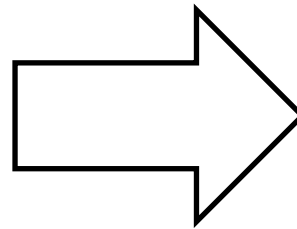


AI Project B

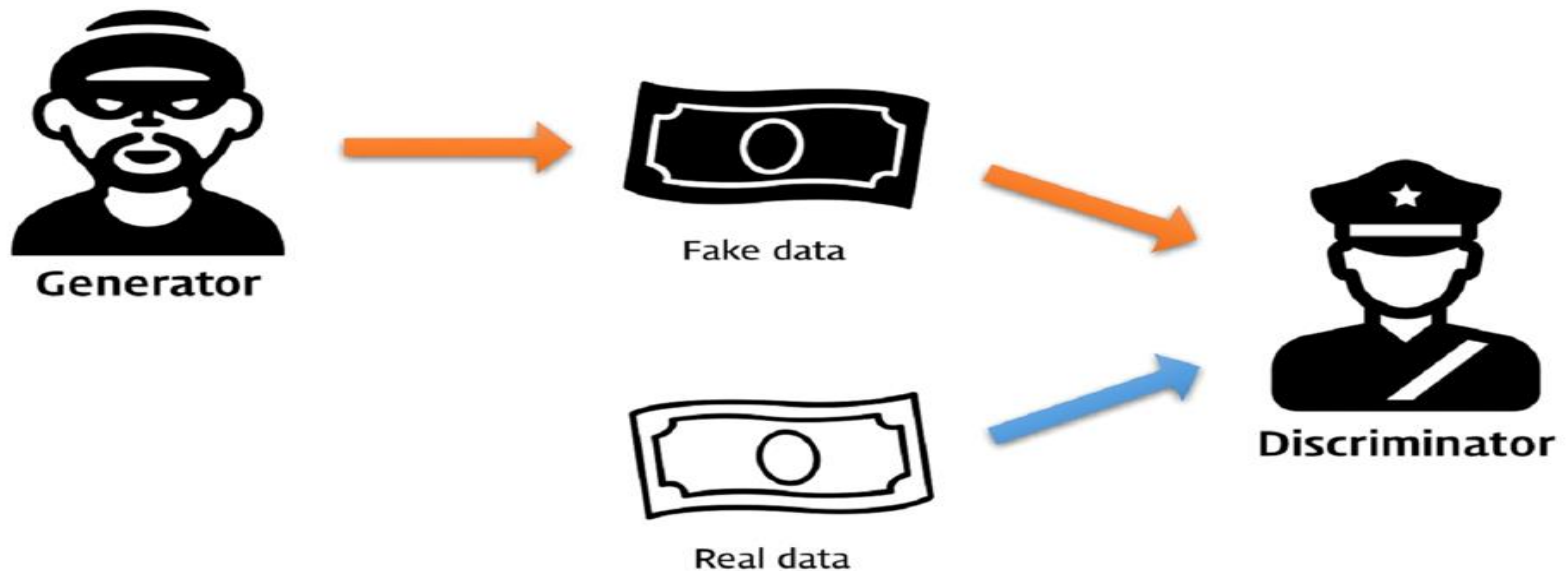
박준민

1. 프로젝트 목표

DC GAN을 이용하여 새로운 개의 이미지 만들기



2 -1. GAN 이란?



두 개의 네트워크 이용 (Generator, Discriminator)

Generator는 random noise를 받아 가짜 이미지 생성

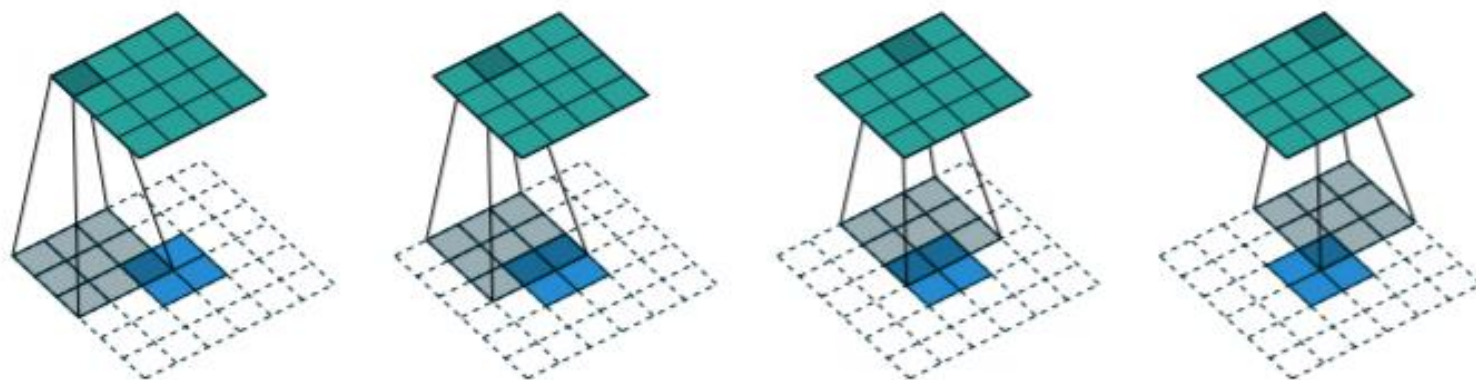
Discriminator는 Generator가 생성한 이미지와 진짜 이미지를 받아
진짜와 가짜 이미지를 구분하는 방향으로 학습

