122 219 193 179 171 183 196 204 210 213 207 211 210 200 196 194 191 195 191 198 192 176 156 167 177 210  92]
 [  0   0  74 189 212 191 175 172 175 181 185 188 189 188 193 198 204 209 210 210 211 188 188 194 192 216 170   0]
 [  2   0   0   0  66 200 222 237 239 242 246 243 244 221 220 193 191 179 182 182 181 176 166 168  99  58   0   0]
 [  0   0   0   0   0   0   0  40  61  44  72  41  35   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]]
```

숫자들을 이미지로 가시화 해보기


```
import tensorflow as tf
import numpy as np
import matplotlib.image as img
import matplotlib.pyplot as pp

fashion_mnist = tf.keras.datasets.fashion_mnist
(x_train, y_train),(x_test, y_test) = fashion_mnist.load_data()
pp.imshow(x_train[0])
```

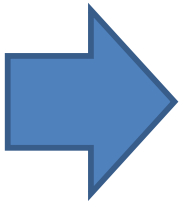


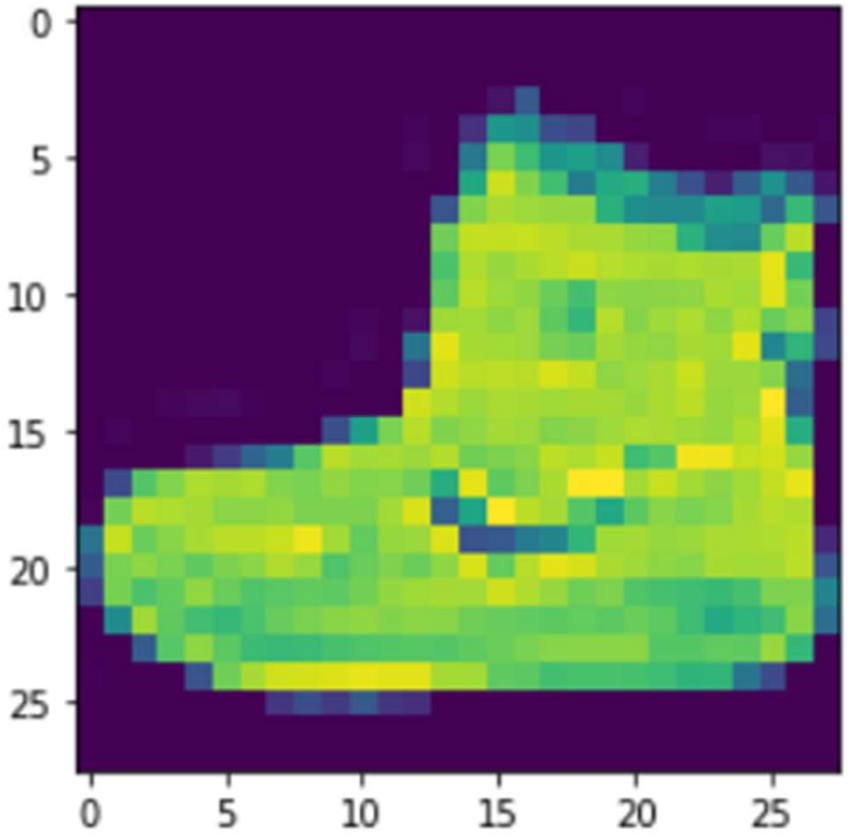
숫자들을 이미지로 가시화 해보기

한국외국어대학교
HANKUK UNIVERSITY OF FOREIGN STUDIES

