

Mnist

손글씨 인식

2021.08.17. 장혜선

CONTENTS

Mnist 손글씨 인식

Chapter 01

소개

Chapter 02

신경망 구현

Chapter 03

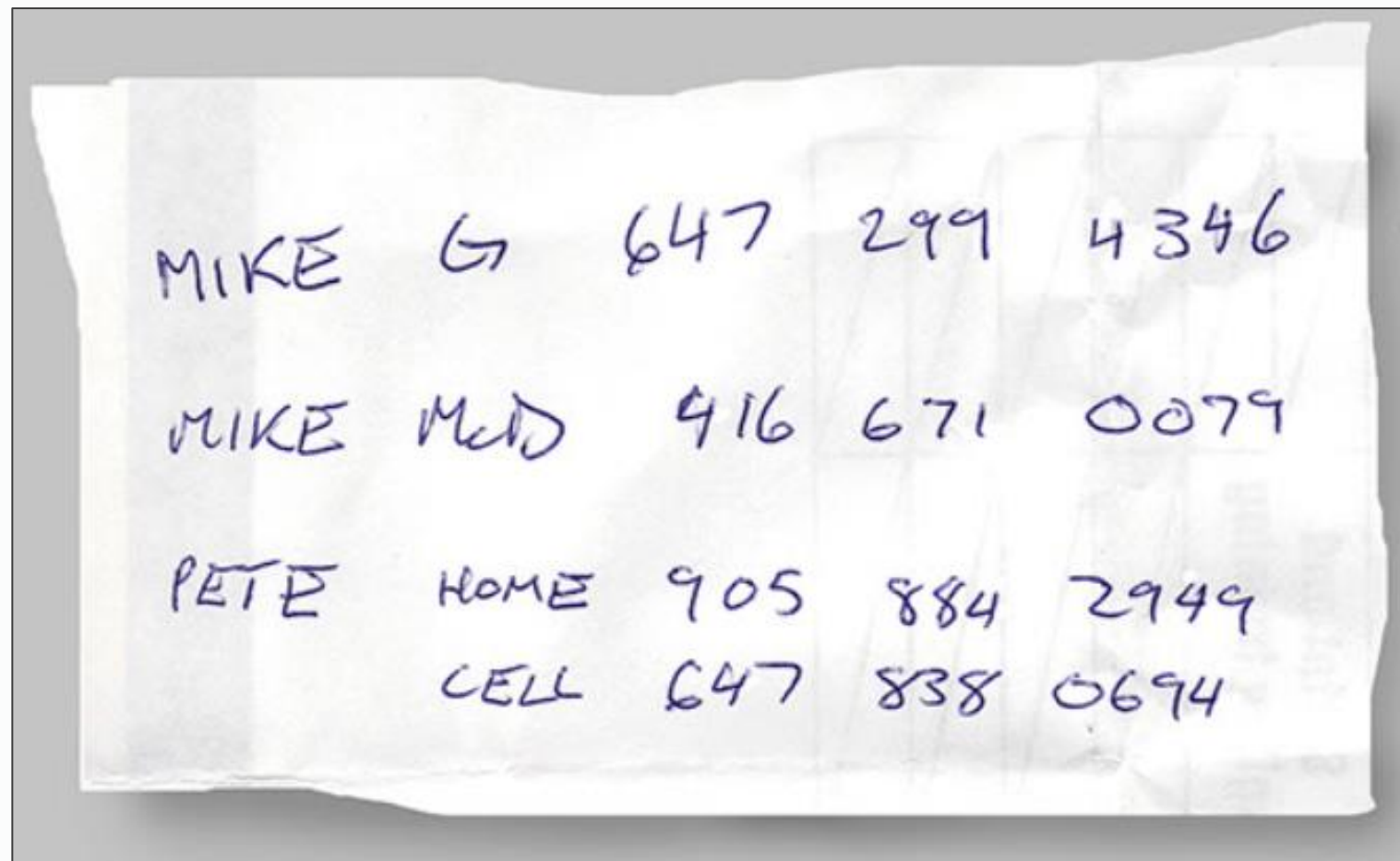
컨볼루션 신경망

Chapter 01

소개

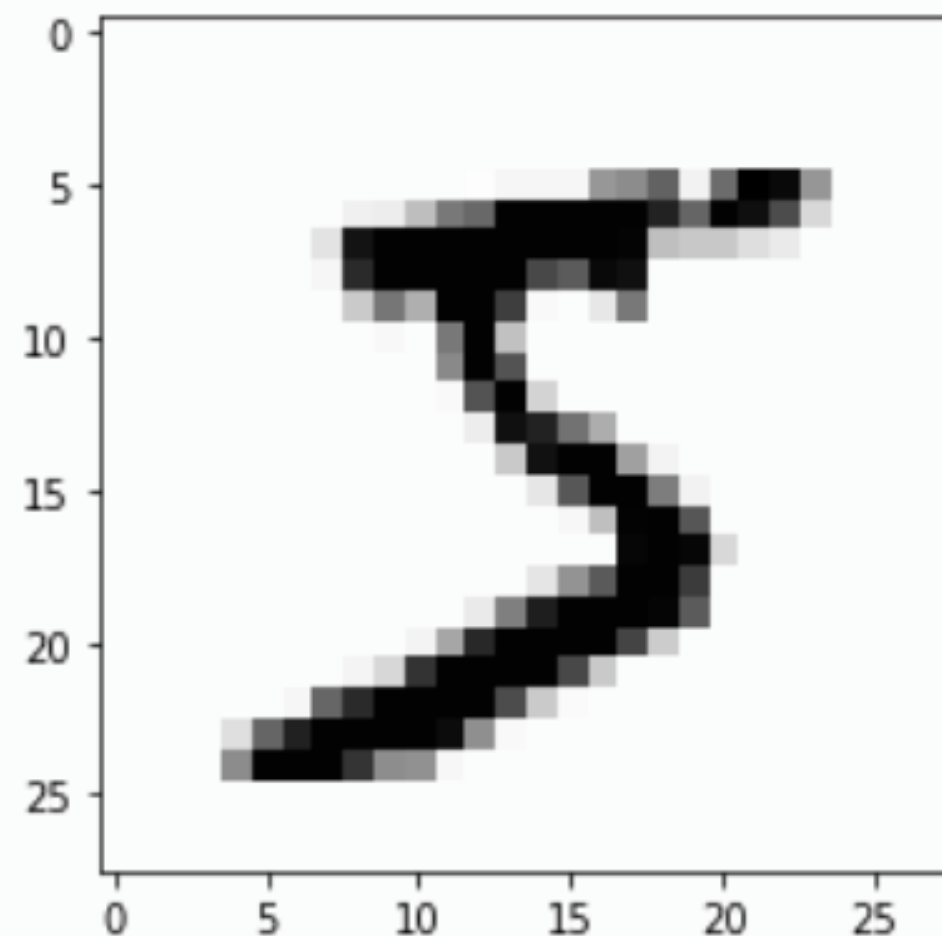
소개

손글씨를 인식할 수 있을까?



