1. 환경 구축

1) 트레인 및 테스트는 성능좋은 본인 pc(ey pc)에서 하는걸로 가정(Geforce RTX 3090)

2) 이 메뉴얼 문서파일과 함께 제공한 코드폴더를 D에다가 다운받는다

(어디든 상관없지만 D가 관리하기 수월)

3) 아나콘다를 열어 다운받은 폴더경로 안으로 이동

텍스트이(가) 표시된 사진

자동 생성된 설명

ㄴ명령어: cd D:\GANomalyBallCounterbar

4) 해당 폴더안에 requirements\_cuda11.1.yaml파일로 가상환경 생성하기

텍스트이(가) 표시된 사진

자동 생성된 설명

ㄴ명령어: conda env create -f requirements\_cuda11.1.yaml -n [새로 만들 가상환경 이름]

5) 방금 만든 가상환경 활성화

텍스트이(가) 표시된 사진

자동 생성된 설명

ㄴ명령어: activate gan

2. 학습(Train)

1) ./data에 본인이 학습하고자하는 학습,테스트 데이터들을 일정한 형식으로 넣어줌

ex)

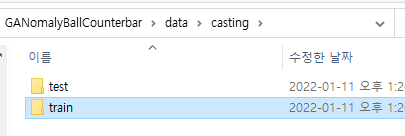
데이터 셋 구성 방법

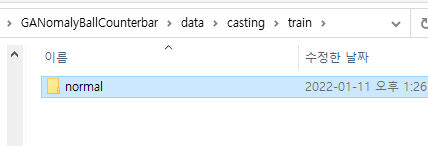
텍스트이(가) 표시된 사진

자동 생성된 설명data폴더가 없으면 생성 후 좌측과 같이 디렉토리를 구성.

[데이터셋 이름]: default는 “casting”으로 되어있음. 변경 시 option.py에 반영 필요.

--트레인데이터 구성



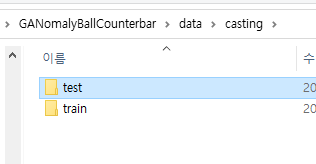


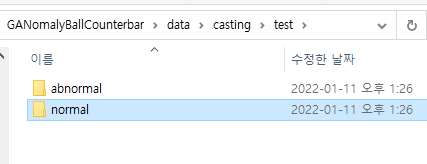
텍스트이(가) 표시된 사진

자동 생성된 설명

\*트레인데이터에는 모방해서 생성할 정답 이미지들만 있어야 하므로 normal만 있으면 됌

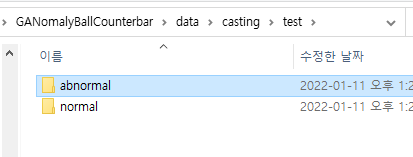
--테스트데이터 구성

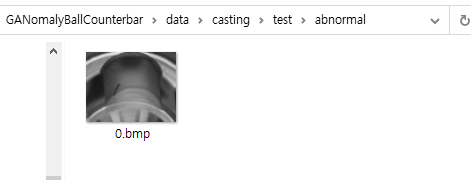




텍스트이(가) 표시된 사진

자동 생성된 설명





ㄴ샘플abnormal사진(그림판으로 그려서 일부러 결함을 만들어줌(회색 라인)

2) train하기



ㄴ명령어: python train.py

\*참고1\* 기존에 트레인한 모델부터 이어서 트레인하는법

- 기존에 트레인해놨던 모델 파일들(.pth)중에 가장 마지막 에폭 또는 사용자가 임의로 선택하고자하는 에폭의 모델파일(netD\_(특정에폭).pth, netG\_(특정에폭).pth)을 골라서 각각 netD.pth, netG.pth로 rename해준다. 고른 netD와 netG 모델파일은 에폭 숫자가 동일한 파일이어야함



ㄴ명령어: python.train.py --resume “모델(netD.pth, netG.pth파일) 두개가 전부 들어있는 폴더경로”

\*참고2\* 트레인을 하면서도 각 에폭마다 성능이 어느정도로 나오는지를 확인할 수 있게하기위하여 ./output/ganomaly/casting/train/images 안에 각 에폭마다 batchsize만큼의 사진들(현재는 2로해놓음)이 인풋이미지(train폴더안에있던) 그대로인 reals\_epoch(에폭숫자).png로 저장되고, gan이 만든 fake이미지들도 fake\_epoch(에폭숫자).png로 차례대로 저장되게끔 해놓음.