```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0  13  73   0   0   1   4   0   0   0   0   1   1   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   3   0  36 136 127  62  54   0   0   0   1   3   4   0   0   3]
 [  0   0   0   0   0   0   0   0   0   0   0   0   6   0 102 204 176 134 144 123  23   0   0   0   0  12  10   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 155 236 207 178 107 156 161 109  64  23  77 130  72  15]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0  69 207 223 218 216 216 163 127 121 122 146 141  88 172  66]
 [  0   0   0   0   0   0   0   0   0   0   1   1   1   0 200 232 232 233 229 223 223 215 213 164 127 123 196 229   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 183 225 216 223 228 235 227 224 222 224 221 223 245 173   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 193 228 218 213 198 180 212 210 211 213 223 220 243 202   0]
 [  0   0   0   0   0   0   0   0   0   0   1   3   0  12 219 220 212 218 192 169 227 208 218 224 212 226 197 209  52]
 [  0   0   0   0   0   0   0   0   0   0   0   6   0  99 244 222 220 218 203 198 221 215 213 222 220 245 119 167  56]
 [  0   0   0   0   0   0   0   0   0   0   4   0   0  55 236 228 230 228 240 232 213 218 223 234 217 217 209  92   0]
 [  0   0   1   4   6   7   2   0   0   0   0   0   0 237 226 217 223 222 219 222 221 216 223 229 215 218 255  77   0]
 [  0   3   0   0   0   0   0   0   0   0  62 145 204 228 207 213 221 218 208 211 218 224 223 219 215 224 244 159   0]
 [  0   0   0   0  18  44  82 107 189 228 220 222 217 226 200 205 211 230 224 234 176 188 250 248 233 238 215   0]
 [  0  57 187 208 224 221 224 208 204 214 208 209 200 159 245 193 206 223 255 255 221 234 221 211 220 232 246   0]
 [  3 202 228 224 221 211 211 214 205 205 205 220 240  80 150 255 229 221 188 154 191 210 204 209 222 228 225   0]
 [ 98 233 198 210 222 229 229 234 249 220 194 215 217 241  65  73 106 117 168 219 221 215 217 223 223 224 229  29]
 [ 75 204 212 204 193 205 211 225 216 185 197 206 198 213 240 195 227 245 239 223 218 212 209 222 220 221 230  67]
 [ 48 203 183 194 213 197 185 190 194 192 202 214 219 221 220 236 225 216 199 206 186 181 177 172 181 205 206 115]
 [  0 122 219 193 179 171 183 196 204 210 213 207 211 210 200 196 194 191 195 191 198 192 176 156 167 177 210  92]
 [  0   0  74 189 212 191 175 172 175 181 185 188 189 188 193 198 204 209 210 210 211 188 188 194 192 216 170   0]
 [  2   0   0   0  66 200 222 237 239 242 246 243 244 221 220 193 191 179 182 182 181 176 166 168  99  58   0   0]
 [  0   0   0   0   0   0   0  40  61  44  72  41  35   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]]
```

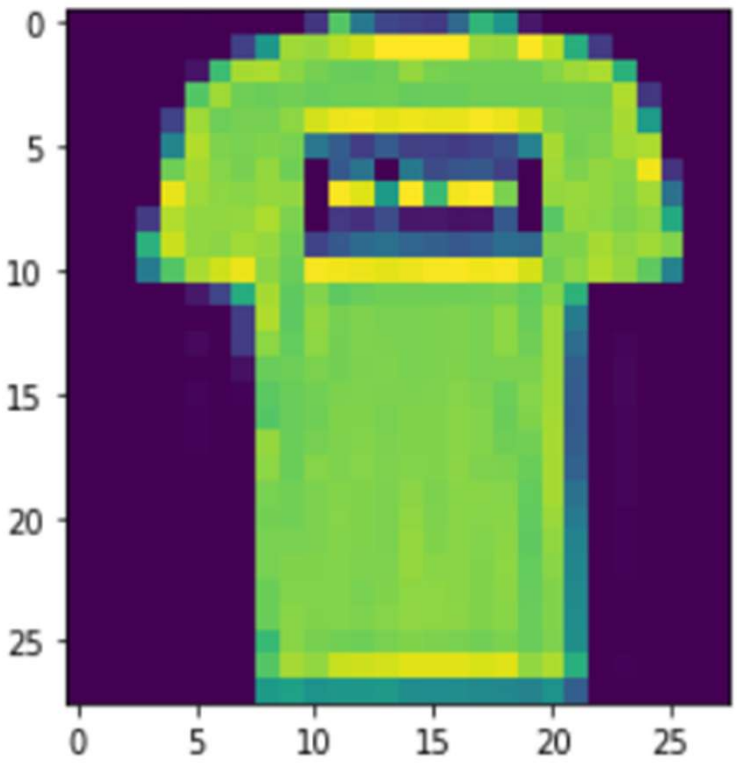
Numpy타입의 숫자배열들



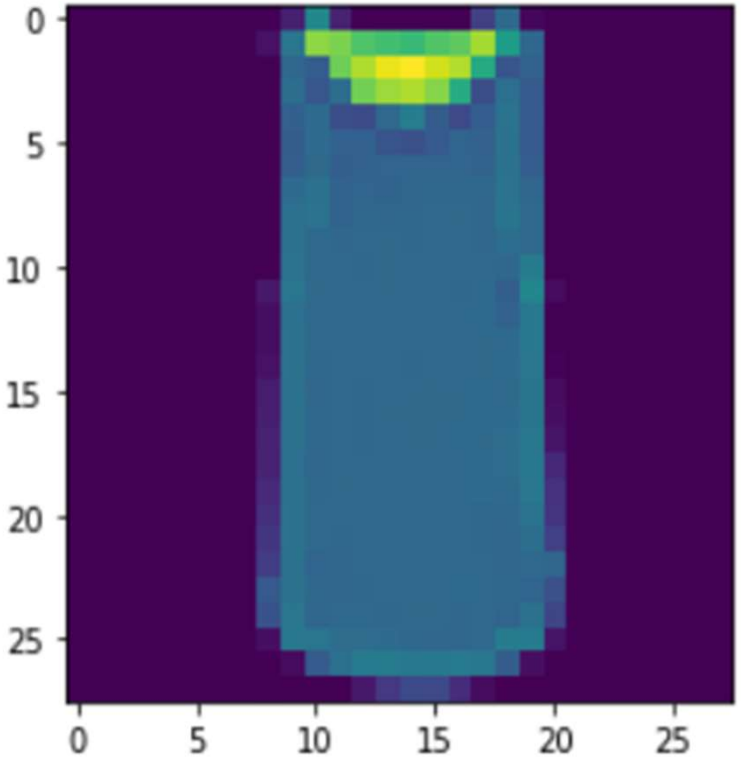


숫자배열들의 값을 기반으로 그림으로 가시화

다른 이미지들도 가시화 해보기



1번

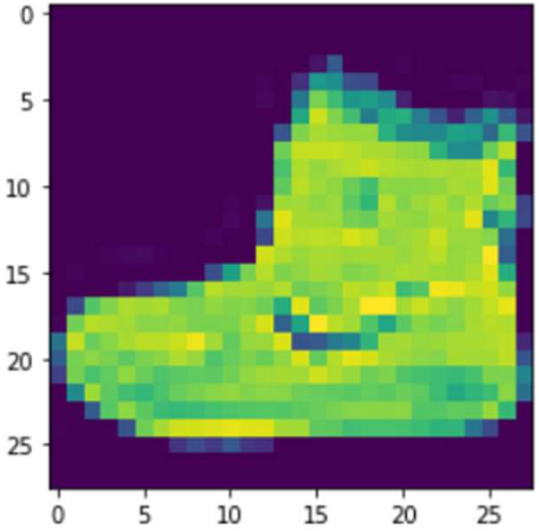


2번

태깅 (정답) 정보 확인해보기

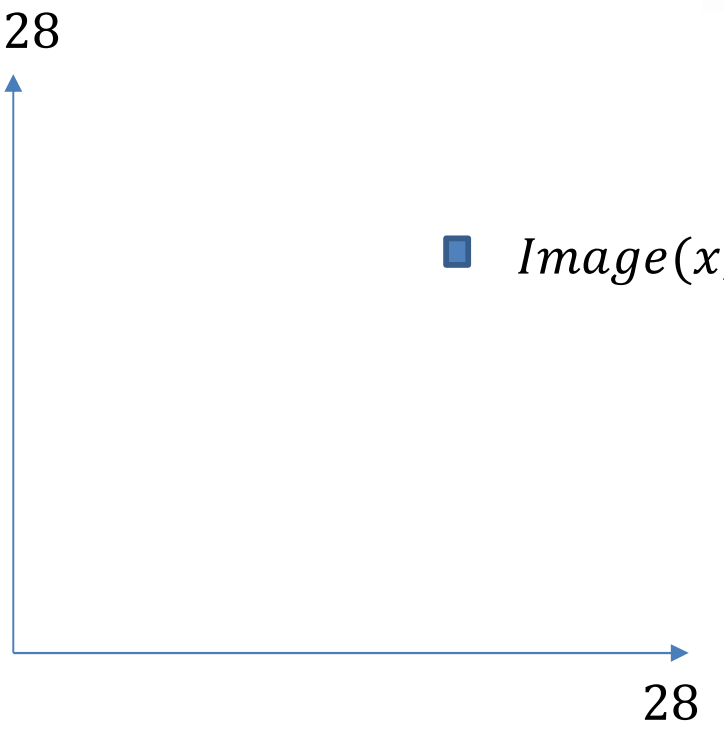