MNIST Dataset





- 이미지 출력

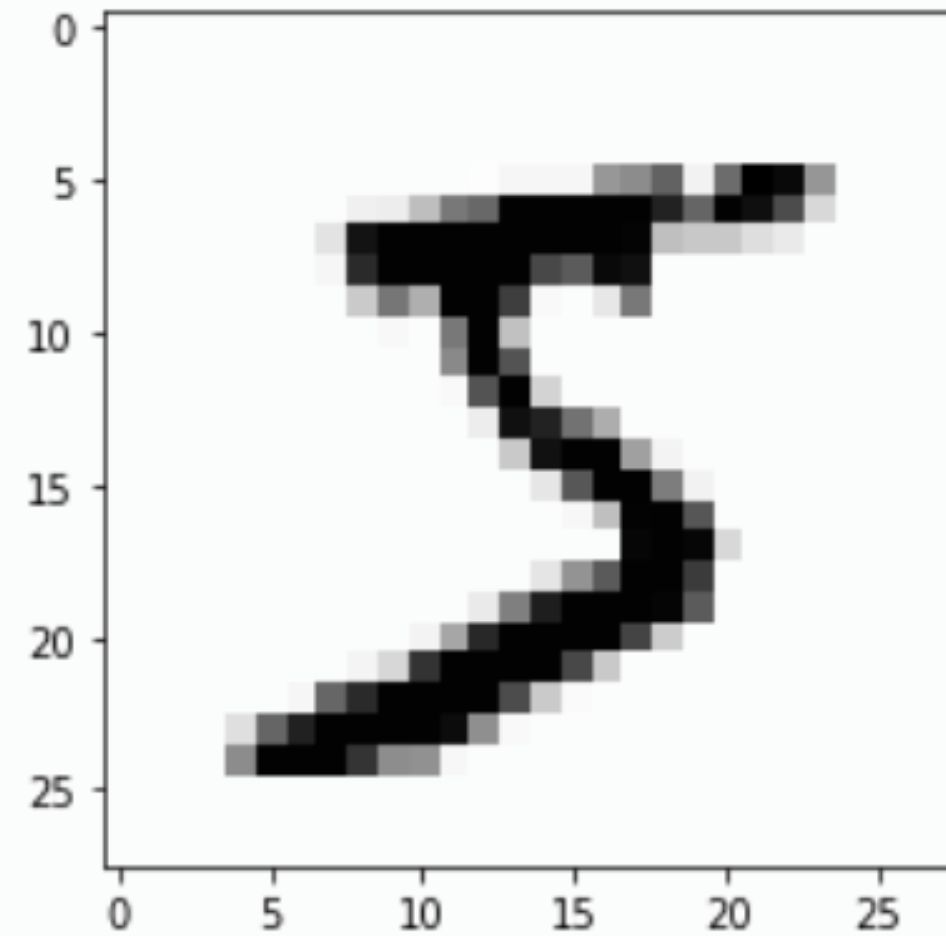
```
import matplotlib.pyplot as plt
plt.imshow(X_train[0], cmap='Greys')
plt.show()
```

- 밝기 정도에 따라 등급매기기

```
for x in X_train[0]:
    for i in x:
        sys.stdout.write('%d\t' % i)
    sys.stdout.write('\n')
```

소개

어떻게 인식할까?



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	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	3	18	18	18	126	136	175	26	166	255	247	127	0	0	0	0	0
0	0	0	0	0	0	0	0	0	30	36	94	154	170	253	253	253	253	253	225	172	253	242	195	64	0	0	0	0	
0	0	0	0	0	0	0	0	49	238	253	253	253	253	253	253	253	253	251	93	82	82	56	39	0	0	0	0	0	
0	0	0	0	0	0	0	0	18	219	253	253	253	253	253	198	182	247	241	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	80	156	107	253	253	205	11	0	43	154	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	14	1	154	253	90	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	139	253	190	2	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	11	190	253	70	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	35	241	225	160	108	1	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	81	240	253	253	119	25	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	45	186	253	253	150	27	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	93	252	253	187	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	249	253	249	64	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	46	130	183	253	253	207	2	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	39	148	229	253	253	253	250	182	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	24	114	221	253	253	253	253	201	78	0	0	0</							

