**EfficientNet을 이용한 이미지 분류**

2201011

V0.2

1. 필요 패키지 설치
   1. 아래 명령어로 필요한 패키지 자동 설치, 모두 최신버전(220107)

>> pip3 install -r requirements.txt

* torch==1.7.1
* torchvision==0.8.2
* numpy==1.19.5
* pillow==8.4.0
* matplotlib==3.3.4
* sklearn
* tqdm==4.62.3

1. 데이터
   1. data\_220104 폴더 안에 ‘anomal’과 ‘normal’ 폴더로 멧돼지 열화상 사진을 나눠놓음
   2. 랜덤으로 train/valid/test셋을 8:1:1 비율로 구분하여 ‘data\_220104\_split’ 폴더에 저장
   3. 최종적으로 ‘data\_220104\_split’ 폴더를 사용
      1. anomal | normal : 1236 | 9855
      2. anomal data proportion(train:valid:test) = 984 : 123 : 129
      3. nomal data proportion(train:valid:test) = 7880 : 985 : 990
      4. data proportion(train:valid:test) = 8864 : 1108 : 1119
2. 학습 관련 코드
   1. 쥬피터노트북으로 1\_train\_classification.ipynb 실행
   2. 데이터(‘data\_220104\_split’)를 이용하여 train셋으로 학습, valid셋으로 점검 한 후 최종 test셋으로 accuracy와 f1 score

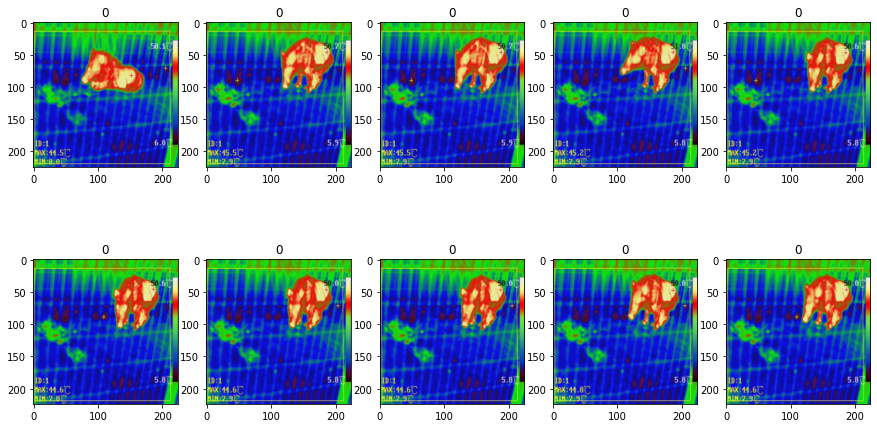


Figure 1, 데이터셋 예시, 0:비정상, 1: 정상

* 1. Batchsize는 GPU에 따라 키우거나 줄여도 되나, 128일 때가 가장 이상적임
  2. 모델 입력 차원: 3x224x224
  3. EfficientNet 모델을 fine-tuning 학습
  4. 파라미터 개수: 4,010,110
  5. 학습 추이
     1. 총 25epoch 학습 중2~5epoch 사이에서 최고 성능 달성. 데이터가 너무 일관되서 쉽다!