Generator는 Discriminator를 속이는 방향으로 학습

2 - 2. DC GAN이란?

- GAN의 응용
- Linear Layer와 Pooling Layer 최대한 배제
- Discriminator는 Convolution Layer 사용하여 진짜 사진과 가짜 사진의 특징 추출
- Generator는 Transposed convolution Layer 사용하여 특징을 만들어냄

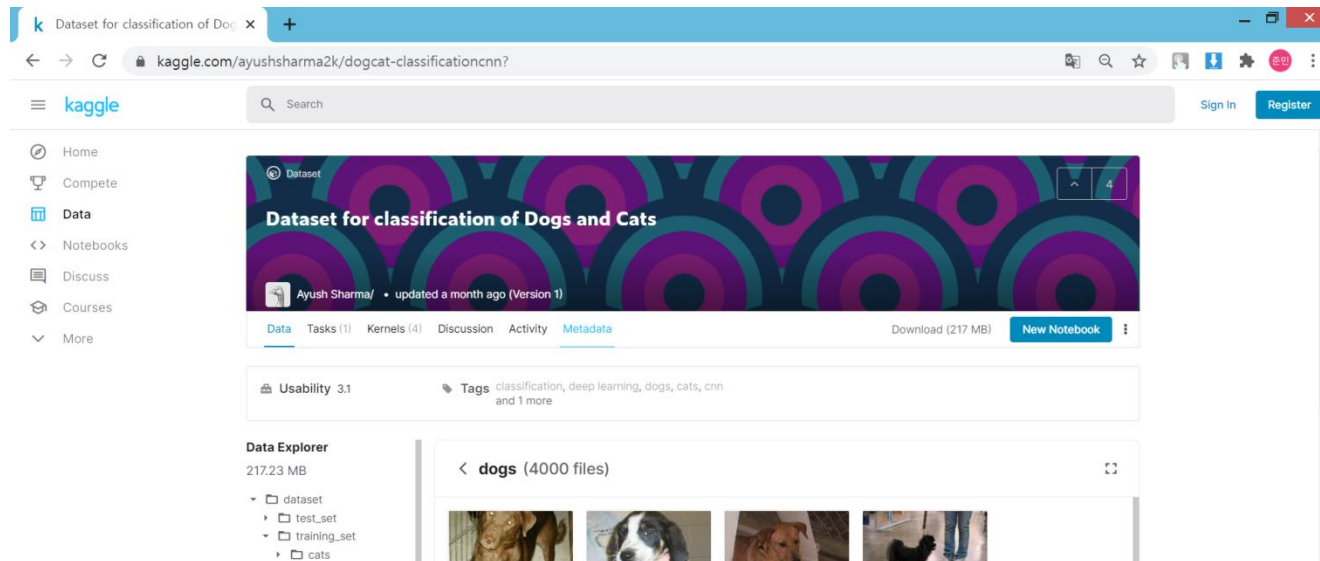
Cf) Transposed Convolution Layer



일반 적인 convolution layer는 filter를 이용해 특징을 추출해 output을 더 작게 만들지만,
Transposed convolution layer는 위의 그림의 아래에서 위로 input을 더 크게 만든다.
해상도를 높이는 작업 등을 할 때 사용된다.

