

과제 #3

■ [프로그램 5-11] 수정 : 더 좋은 모델 만들기

- 공정한 비교를 위해 모든 옵티마이저는 `adam`, `batch_size = 256`, `epochs = 20` 사용

프로그램 5-11

옵티마이저의 성능 비교: SGD, Adam, Adagrad, RMSprop

```
01 import numpy as np
02 import tensorflow as tf
03 from tensorflow.keras.datasets import fashion_mnist
04 from tensorflow.keras.models import Sequential
05 from 21    ### WARN: Must have ###
06 from 22    # to make this notebook's output stable across runs
07     23 def reset_random_seeds():
08     # fa 24     os.environ['PYTHONHASHSEED']=str(1)
09     (x_t 25     tf.random.set_seed(1)
10     x_tr 26     np.random.seed(1)
11     x_te 27     random.seed(1)
12         28     os.environ['TF_DETERMINISTIC_OPS'] = '1'
13 x_train=x_train.astype(np.float32)/255.0
14 x_test=x_test.astype(np.float32)/255.0
15 y_train=tf.keras.utils.to_categorical(y_train,10)
16 y_test=tf.keras.utils.to_categorical(y_test,10)
```

<< 동일결과 출력을 위해,
랜덤성 동일화

[주의]

- GPU로 에러날 경우 : CPU로 쓸 것
- Method 추가로 에러날 경우 : 해당 method 사용 금지

과제 #3

학습이 오래 걸리기 때문에, 비교모델은 더 단순
화된 모델로 정함(LMS의 코드를 참조!)



```
# 모델을 설계해주는 함수 (모델을 나타내는 객체 model을 반환)
def run_model(x_train, y_train):
    reset_random_seeds()
    model=Sequential()
    model.add(Dense(128,activation='relu',input_shape=(784,)))
    model.add(Dense(64,activation='relu'))
    model.add(Dense(10,activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer=Adam(),
                  metrics=['accuracy'])
    model.fit(x_train, y_train, batch_size=256, epochs=20)
    return model
```



공정비교 위해
동일하게 해야!

과제 #3

39 # SGD 옵티마이저를 사용하는 모델

40

41

삭제

42

43

44 # Adam 옵티마이저를 사용하는 모델

45 dmlp_adam=build_model()

46 dmlp_adam.compile(loss='categorical_crossentropy',optimizer=Adam(),metrics=['
accuracy'])

47 hist_adam=dmlp_adam.fit(x_train,y_train,batch_size=batch_siz,epochs=n_
epoch,validation_data=(x_test,y_test),verbose=2)

48

49 # Adagrad 옵티마이저를 사용하는 모델

50

51

삭제

52

53

54 # RMSprop 옵티마이저를 사용하는 모델

55

56

삭제

57

58

과제 #3

```
59 # 네 모델의 정확률을 출력
```

```
60
```

```
61
```

```
62
```

```
63
```

```
64
```

```
65 import matplotlib.pyplot as plt
```

```
66
```

```
67 # 네 모델의 정확률을 하나의 그래프에서 비교
```

```
68
```

```
69
```

```
70
```

```
71
```

```
72
```

```
73
```

```
74
```

```
75
```

```
76
```

```
77
```

```
78
```

```
79
```

```
80
```

```
81
```

```
82
```


삭제

삭제

al_

과제 #3

```
67 #####
68 # 여기부터 : 아래 함수를 수정하여 본인만의 새로운 모델을 만드시오
69 #####
70 def run_proposed_model(x_train, y_train):
71     reset_random_seeds()
72     model = Sequential()
73     model.add(Dense(***,activation='relu',input_shape=(784,)))
74     model.add(Dense(***,activation='relu'))
75     #model.add(Dense(***,activation='relu')) #레이어 추가하고 싶은 경우 예시
76     model.add(Dense(10,activation='softmax'))
77     model.compile(loss='categorical_crossentropy', optimizer=Adam(),
78                   metrics=['accuracy'])
79
80     # [*주의] 변경하면 안되는 옵션 : epochs, batch_size, _train, _text
81     model.fit(x_train, y_train, batch_size=256, epochs=20)
82     return model
```