Chapter 02

신경망 구현

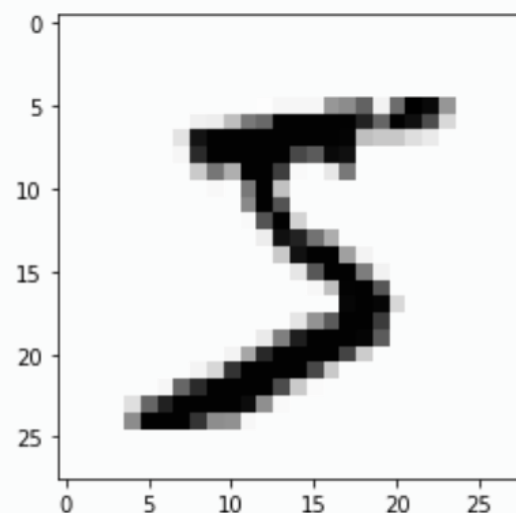
```
(X_train, Y_class_train), (X_test, Y_class_test) = mnist.load_data()
```

■ Trainset

■ Testset

60,000

10,000



```
X_train = X_train.reshape(X_train.shape[0], 784).astype('float32') / 255
X_test = X_test.reshape(X_test.shape[0], 784).astype('float64') / 255
```

2차원 -> 1차원

0~255 -> 0~1

- reshape(총 샘플 수, 1차원 속성의 수)

신경망

데이터 처리(원-핫 인코딩)

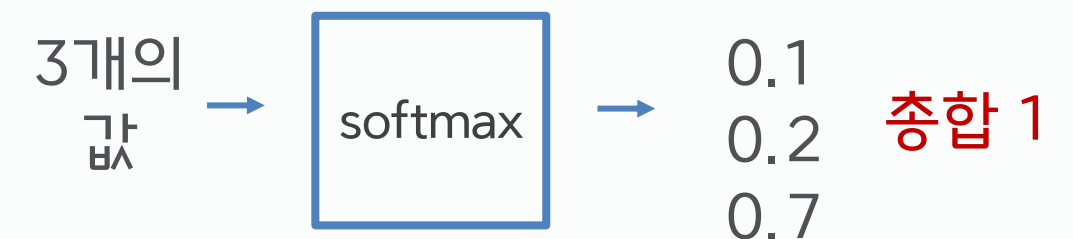
```
Y_train = np_utils.to_categorical(Y_class_train, 10)  
Y_test = np_utils.to_categorical(Y_class_test, 10)
```

[0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]

신경망

신경망 구조

```
model = Sequential()  
model.add(Dense(512, input_dim=784, activation='relu'))  
model.add(Dense(10, activation='softmax'))  
model.compile(loss='categorical_crossentropy',  
              optimizer='adam',  
              metrics=['accuracy'])
```



- 오차가 좋아질 때마다 모델 업데이트

```
modelpath="./models/{epoch:02d}-{val_loss:.4f}.hdf5"  
checkpointer = ModelCheckpoint(filepath=modelpath, monitor='val_loss',  
                               verbose=1, save_best_only=True)
```