3. DC GAN 구현하기

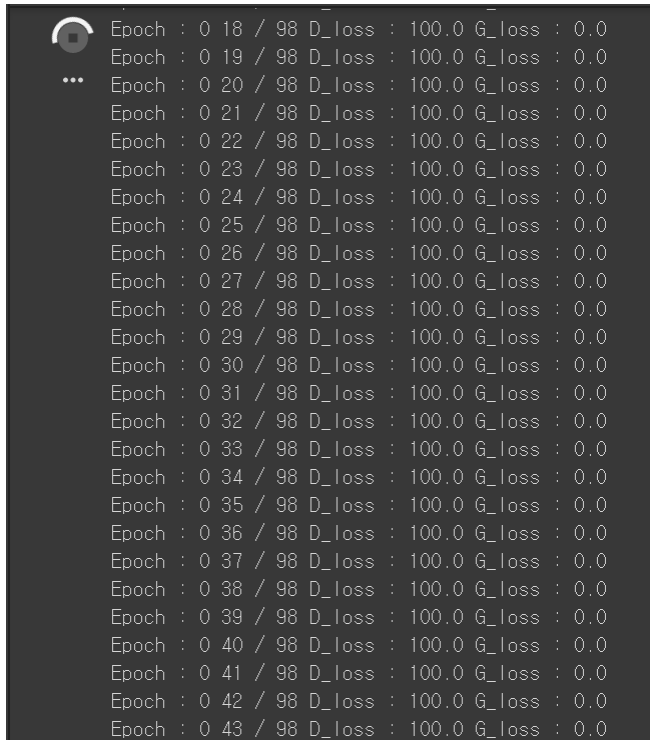
3-1) 데이터 모으기



Kaggle 개, 고양이 이미지 분류 project에서 개 사진 다운

3. DC GAN 구현하기

3-2) 네트워크 구현하고 학습하기



```
Epoch : 0 18 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 19 / 98 D_loss : 100.0 G_loss : 0.0
...
Epoch : 0 20 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 21 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 22 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 23 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 24 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 25 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 26 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 27 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 28 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 29 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 30 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 31 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 32 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 33 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 34 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 35 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 36 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 37 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 38 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 39 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 40 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 41 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 42 / 98 D_loss : 100.0 G_loss : 0.0
Epoch : 0 43 / 98 D_loss : 100.0 G_loss : 0.0
```

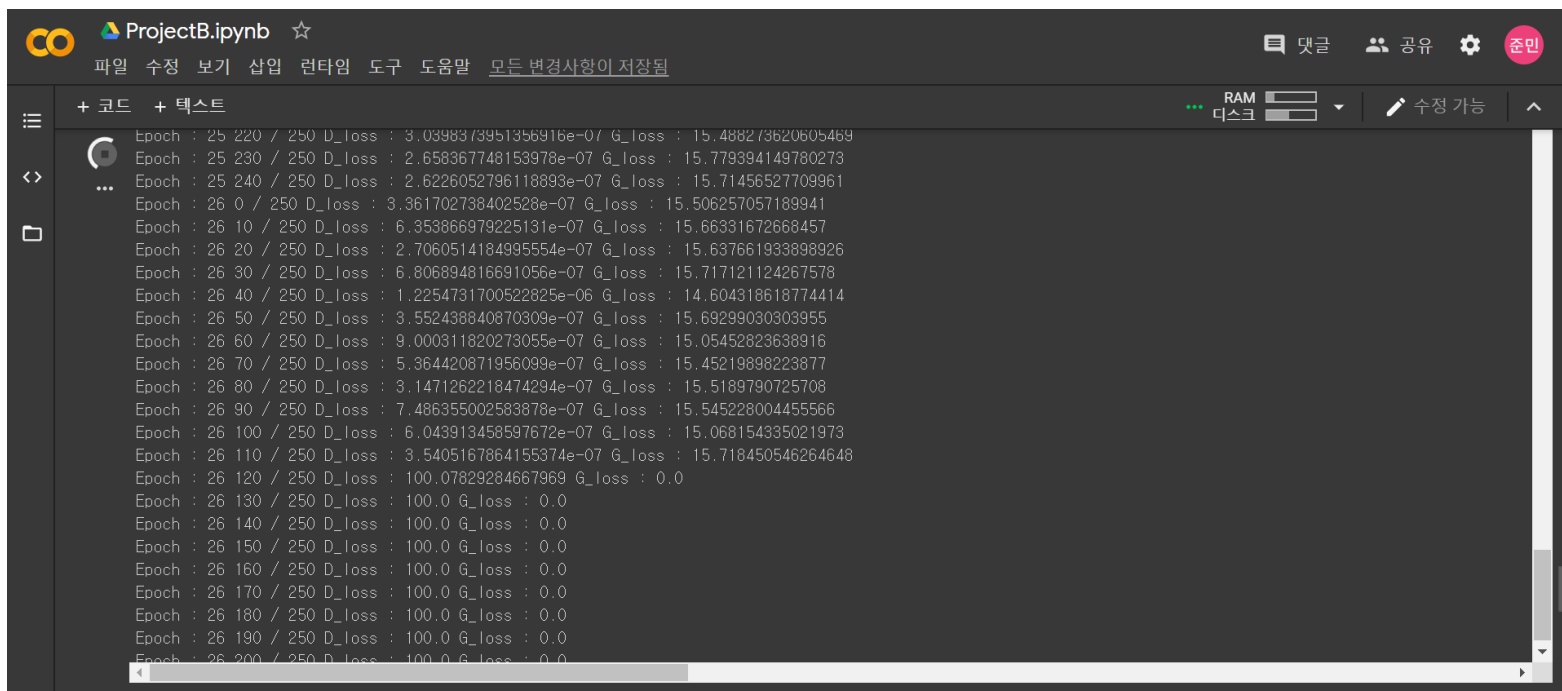
Generator와 Discriminator의 학습률을 0.01로 학습 시켰지만 1epoch부터 D_loss는 100, G_loss는 0으로 발산