```
import tensorflow as tf
import numpy as np
import matplotlib.image as img
import matplotlib.pyplot as pp

fashion_mnist = tf.keras.datasets.fashion_mnist
(x_train, y_train),(x_test, y_test) = fashion_mnist.load_data()
pp.imshow(x_train[0])
print(y_train[0])
```

Ankle boots -> 9번 클래스



Label	Description
0	T-Shirt/Top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandals
6	Shirt
7	Sneaker
8	Bag
9	Ankle boots

이미지의 수학적으로 바라보기

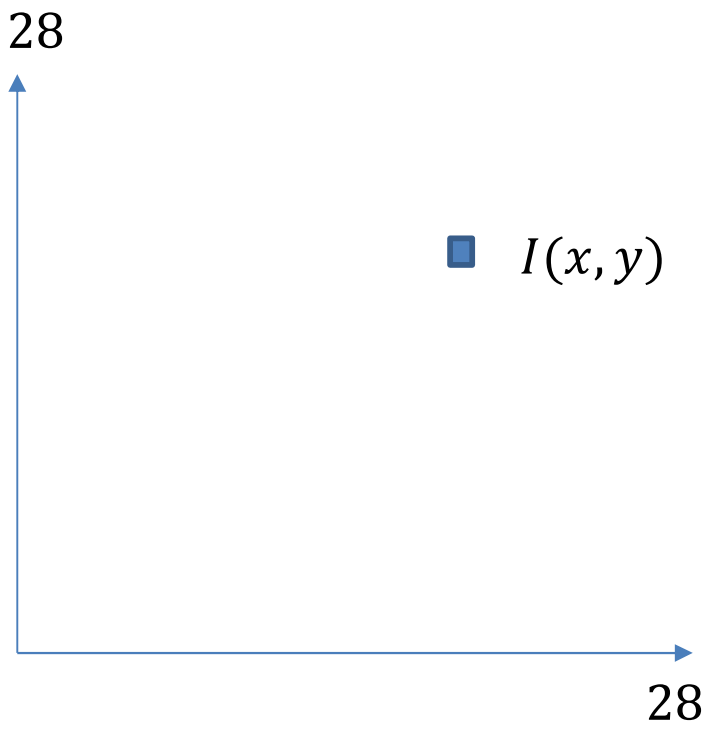


현재 이미지 각 픽셀의 scale

$$0 \leq Image(x, y) \leq 255$$

영상을 함수로 취급해서 생각해보기

이미지의 정규화



영상을 함수로 취급해서 생각해보기

현재 이미지 각 픽셀의 스케일

$$0 \le I(x,y) \le 255$$

$$\frac{0}{255} \le \frac{I(x,y)}{255} \le \frac{255}{255}$$

$$0 \le \frac{I(x,y)}{255} \le 1$$

$$0 \le \hat{I}(x,y) \le 1$$

정규화된 각 픽셀의 스케일

Numpy Broadcasting을 통한 간단한 정규화

현재 이미지 각 픽셀의 스케일

$$0 \leq I(x,y) \leq 255$$

$$\frac{0}{255} \leq \frac{I(x,y)}{255} \leq \frac{255}{255}$$

$$0 \leq \frac{I(x,y)}{255} \leq 1$$

$$0 \leq \hat{I}(x,y) \leq 1$$

정규화된 각 픽셀의 스케일

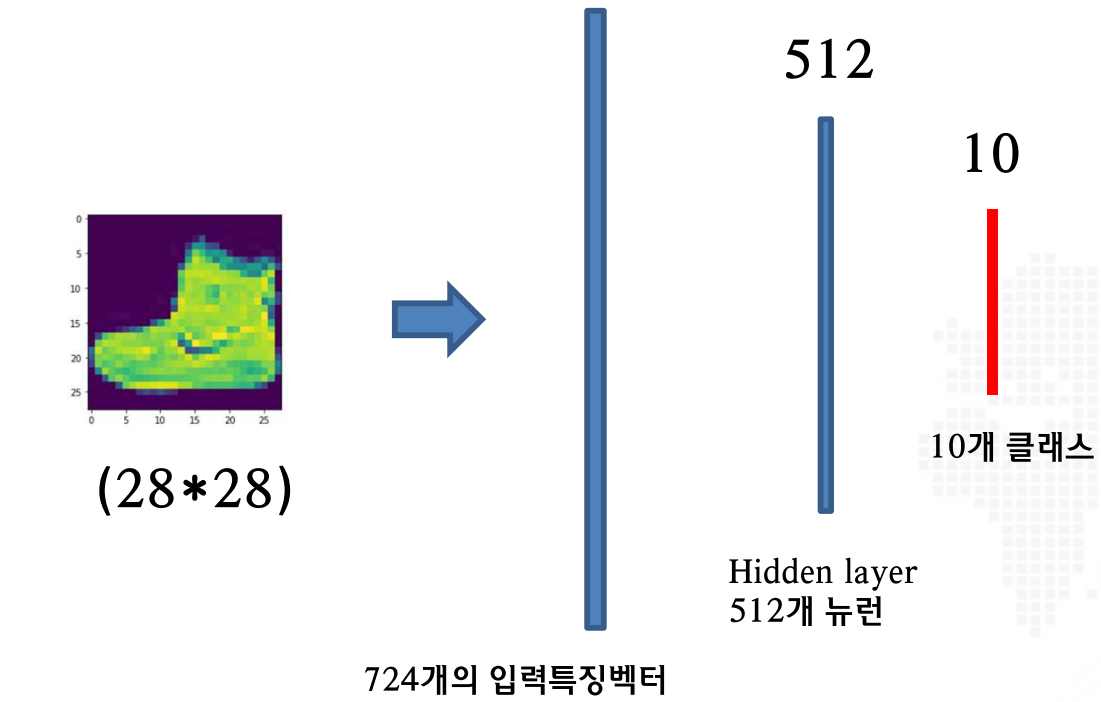
```
x_train = x_train/255.0
x_test = x_test/255.0
```

이렇게만 해주면 0에서 1사이의 실수 영역으로 정규화

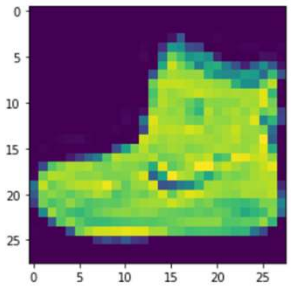
Keras Sequential API 기반 모델 사용


```
model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(512, activation=tf.nn.relu),
    tf.keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