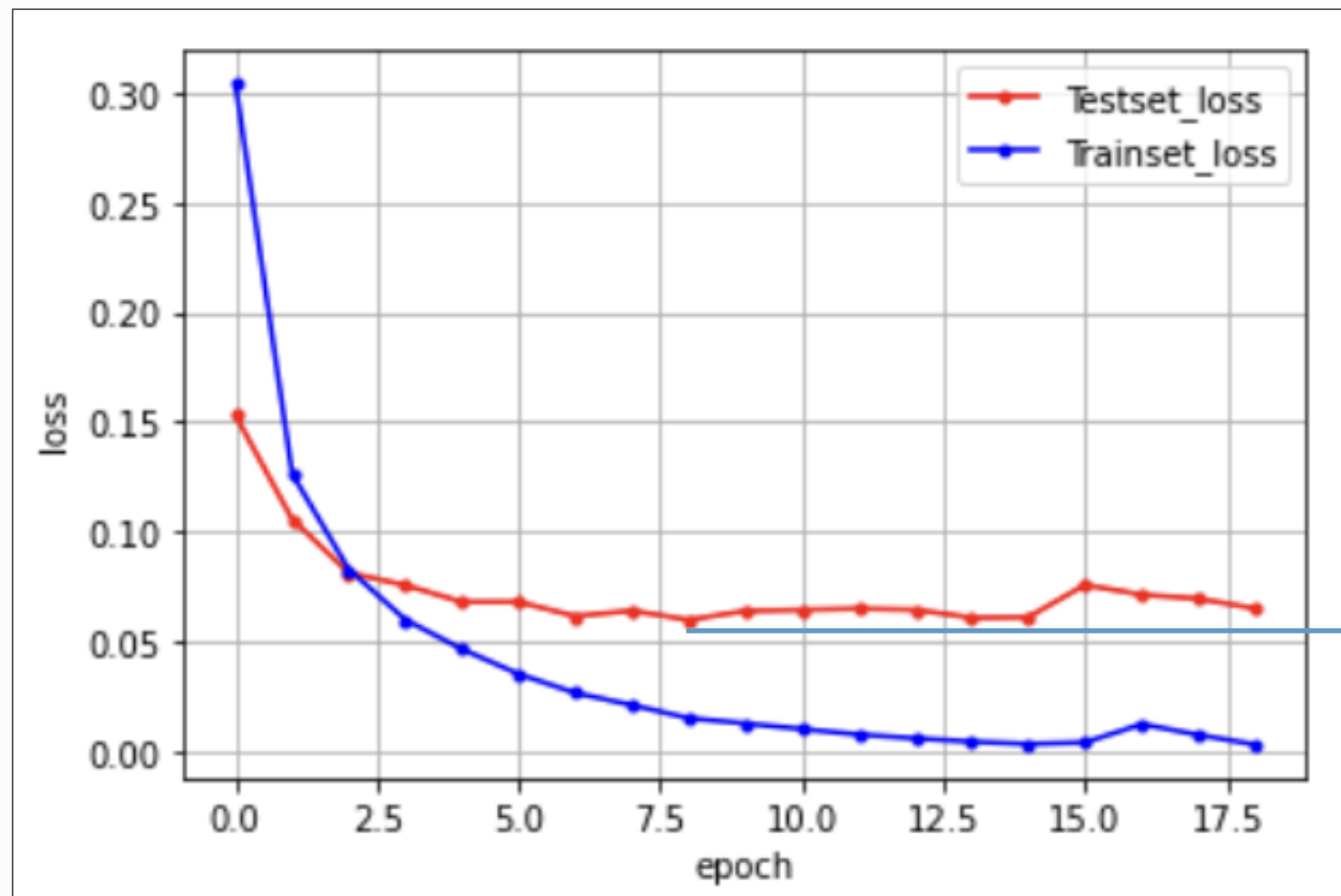
- 10번 안에 오차가 좋아지지 않을 시 중단

```
early_stopping_callback = EarlyStopping(monitor='val_loss', patience=10)
```

- 학습 세부 설정

```
history = model.fit(X_train, Y_train, validation_data=(X_test, Y_test),  
                   epochs=30, batch_size=200, verbose=0,  
                   callbacks=[early_stopping_callback, checkpointer])
```

- 테스트 정확도 : 0.9831



중단

Chapter 03

컨볼루션 신경망

- 이미지 인식 분야에서 좋은 성능을 보임
입력된 이미지에서 다시 한 번 특징 추출을 위해 마스크 도입

1	0	1	0
0	1	1	0
0	0	1	1
0	0	1	0

마스크

X1	X0
X0	X1



$$(1 \times 1) + (0 \times 0) + (0 \times 0) + (1 \times 1) = 2$$

1X1	0X0	1	0
0X0	1X1	1	0
0	0	1	1
0	0	1	0

- 이미지 인식 분야에서 좋은 성능을 보임
입력된 이미지에서 다시 한 번 특징 추출을 위해 마스크 도입

1	0	1	0
0	1	1	0
0	0	1	1
0	0	1	0



X1	X0
X0	X1



2	1	1
0	2	2
0	1	1

- 결과가 크고 복잡하면 다시 한 번 축소를 거침
서브 샘플링이라고도 불림

< 맥스 풀링 >

1	0	1	0
0	4	2	0
0	1	6	1
0	0	1	0



4	2
1	6

- 컨볼루션 -> 풀링 -> 드롭아웃 -> Flatten

```
model = Sequential()  
model.add(Conv2D(32, kernel_size=(3, 3), input_shape=(28, 28, 1), activation='relu'))  
model.add(Conv2D(64, (3, 3), activation='relu'))
```