적절한 학습률 찾는 과정을 거쳤다.

3. DC GAN 구현하기

3-2) 네트워크 구현하고 학습하기

ex) Generator lr : 0.000001, Discriminator lr :0.0001



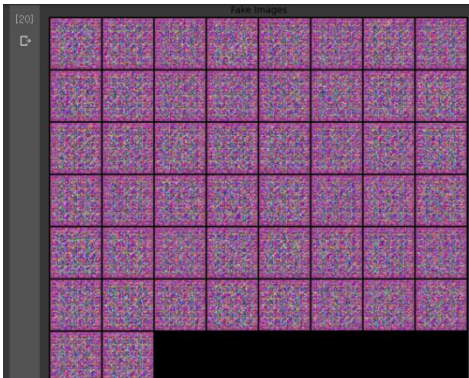
```
Epoch : 25 220 / 250 D_loss : 3.0398373951356916e-07 G_loss : 15.488273620605469
Epoch : 25 230 / 250 D_loss : 2.658367748153978e-07 G_loss : 15.779394149780273
Epoch : 25 240 / 250 D_loss : 2.6226052796118893e-07 G_loss : 15.71456527709961
Epoch : 26 0 / 250 D_loss : 3.361702738402528e-07 G_loss : 15.506257057189941
Epoch : 26 10 / 250 D_loss : 6.353866979225131e-07 G_loss : 15.66331672668457
Epoch : 26 20 / 250 D_loss : 2.7060514184995554e-07 G_loss : 15.637661933989826
Epoch : 26 30 / 250 D_loss : 6.806894816691056e-07 G_loss : 15.717121124267578
Epoch : 26 40 / 250 D_loss : 1.2254731700522825e-06 G_loss : 14.604318618774414
Epoch : 26 50 / 250 D_loss : 3.552438840870309e-07 G_loss : 15.69299030303955
Epoch : 26 60 / 250 D_loss : 9.000311820273055e-07 G_loss : 15.05452823638916
Epoch : 26 70 / 250 D_loss : 5.364420871956099e-07 G_loss : 15.45219898223877
Epoch : 26 80 / 250 D_loss : 3.1471262218474294e-07 G_loss : 15.5189790725708
Epoch : 26 90 / 250 D_loss : 7.486355002583878e-07 G_loss : 15.545228004455566
Epoch : 26 100 / 250 D_loss : 6.043913458597672e-07 G_loss : 15.068154335021973
Epoch : 26 110 / 250 D_loss : 3.5405167864155374e-07 G_loss : 15.718450546264648
Epoch : 26 120 / 250 D_loss : 100.07829284667969 G_loss : 0.0
Epoch : 26 130 / 250 D_loss : 100.0 G_loss : 0.0
Epoch : 26 140 / 250 D_loss : 100.0 G_loss : 0.0
Epoch : 26 150 / 250 D_loss : 100.0 G_loss : 0.0
Epoch : 26 160 / 250 D_loss : 100.0 G_loss : 0.0
Epoch : 26 170 / 250 D_loss : 100.0 G_loss : 0.0
Epoch : 26 180 / 250 D_loss : 100.0 G_loss : 0.0
Epoch : 26 190 / 250 D_loss : 100.0 G_loss : 0.0
Epoch : 26 200 / 250 D_loss : 100.0 G_loss : 0.0
```