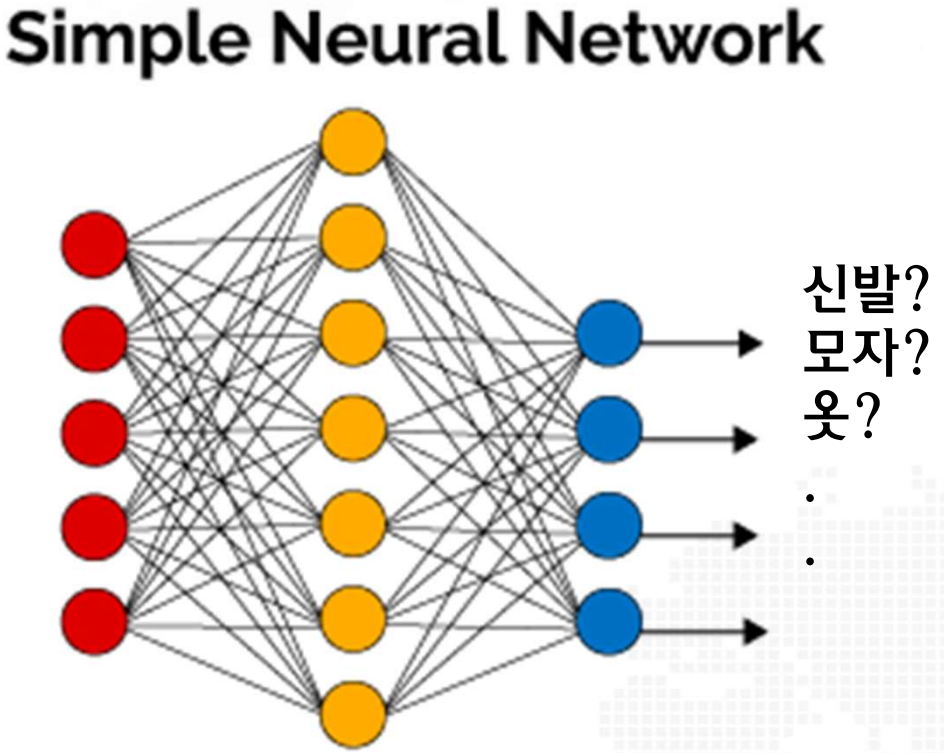
네트워크 Layer 정의



Simple Neural Network

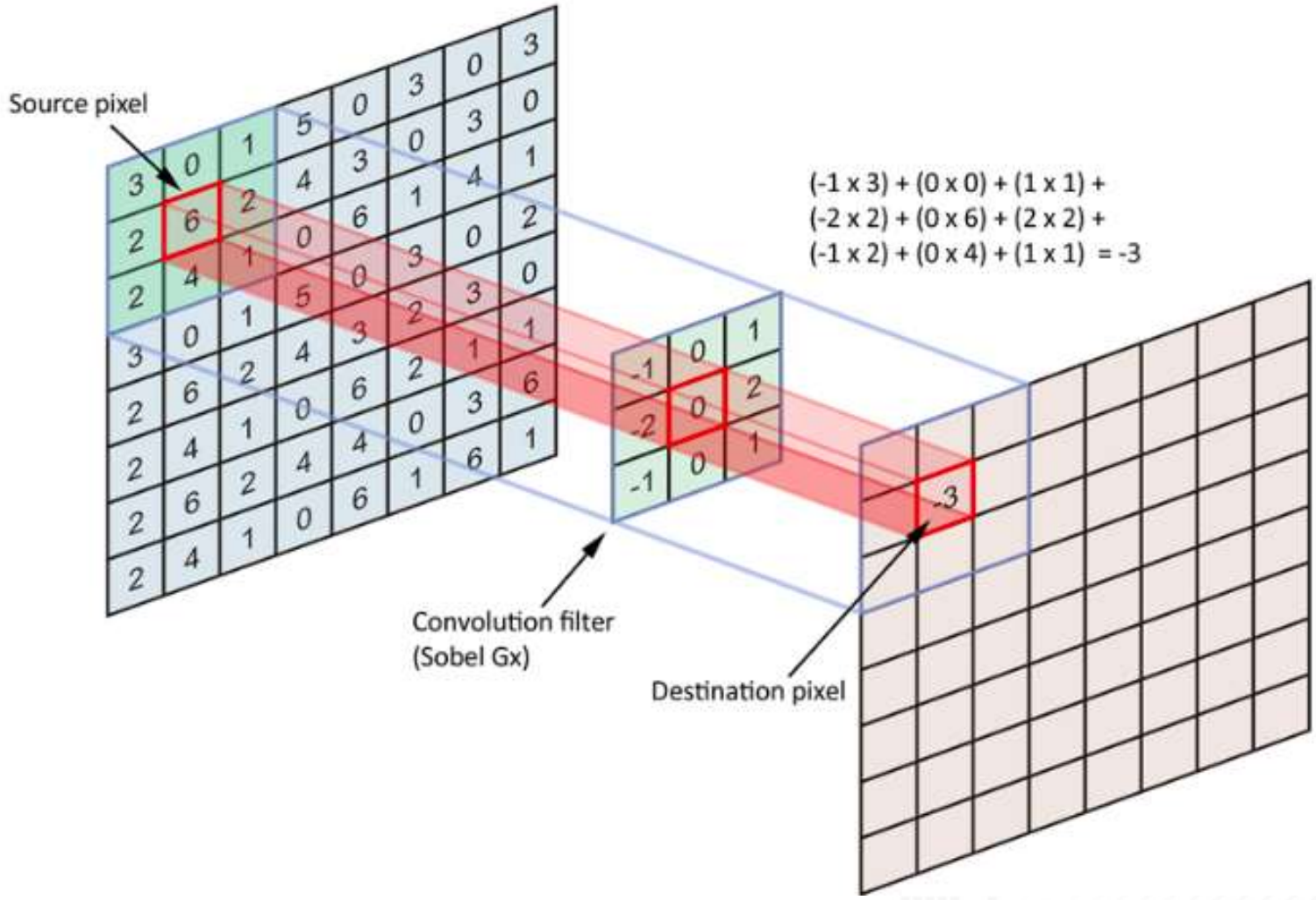


(28*28)