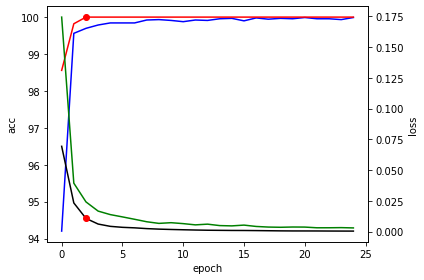
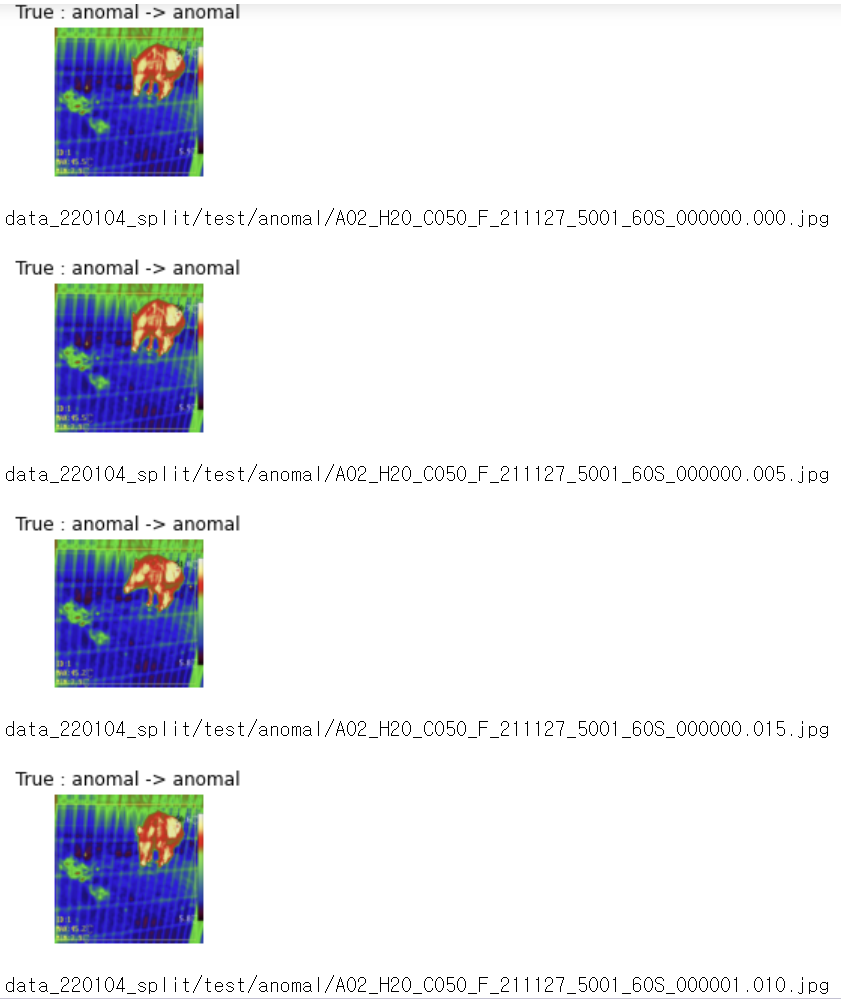
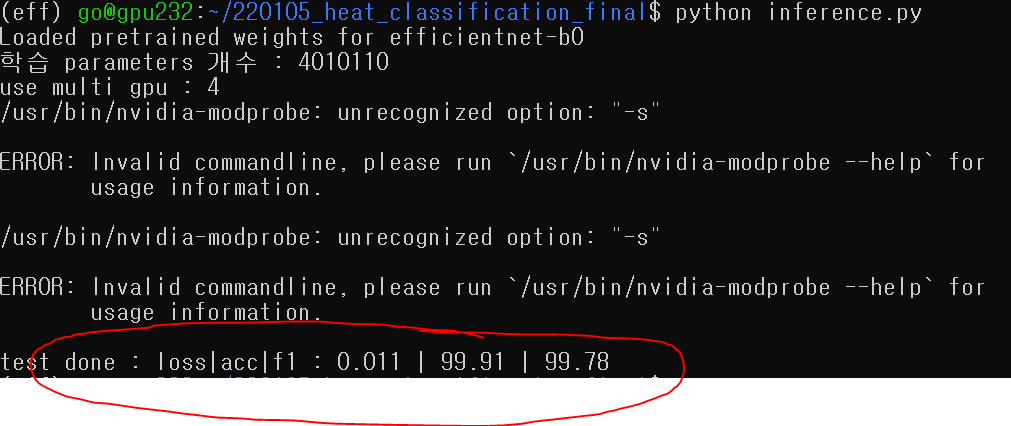


Figure 2, epoch에 따른 train/valid 정확도 추이

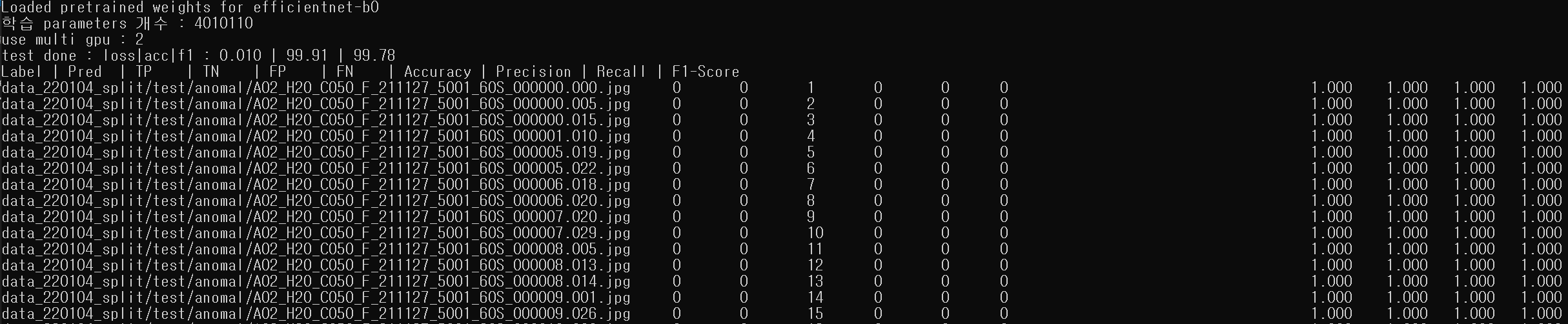
* 1. 학습 결과(모델)파일은 ‘output’ 폴더에 자동 저장됨
  2. 몇번째 epoch일 때 validation 정확도와 f1 점수가 같이 저장
     1. Model\_epoch\_정확도\_f1.pt 형식
     2. Ex) model\_2\_100.00\_100.00.pt’ # 2epoch일 때 validation셋 정확도/f1 모두 100
  3. 테스트 결과
     1. 테스트 데이터셋에 Loss와 정확도, f1 score를 측정
     2. 정확도 99.91, f1 99.78 달성
        1. test done : loss|acc|f1 : 0.011 | 99.91 | 99.78
        2. label에 맞게 잘 분류되었는지 직접 확인 완료



1. 추론 코드
   1. 2\_inference.py 파일을 파이썬에서 아래 명령어로 실행
      1. >> python3 2\_inference.py
      2. Test 데이터셋의 loss와 정확도(accuracy), f1 score 값이 나옴
      3. Acc&F1 모두 99 이상의 성능을 달성!!



요청한 유효성 지표에 맞춰 print 로그 출력





출력 결과 result.csv로 저장