3. DC GAN 구현하기

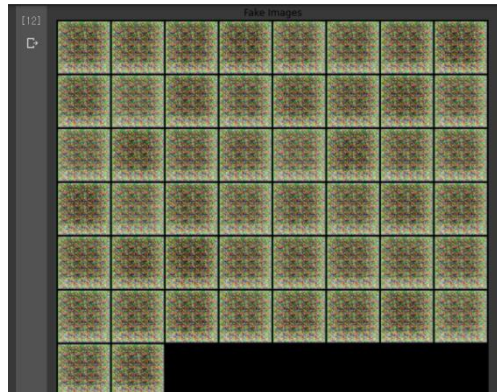
3-2) 네트워크 구현하고 학습하기

epoch 수를 작게 해서 학습이 되는지 확인

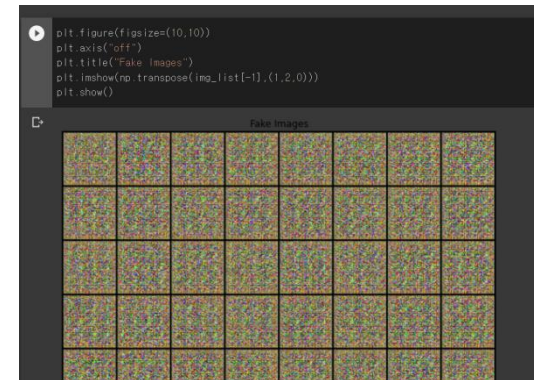
1 epoch



10 epoch



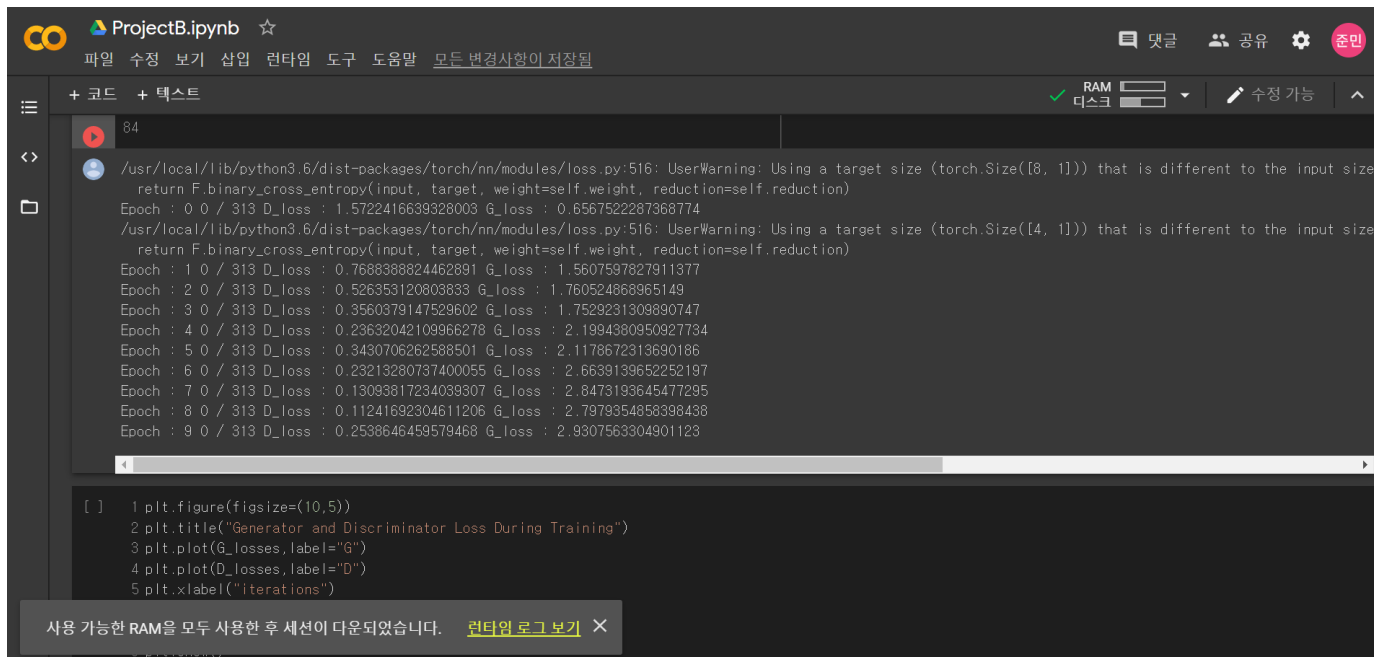
20 epoch



학습이 제대로 되고 있지 않거나 가짜 이미지를 나타내는 코드에 문제가 있을 것 이라고 판단

3. DC GAN 구현하기

3-2) 네트워크 구현하고 학습하기



The screenshot shows a Jupyter Notebook titled 'ProjectB.ipynb' with a dark theme. The code cell contains training logs for a DC GAN. The logs show 9 epochs of training with G_loss and D_loss values. A warning message is displayed at the top of the code cell: 'UserWarning: Using a target size (torch.Size([8, 1])) that is different to the input size'. Below the logs, there is a plot command: `plt.figure(figsize=(10,5))`, `plt.title("Generator and Discriminator Loss During Training")`, `plt.plot(G_losses,label="G")`, `plt.plot(D_losses,label="D")`, and `plt.xlabel("iterations")`. At the bottom of the notebook, a message states: '사용 가능한 RAM을 모두 사용한 후 세션이 다운되었습니다. [런타임 로그 보기](#)'.

```
84
/usr/local/lib/python3.6/dist-packages/torch/nn/modules/loss.py:516: UserWarning: Using a target size (torch.Size([8, 1])) that is different to the input size
  return F.binary_cross_entropy(input, target, weight=self.weight, reduction=self.reduction)