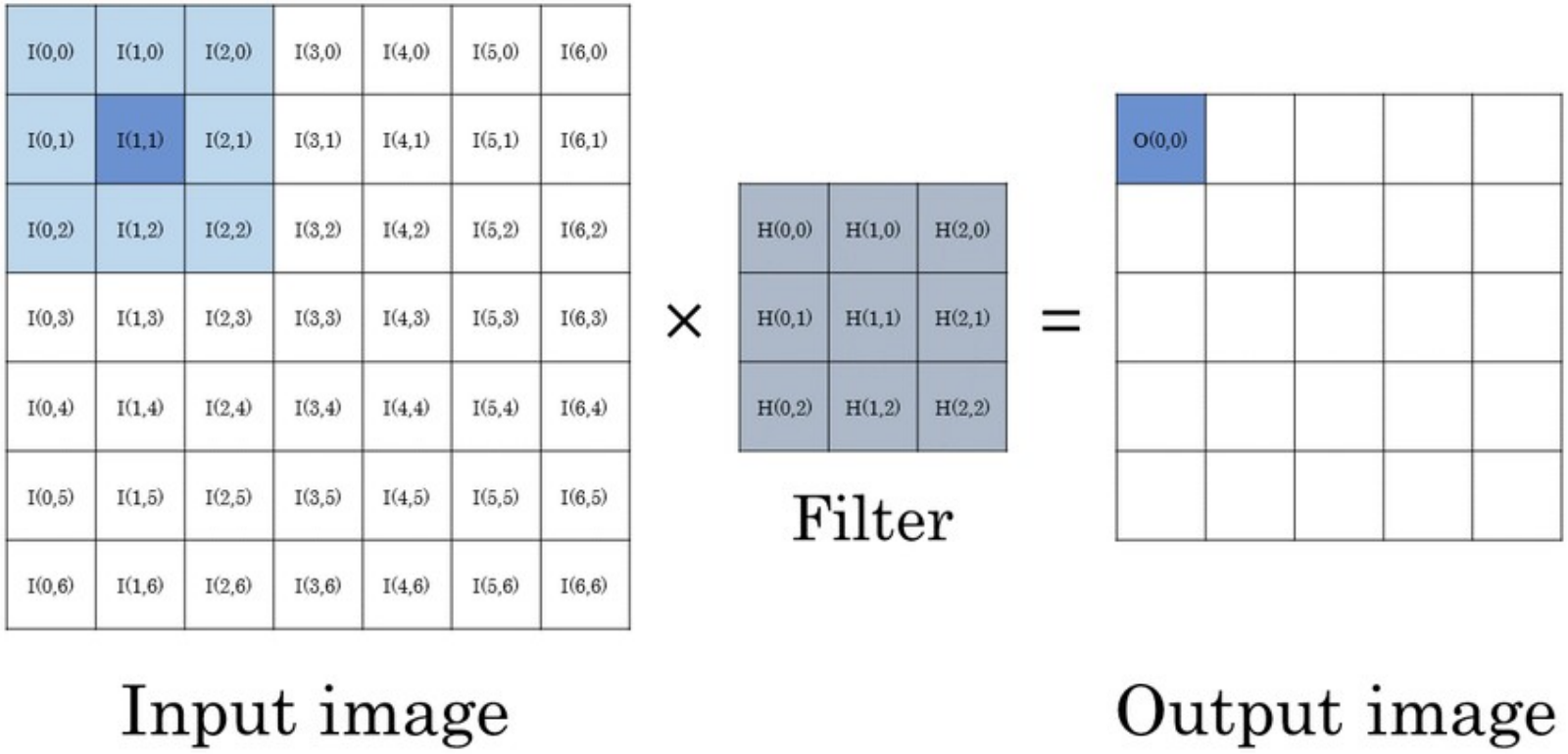
Convolutional Neural Network

Convolution and Image Filter



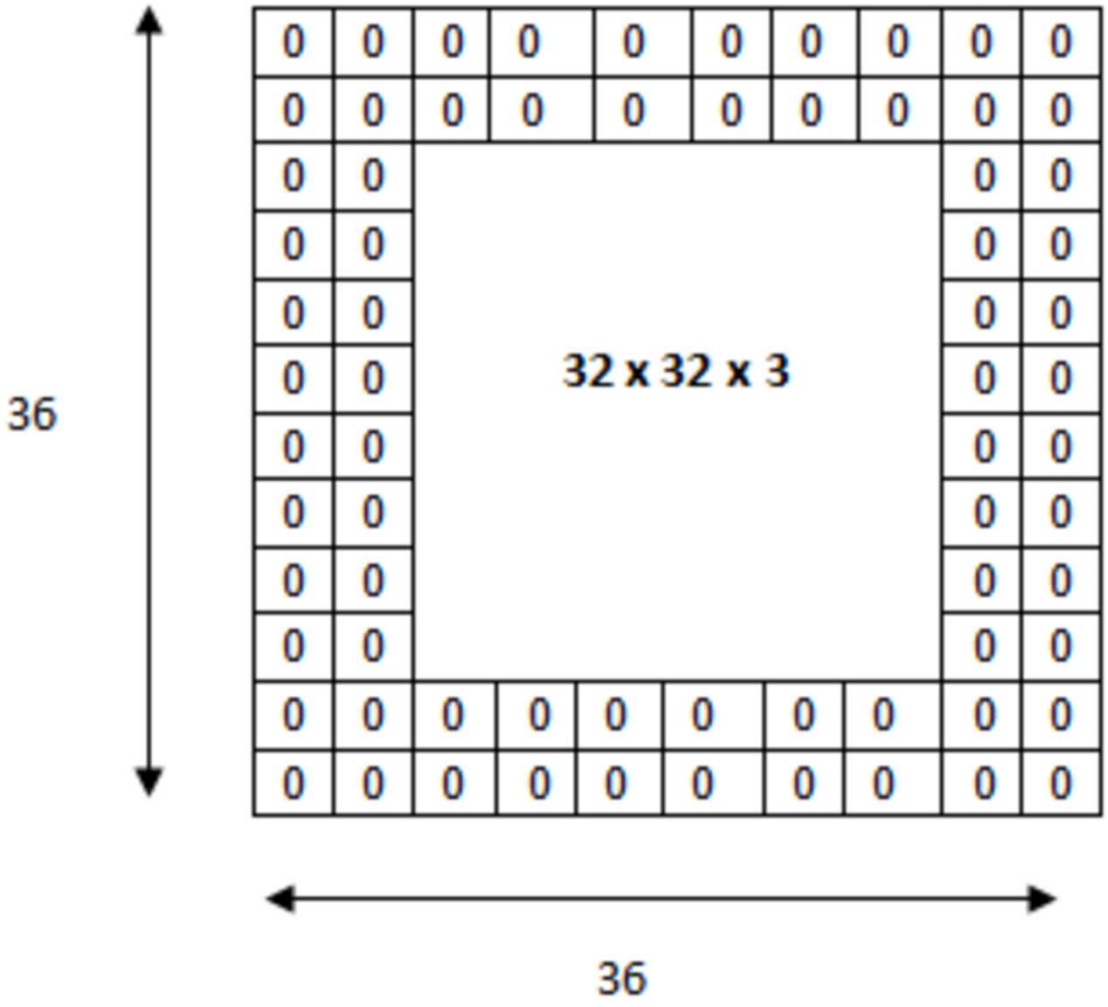
3x3 convolution **필터를 이용한** convolution **연산 과정**