Conv2D

1. 마스크 적용 개수 (32,
2. 마스크 크기 kernel_size=(3, 3),
3. 입력값(행, 열, 색상 또는 흑백) input_shape=(28, 28, 1)
4. 활성화 함수 activation='relu'))

CNN

구현(맥스 풀링)

- 컨볼루션 -> 풀링 -> 드롭아웃 -> Flatten

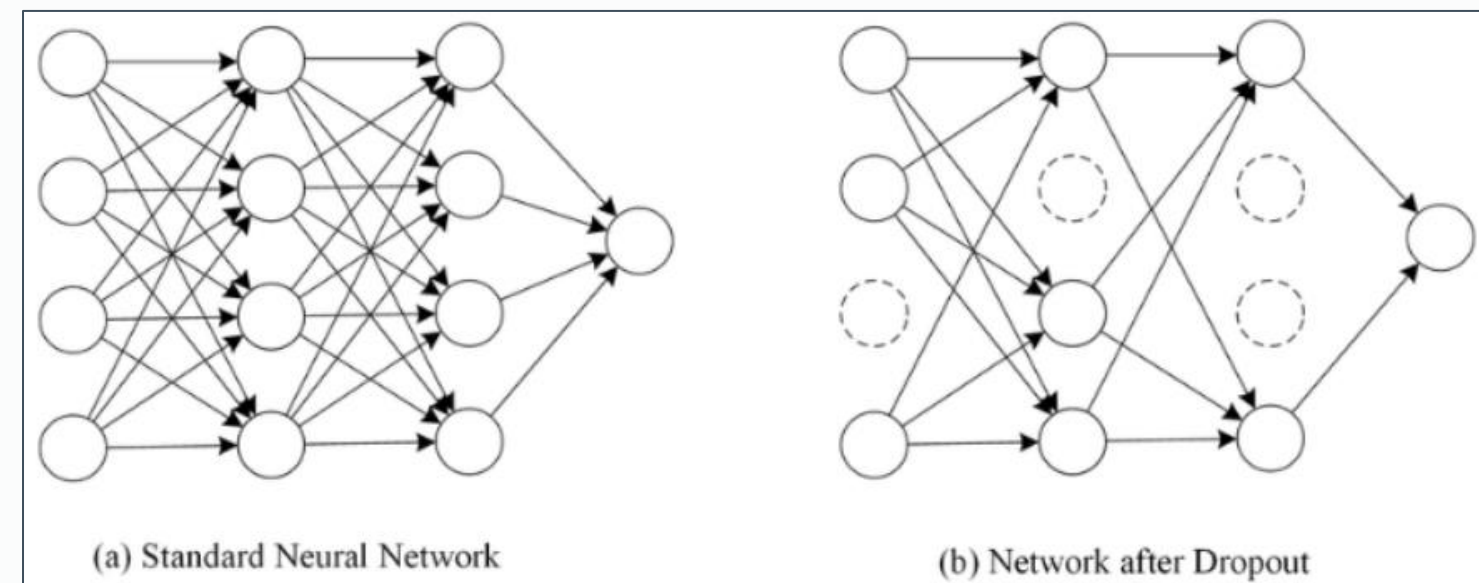
```
model.add(MaxPooling2D(pool_size=2))
```