Epoch : 0 0 / 313 D_loss : 1.5722416639328003 G_loss : 0.6567522287368774
/usr/local/lib/python3.6/dist-packages/torch/nn/modules/loss.py:516: UserWarning: Using a target size (torch.Size([4, 1])) that is different to the input size
  return F.binary_cross_entropy(input, target, weight=self.weight, reduction=self.reduction)
Epoch : 1 0 / 313 D_loss : 0.7688388824462891 G_loss : 1.5607597827911377
Epoch : 2 0 / 313 D_loss : 0.526353120803833 G_loss : 1.760524668965149
Epoch : 3 0 / 313 D_loss : 0.3560379147529602 G_loss : 1.7529231309890747
Epoch : 4 0 / 313 D_loss : 0.23632042109966278 G_loss : 2.1994380950927734
Epoch : 5 0 / 313 D_loss : 0.3430706262588501 G_loss : 2.1178672313690186
Epoch : 6 0 / 313 D_loss : 0.23213280737400055 G_loss : 2.6639139652252197
Epoch : 7 0 / 313 D_loss : 0.13093817234039307 G_loss : 2.8473193645477295
Epoch : 8 0 / 313 D_loss : 0.11241692304611206 G_loss : 2.7979354858398438
Epoch : 9 0 / 313 D_loss : 0.2538646459579468 G_loss : 2.9307563304901123

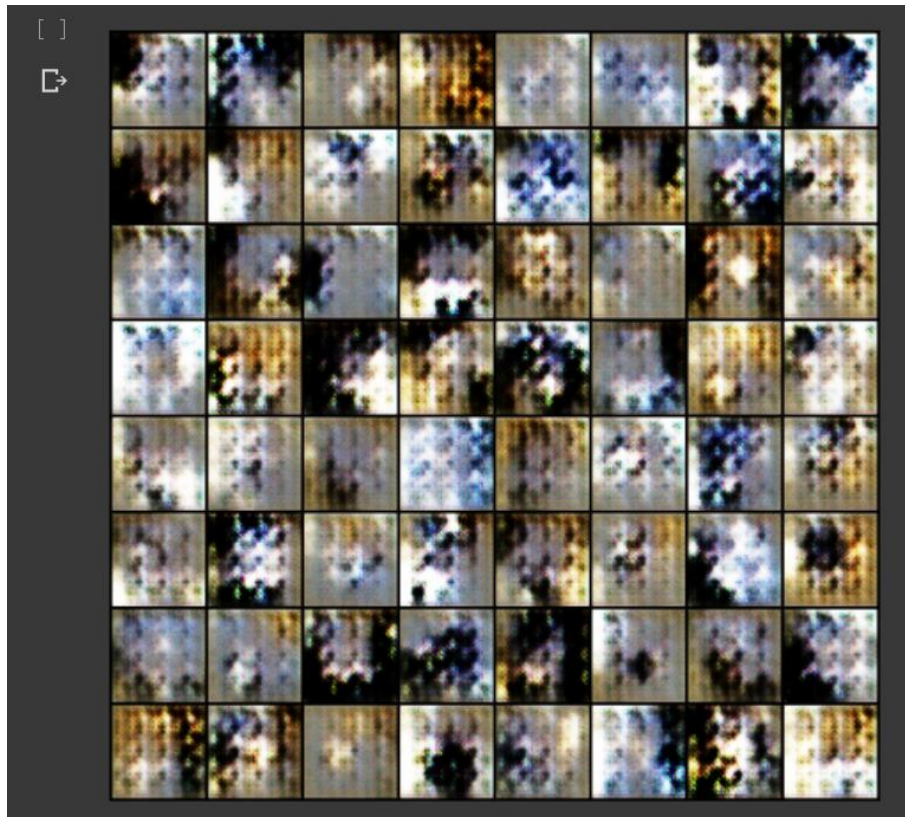
[ ] 1 plt.figure(figsize=(10,5))
2 plt.title("Generator and Discriminator Loss During Training")
3 plt.plot(G_losses,label="G")
4 plt.plot(D_losses,label="D")
5 plt.xlabel("iterations")
```