Convolution and Image Filter



Convolution 연산 후 이미지가 축소되는 이유

패딩 (padding)

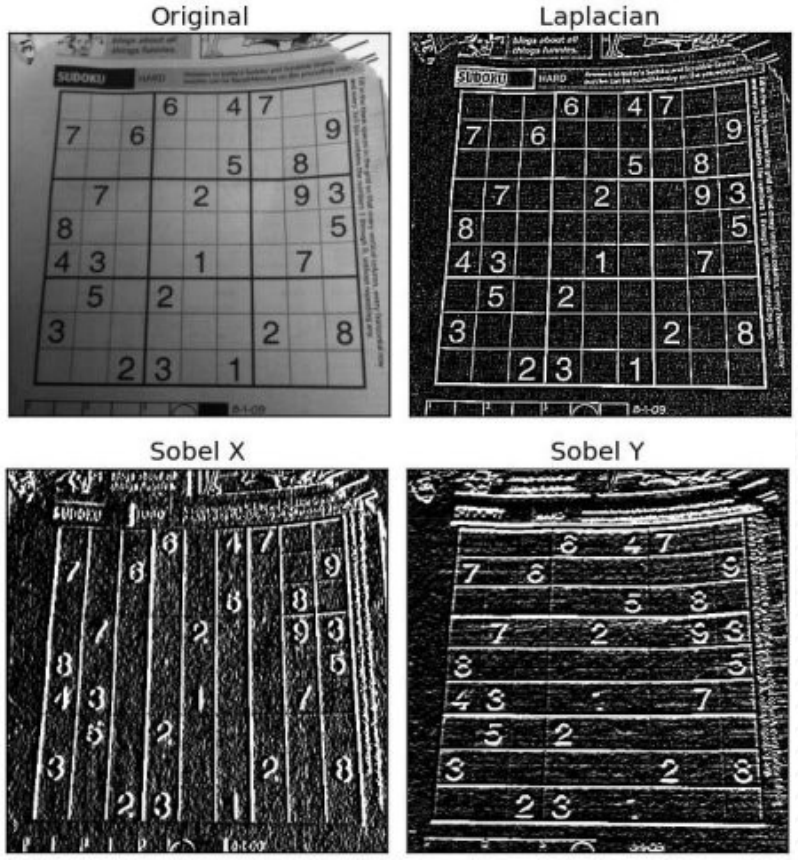


이미지 축소 방지

Convolution연산 으로 할 수 있는 것들

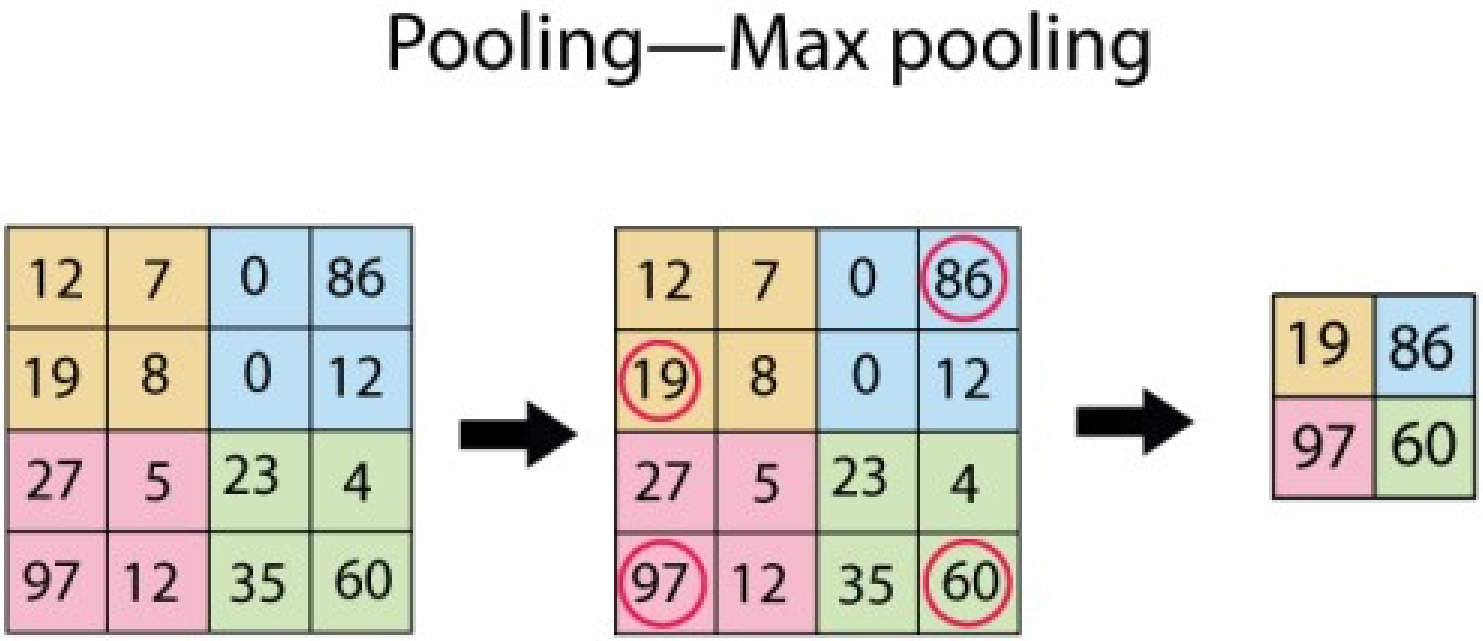
- Image Smoothing (부드럽게 하기)
- Image Sharpening (강조하기)
- Partial Derivatives with noisy images (x, y 편미분)
- Image Edge Detection (윤곽선 검출)
- .
- .
- 등등
- 이미지로부터 특징을 추출할 수 있다는 것

인공지능에서 Convolution과정을 통하여 다양한 특징 추출 과정을 거칠 수 있다.