CNN

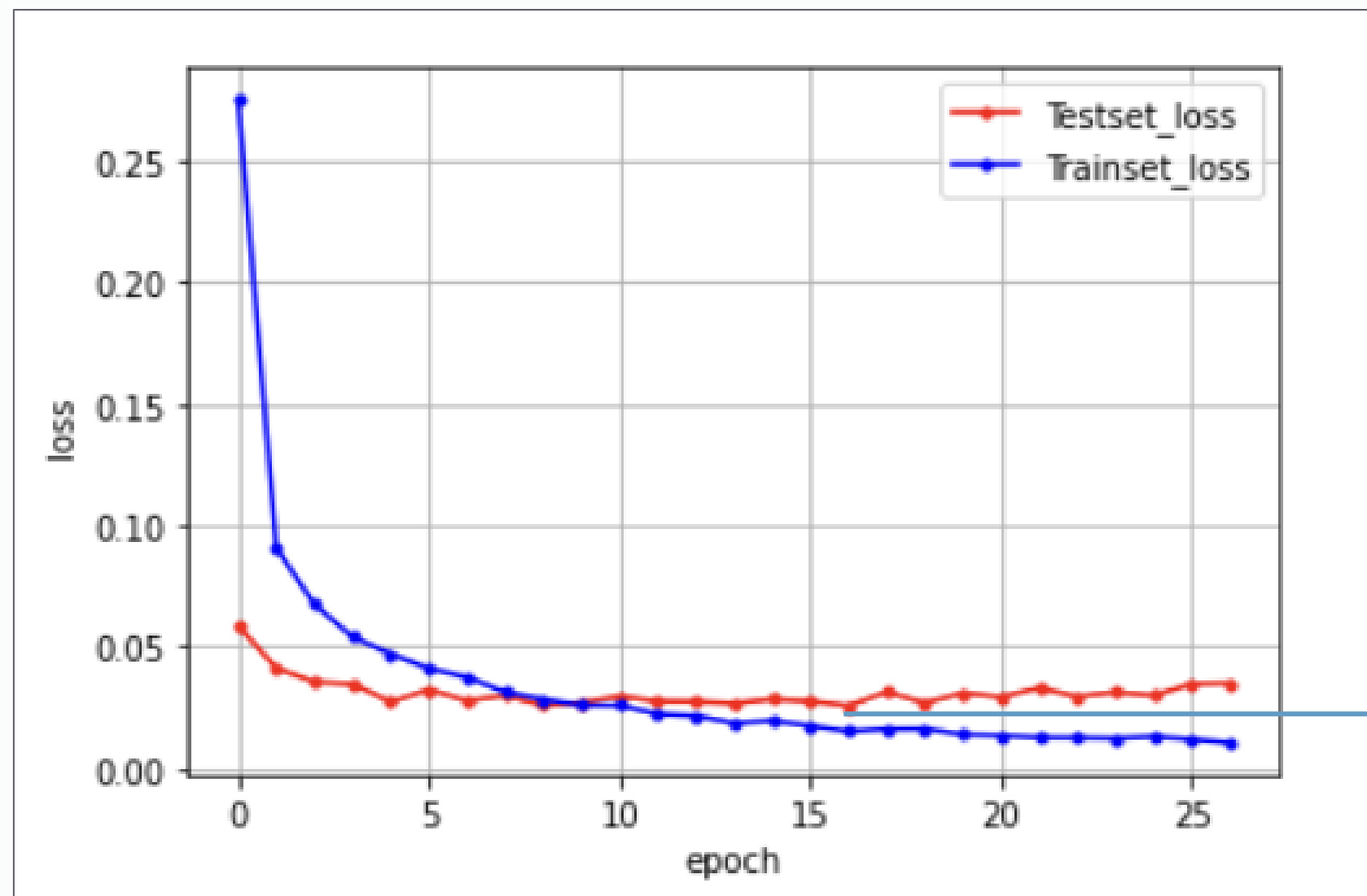
드롭아웃 & Flatten

- 컨볼루션 -> 풀링 -> 드롭아웃 -> Flatten

```
model.add(Dropout(0.25))  
model.add(Flatten())
```



- Test Accuracy: 0.9921



중단

감사합니다.