사용 가능한 RAM을 모두 사용한 후 세션이 다운되었습니다. [런타임 로그 보기](#) ✕

Colab에서 사용 가능한 RAM 제한을 넘겨버리는 문제 발생
=> 코드 수정

3. DC GAN 구현하기

3-2) 네트워크 구현하고 학습하기



학습 코드를 수정하고 적절한 Learning rate를 찾은 후, 3epoch 만큼 학습시킨 결과를 보았다.

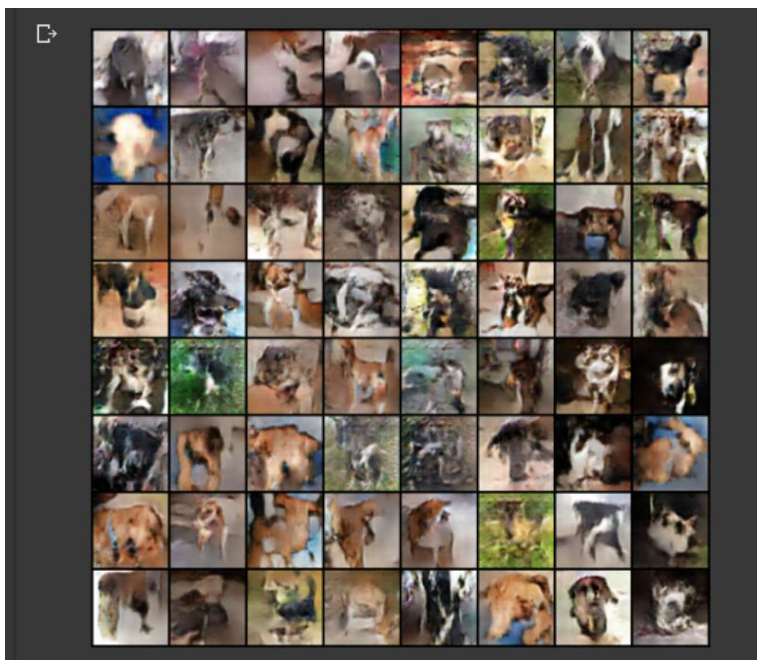
앞선 결과와는 다르게 개의 형태를 나타내기 시작했다.

Epoch 수를 늘려 학습시켜 보기로 했다.

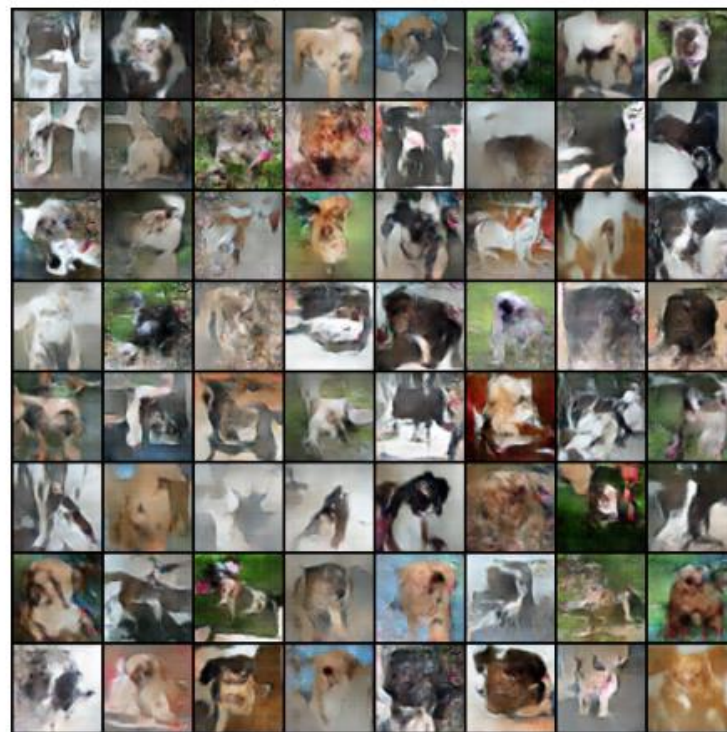
3. DC GAN 구현하기

3-2) 네트워크 구현하고 학습하기

50 epoch



100 epoch



3. DC GAN 구현하기

3-3) 결과



100 epoch을 학습시킨 결과

개의 형체가 꽤 잘 나타나는 것을
확인할 수 있었다.