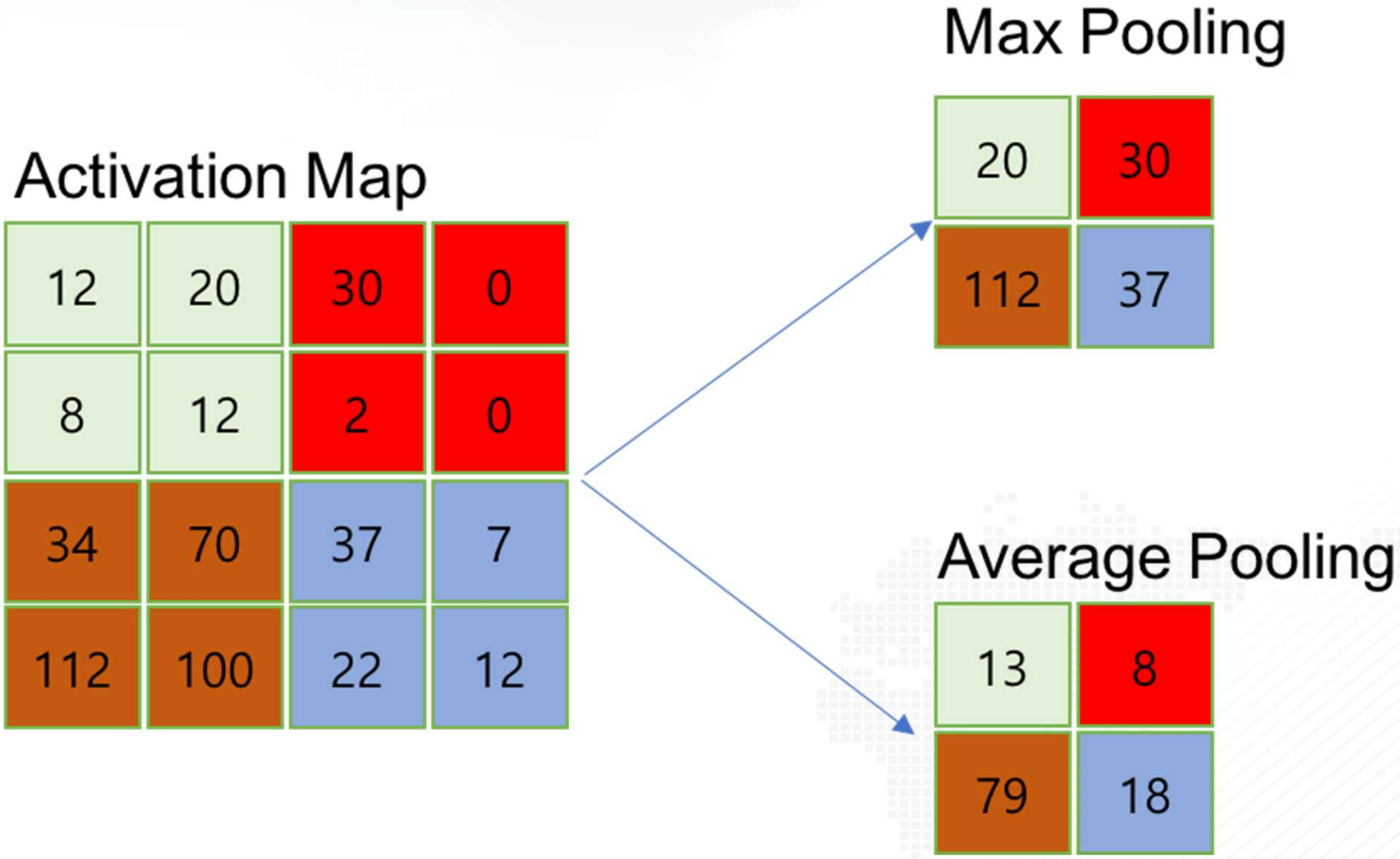
윤곽선 검출 예시

Max Pooling 이란?