과제 #3

■ 결과를 출력하는 코드 부분

```
108 #####
109 # 내가 개선한 모델의 정확률을 기존과 비교하며 출력
110 if g_pro_model_acc > (g_org_model_acc + 0.3):
111     print('SUCCESS! Difference: {0:0.3f}'.format(
112         (g_pro_model_acc - g_org_model_acc)))
113 else:
114     print('TRY DIFFERENTLY! Difference: {0:0.3f}'.format(
115         (g_pro_model_acc - g_org_model_acc)))
116 print('TF {0} under GPU {1}'.format(
117     tf.__version__, tf.test.is_gpu_available()))
118 import sys
119 print("python version", sys.version)
120 #####
121 # *** capture the below prints and put them in the report ***
```

■ 출력결과 예시

```
SUCCESS! Difference: 1.
TF 2.11.0 under GPU True
python version 3.9.16 (main, Dec 7 2022, 01:11:51)
[GCC 9.4.0]
```

과제 #3

■ Requirement

- **Baseline**(기본모델) 대비 정확도 차이값이 0.3% 이상의 모델 만들기
 - 총 2개의 파일 : 코드(.py or .ipynb) + 보고서(.docx or .pdf) 제출(압축파일 금지)
- 코드 : 1개의 파일에 Baseline과 your model이 모두 들어가 있어야 함★
- 보고서 :
 - Baseline(주어진 코드) 대비 무엇을 수정했는지 서술 및 관련 코드를 캡처하여 첨부
 - 최종 결과 캡처 2개(Baseline vs. Mine)

```
SUCCESS! Difference: 1.0
TF 2.11.0 under GPU True
python version 3.9.16 (main, Dec 7 2022, 01:11:51)
[GCC 9.4.0]
```

- "SUCCESS!" → 과제 성공을 의미함
- Baseline 대비 몇% 증감했는지 서술
- **Runtime environment**
 - Versions (딥러닝 라이브러리 버전, 파이썬 버전 필수)
 - Cloud or local


과제 #3

■ Requirement

- Runtime environment
 - Versions (딥러닝 라이브러리 버전, 파이썬 버전 필수)
 - **Cloud** or local
 - » Cloud일 경우 CPU/GPU/TPU 중 어떤 리소스 사용하였는지 서술 필수
 - » Local일 경우, GPU 장치 성능을 기록해야 함(예컨대 시리즈 이름 3080 Ti, 메모리 11GB).



노트 설정

하드웨어 가속기
GPU 

Colab를 최대한 활용하려면 필요하지 않은 경우 GPU를 사용하지 않는 것이 좋습니다. [자세히 알아보기](#)

☐ 백그라운드 실행

브라우저를 닫은 후에도 노트북을 계속 실행하고 싶으신가요? [Colab Pro+ 버전으로 업그레이드](#)

