

학습률 범위 테스트

학습률 범위 테스트

학습률이 지나치게 작으면 → 학습이 느림

지나치게 크면 → 학습이 불안정

학습률을 아주 작은 값부터 서서히 올려가면서 테스트

배치 수 계산

배치 수 계산

```
batch_size = 32  
num_batch = x_train.shape[0] / batch_size
```

테스트에 사용할 에포크 수

```
epochs = 1
```

학습률 범위 설정

학습률을 `1e-5` 즉 0.00001부터 시작해서 100까지 상승

```
min_lr = 1e-5  
max_lr = 100
```

학습률 상승률 계산

그러려면 배치마다 학습률을 몇 배씩 증가시켜야 하는지 계산한다.

```
rate = (max_lr / min_lr) ** (1/(num_batch*epochs))  
rate
```

학습률 스케줄 설정

ExponentialDecay 는 매 배치마다 일정 비율로 학습률을 변경

```
schedule = tf.keras.optimizers.schedules.ExponentialDecay(  
    initial_learning_rate=min_lr,  
    decay_steps=1,  
    decay_rate=rate)
```

설정

```
opt = tf.keras.optimizers.Adam(learning_rate=schedule)
model.compile(optimizer=opt,
               loss='sparse_categorical_crossentropy',
               metrics=['accuracy'])
```

손실 기록 함수

```
losses = []  
def record_loss(batch, logs):  
    losses.append(logs['loss'])
```


학습 진행

`LambdaCallback` 을 이용해서 매 배치의 끝마다 `record_loss` 함수를 실행

```
model.fit(  
    x_train, y_train, epochs=epochs,  
    callbacks=[tf.keras.callbacks.LambdaCallback(on_batch_end=record_loss)])
```

손실 변화 그래프

```
import matplotlib.pyplot as plt  
plt.plot(losses)
```

후반부에서 손실이 너무 커져서 전반부의 차이가 잘 보이지 않으면 로그를 적용해서 그림

```
import numpy as np  
plt.plot(np.log(losses))
```

배치의 학습률 구하기

손실이 하락하기 시작한 시점과 다시 증가하기 시작하는 시점의 학습률을 확인

```
schedule(50)
```

Early Stopping과 Checkpoint

Early Stopping

모니터링하고 있는 지표가 향상되지 않으면 학습을 중단

```
model.fit(x_train, y_train, epochs=30, validation_split=0.1,  
         callbacks=[tf.keras.callbacks.EarlyStopping(monitor='val_accuracy')])
```

저장 폴더 만들기

```
import os  
os.mkdir('checkpoints')
```

체크 포인트

모니터링하고 있는 지표가 향상되면 모형의 학습된 상태를 저장

```
model.fit(  
    x_train, y_train, epochs=3, validation_split=0.1,  
    callbacks=[  
        tf.keras.callbacks.ModelCheckpoint(  
            filepath='checkpoints/{epoch:02d}-{val_accuracy:.2f}.hdf5',  
            save_best_only=True,  
            monitor='val_accuracy')])
```

최적 모형 불러오기

가장 나중에 만들어진 파일명을 찾는다

```
import glob  
best_model_path = sorted(glob.glob('checkpoints/*.hdf5'))[-1]
```

불러오기

```
best_model = tf.keras.models.load_model(best_model_path)
```


대상 탐지

Faster R-CNN

```
import tensorflow_hub as hub
module = hub.load(
    "https://tfhub.dev/google/faster_rcnn/openimages_v4/inception_resnet_v2/1")
detector = module.signatures['default']
```

SSD

```
module = hub.load("https://tfhub.dev/google/openimages_v4/ssd/mobilenet_v2/1")  
detector = module.signatures['default']
```

예제 이미지

다운로드

```
!wget -c "https://upload.wikimedia.org/wikipedia/commons/6/60/Naxos_Taverna.jpg"
```

열기

```
import PIL.Image
img = PIL.Image.open('Naxos_Taverna.jpg')
width, height = 640, 480
img = img.resize((width, height))
arr = np.array(img) / 255
arr = np.expand_dims(arr, 0)
x = tf.convert_to_tensor(arr, dtype=tf.float32)
```

물체 탐지

탐지

```
result = detector(x)
```

결과

```
entities = result['detection_class_entities'].numpy() # 레이블  
scores = result['detection_scores'].numpy() # 점수  
boxes = result['detection_boxes'].numpy() # 테두리
```

결과 보기

```
idx = 0 # 대상의 번호 (0 ~)
label = entities[idx].decode()
print(f'label: {label}')
print(f'score: {scores[idx]}')
ymin, xmin, ymax, xmax = boxes[idx]
img.crop((xmin * width, ymin * height, xmax * width, ymax * height))
```