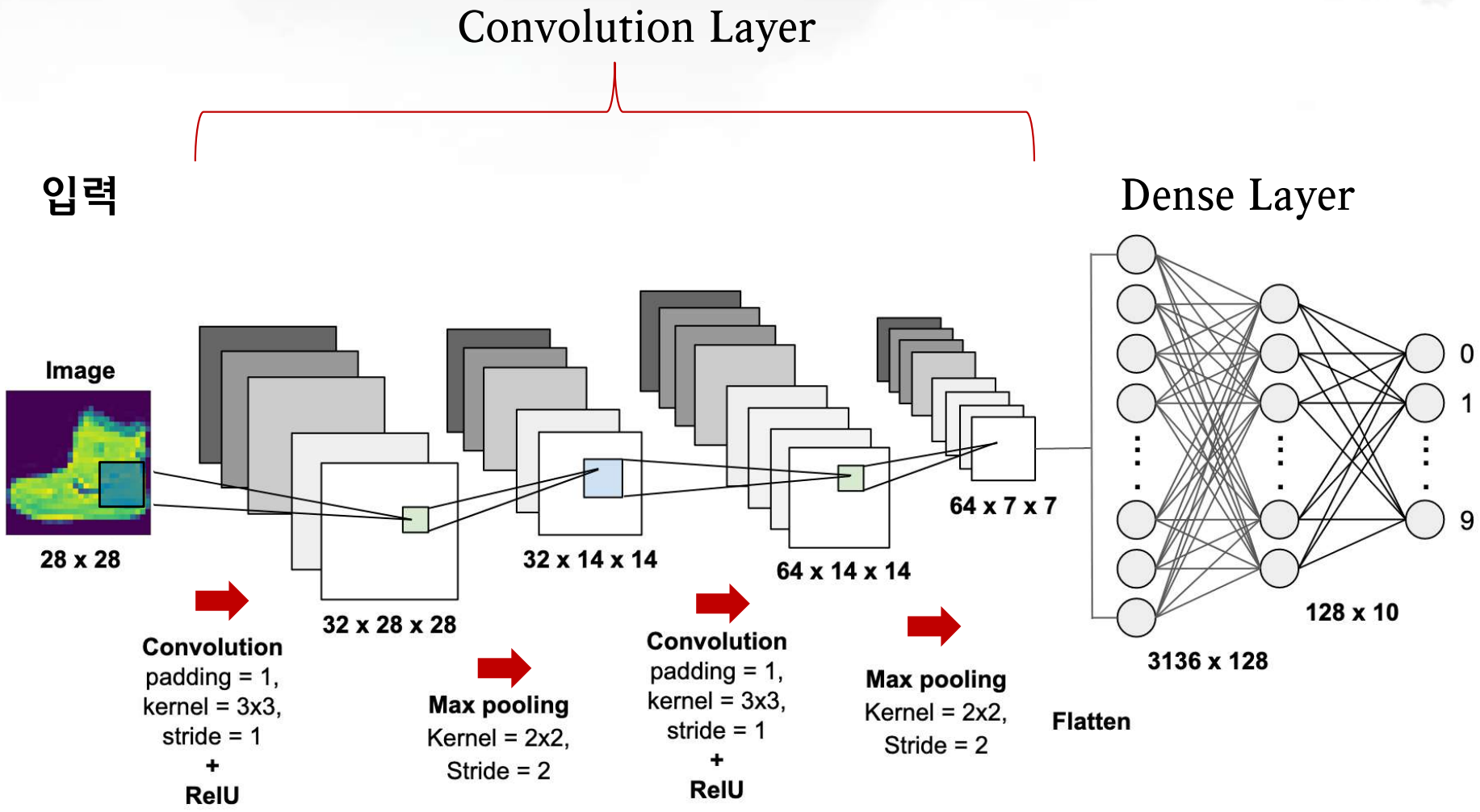


가장 큰 값, 즉 가장 도드라지는 특징만을 가져오는 것
정보는 최대한 보존하면서 압축된 큰 특징정보만 추출하고자 하는 목적

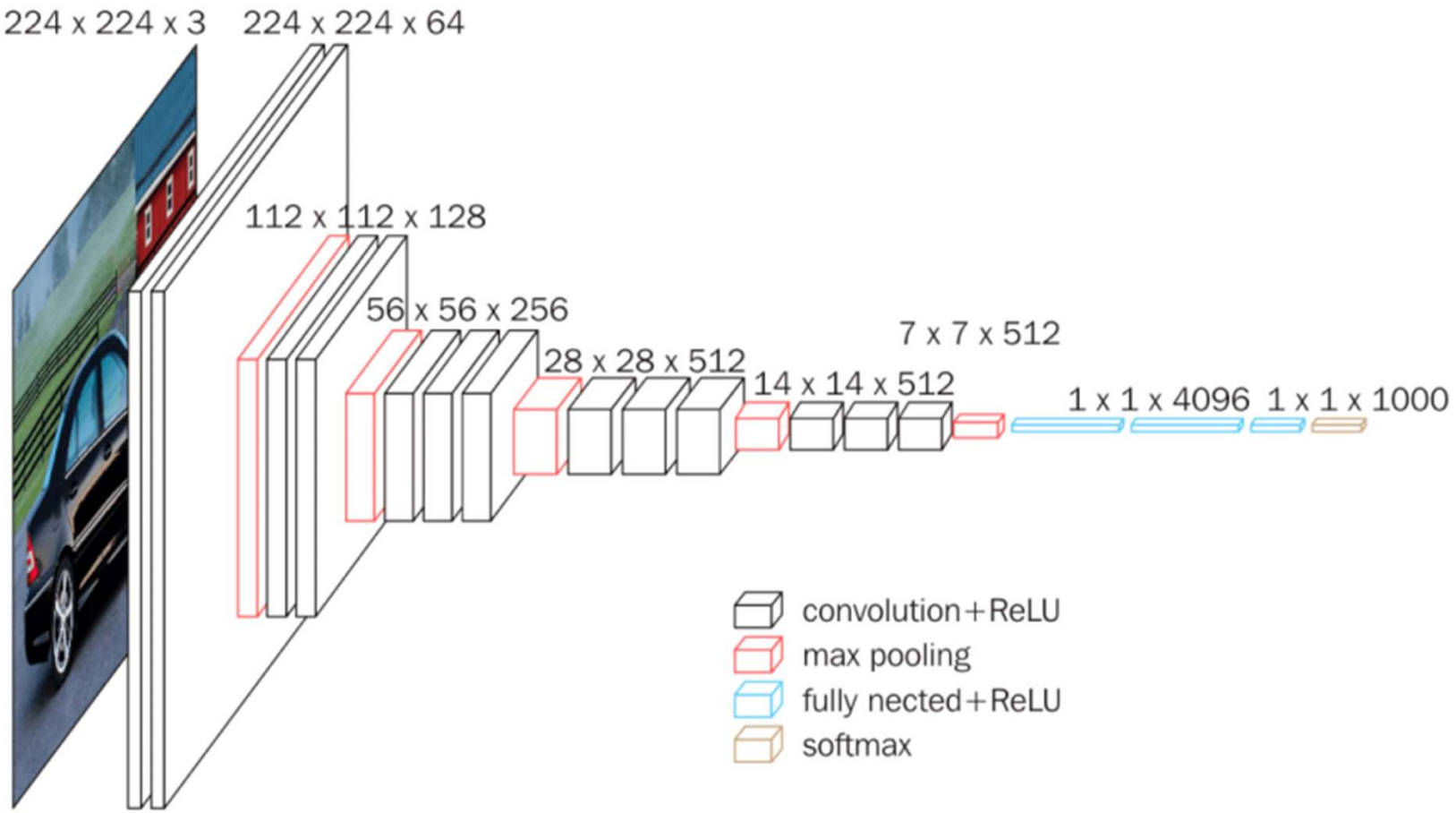
다양한 Pooling

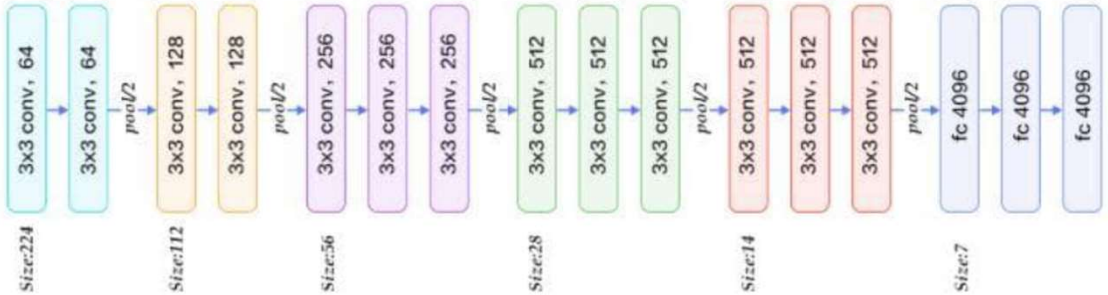

Activation Map
12
20
30
0
8
12
2
0
34
70
37
7
112
100
22
12
Max Pooling
20
30
112
37
Average Pooling
13
8
79
18

Convolutional Neural Network (CNN)



VGG16


224 x 224 x 3
224 x 224 x 64
112 x 112 x 128
56 x 56 x 256
28 x 28 x 512
14 x 14 x 512
7 x 7 x 512
1 x 1 x 4096
1 x 1 x 1000
convolution+ReLU
max pooling
fully nected+ReLU
softmax


3x3 conv, 64
3x3 conv, 64
pool/2
3x3 conv, 128
3x3 conv, 128
pool/2
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
pool/2
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool/2
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool/2
fc 4096
fc 4096
fc 4096
Size:224
Size:112
Size:56
Size:28
Size:14
Size:7