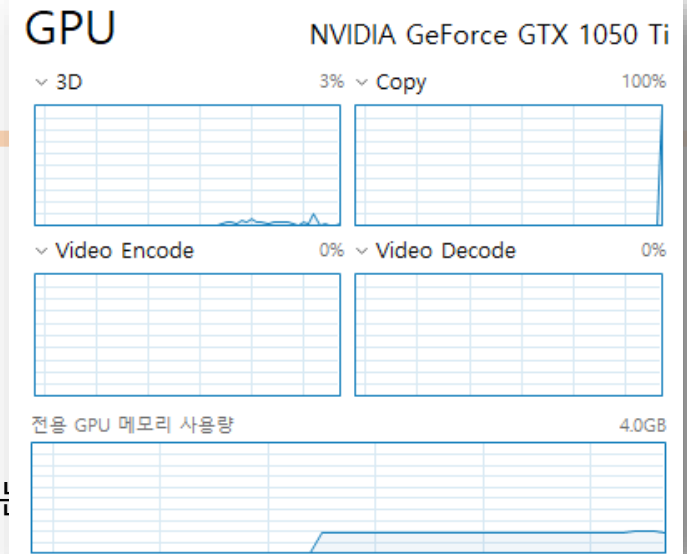
☐ 이 노트를 저장할 때 코드 셀 출력 생략

취소 저장

과제 #3

■ Requirement

- Runtime environment
 - Versions (딥러닝 라이브러리 버전, 파이썬 버전 필수)
 - Cloud or **local**
 - » Cloud일 경우 CPU/GPU/TPU 중 어떤 리소스 사용하였는지
 - » Local일 경우, **GPU 장치 이름을** 기록해야 함.
 - » 또한, **tensorflow-gpu**를 사용했는지 여부를 표시해야 함. (Anaconda Prompt에서 출력 必)



```
Anaconda Prompt
six 1.16.0 pyhd3eb1b0_0
sqlite 3.37.0 h2bbff1b_0
tensorboard 2.6.0 py_1
tensorboard-data-server 0.6.0 py38haa95532_0
tensorboard-plugin-wit 1.6.0 py_0 anaconda
tensorflow 2.3.0 mkl_py38h37f7ee5_0
tensorflow-base 2.3.0 eigen_py38h75a453f_0
tensorflow-estimator 2.3.0
```

CPU버전 TF
설치 예시

TF GPU버전일 경우, 이런 lib 이름으로 출력됨

```
tensorflow-gpu 2.3.0
testpath 5.0.0 pyhd3eb1b0_0
tornado 6.1 py38h2bbff1b_0
traitlets 5.1.1 pyhd3eb1b0_0
urllib3 1.25.11 py_0 anaconda
vc 14.2 h21ff451_1
vs2015_runtime 14.27.29016 h5e58377_2
wcwidth 0.2.5 pyhd3eb1b0_0
webencodings 0.5.1 py38_1
werkzeug 1.0.1 py_0 anaconda
wheel 0.37.1 pyhd3eb1b0_0
win_inet_pton 1.1.0 py38_0 anaconda
wincertstore 0.2 py38haa95532_2
winpty 0.4.3 4
wrapt 1.12.1 py38he774522_1 anaconda
yarl 1.6.2 py38he774522_0 anaconda
zipp 3.7.0 pyhd3eb1b0_0
zlib 1.2.11 vc14h1cdd9ab_1 [vc14] anaconda
```

(aibasic) C:\Users\wfori>conda list --n aibasic

참고 : Windows에서 Tensorflow GPU 설치하기

- 간단 설치법 : <https://youtu.be/M4urbN0fPyM>
- GPU 사용 가능하게 하고 실험하기를 권장함

과제3 - 부록(랭킹반영 원하는 경우 한정★)

- 별도의 과제로 Kaggle에서 오픈하며, 성적에 반영하지 않음
- 데드라인 다름(혼동하지 않고 기존과제에 영향주지 않기 위함)

■ 과제3 코드/리포트 기준으로 성능개선이 0.8% 이상인 경우 한정

- 과제 채점 이후, 높은 성능 순으로 20명 이내 대상자를 공개할 예정
- 마감 : 대상자 공개한 시점부터 1주일 이내

■ 과제3 랭킹용 조건

- 목표 : 최고 성능을 갖는 코드
 - 성능평가 : 정확도 차이값
- 제안모델에서의 조건
 - Train/test dataset은 동일해야 함(캐글에서 올라오는 데이터로 진행할 것)
 - **Epoch** 제한은 최대 20회(조기종료 가능)
 - 전처리 제한 없음(기존 train set에서 augmentation 허용)
 - 전이학습 등 모델구조 및 레이어 종류 제한 없음
- 랭킹 1~2위의 경우, 본인의 코드를 설명하는 동영상 녹화본 제공 필수(5분 ~ 10분 사이)
 - 사전 연락 후 동영상 제공 가능한 경우에만 랭킹으로 집계함