4. 한계

학습시키는 중, loss가 발산하고 RAM 사용량을 초과하는 등의 문제가 있었다.

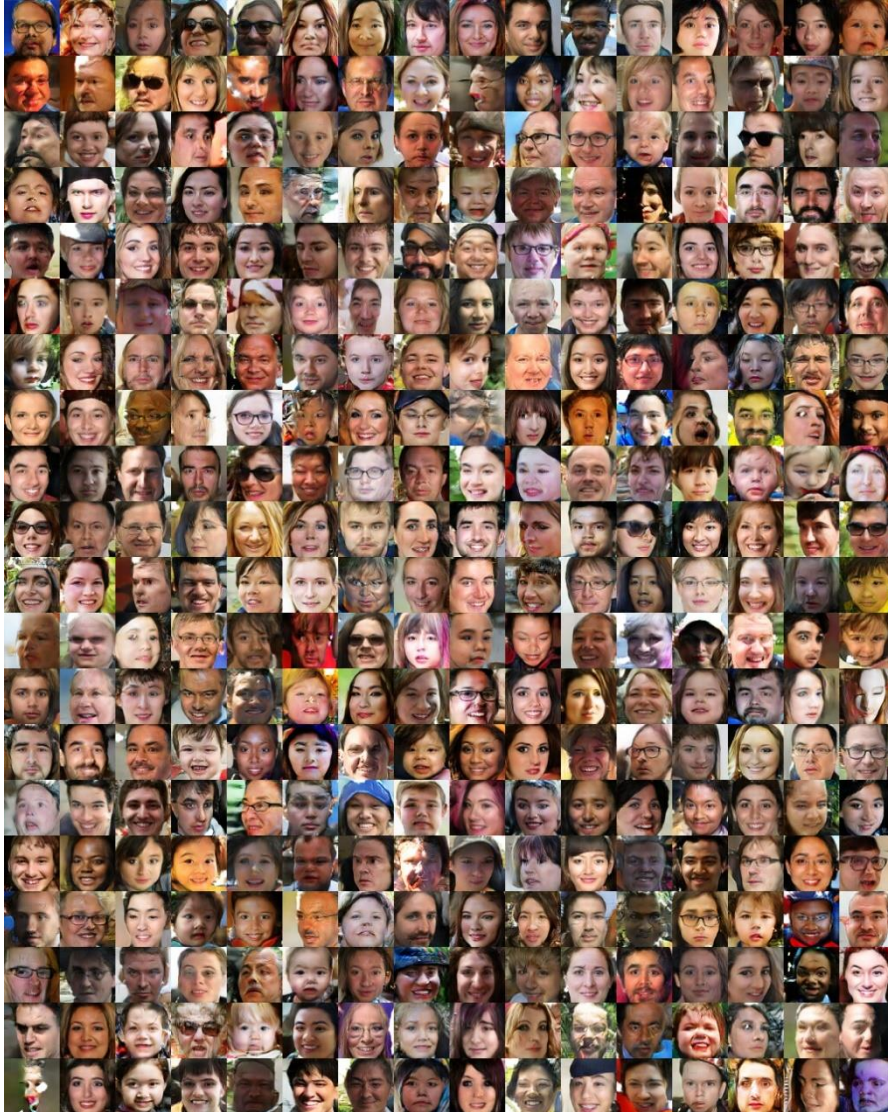
이를 해결하기 위해 학습 코드를 수정하고 원본 이미지의 크기를 수정했다.

이 과정에서 원본 사진의 크기를 128*128로 일관 조정했으며 학습시키는 사진의 크기는 64*64였다.

그 결과 Input 이미지의 크기와 해상도가 많이 낮아졌으며 RAM 사용량과 시간의 문제로 많은 epoch을 학습시킬 수 없었다.

이로 인해 학습의 결과물의 해상도 또한 낮으며 형체는 알아볼 수 있지만 사람을 속일 수 있을 정도의 결과는 얻지 못했다.

4. 한계



인터넷에서 찾아 본 사람 얼굴 사진을 학습시켜 사람 얼굴 사진을 생성한 결과이다.

꽤 높은 결과를 볼 수 있다.

시간과 비용의 문제만 해결된다면 개의 이미지도 DC GAN을 사용해 만들어 낼 수 있을 것이다.