맨 처음 train을 시작했을때의 fake\_epoch01~10.png사진들

텍스트, 점수판, 바둑판식, 스크린샷이(가) 표시된 사진

자동 생성된 설명

거의 무슨사진인지 식별이 불가함

시간이지나 에폭 400정도가 됐을 때의 fakes\_epoch401~410.png 사진들

소파, 좌석이(가) 표시된 사진

자동 생성된 설명

에폭이 증가하면서 얼추 train normal데이터로 넣어줬던 사진들과 비슷하게 gan이 이미지를 생성하는것을 알 수 있음 🡪 따라서 test로 에폭400~410.pth파일을 써야 성능이 잘 나올것 유추 가능

\*참고3\* 쭉 트레인 시켜놓으면 weights폴더에 너무 많은 .pth파일들이 저장될 수 있음. 1에폭마다 저장시키지 않고 텀을 줘서 10에폭, 50에폭, 100에폭마다 저장시키고 싶으면 -- save\_weight\_freq [원하는 에폭텀] 옵션을 주면됨



ㄴ명령어: python train.py -- save\_weight\_freq [원하는 에폭텀]

3. 테스트(Test)

1) 트레인 시 ./output/ganomaly/casting/train/weights폴더에 netD와 netG가 에폭대로 쌓임

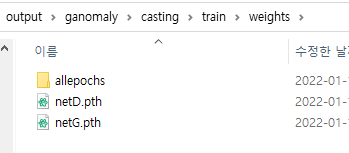
2) 게중에 성능이 좋았던 epoch으로 netG,netD세트를 고른 후 이름을 바꿔줌

ㄴnetD\_선택에폭.pth 🡪 netD.pth로, netG\_선택에폭.pth 🡪 netG.pth로

테이블이(가) 표시된 사진

자동 생성된 설명

위 에폭파일들은 allepochs폴더안에 넣어줌으로써 정리한 후 netD.pth와 netG.pth만 남겨도 됨



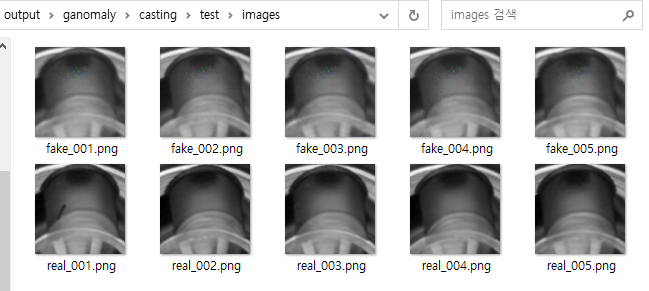
3) 해당폴더에 test명령어.txt파일로도 남겨두었던 테스트 명령어는 다음과같다

텍스트이(가) 표시된 사진

자동 생성된 설명

ㄴ명령어: python train.py --phase "test" --save\_test\_images --load\_weights --batchsize 1

4) 위 명령어를 실행시켜주면 ./output/ganomaly/casting/test/images폴더에 ./data/casting/test 폴더안에 들어있는 (abnormal, normal폴더안에있는 사진들 전부더함) 사진들에대한 fake영상과 real영상들이 저장된다



- 샘플 사진들은 기존 test data에 normal로 위 사진에서 real\_002~005.png들을 넣었었고 abnormal케이스로는 위 사진에서 real\_001.png를 넣었었다. Abnormal과 normal 모든 케이스에 대하여 정상 fake영상이 잘 만들어진걸 볼 수 있다. (아직 학습이 충분히 덜된 모델이라 fake영상에 하늘색 잡음이 껴있음🡪더 오랜 학습 후에 잡음 사라질 수 있음)

\*\*참고\*\*

테스트 시 한 개의 파일만을 테스트하고 싶을 경우

python train.py --phase "test" --save\_test\_images --load\_weights --batchsize 1 --testOnefilepath "(파일경로)"

이렇게 쳐주면 testResultOneFiles 에 해당 파일의 fake영상과 해당파일인 real영상만 저장된다