Combo GAN

Unrestrained Scalability for Image Domain Translation

이미지 변환 기술의 동향(Cycle GAN)

CycleGAN

- 지금까지의 이미지 학습을 통한 이미지 변환의 새로운 도약
- 만족스러운 이미지 변환 출력
- 2개의 domain 맞춤형 모델

이미지 변환 기술의 동향(Cycle GAN)

CycleGAN

- 더 많은 domain에 적용하려면 모델이 지수적으로 증가(모델의 폭발적 증가)
- 2개의 domain에 대한 학습도 많은 시간 소요

→ 결국 시간과 자원에 대한 제약

ComboGAN??

CycleGAN의 단점을 해결할 모델

여러 개의 domain에도 적용 가능한 모델

모델 학습이 빠른 모델

도메인의 수가 증가함에도 폭발적으로 모델이 증가하지 않는 모델

ComboGAN??

CycleGAN의 단점을 해결할 모델

여러 개의 domain에도
적용 가능한 모델

모델 학습이 빠른 모델

도메인의 수가 증가함에도 폭발적
으로 모델이 증가하지 않는 모델

ComboGAN!!

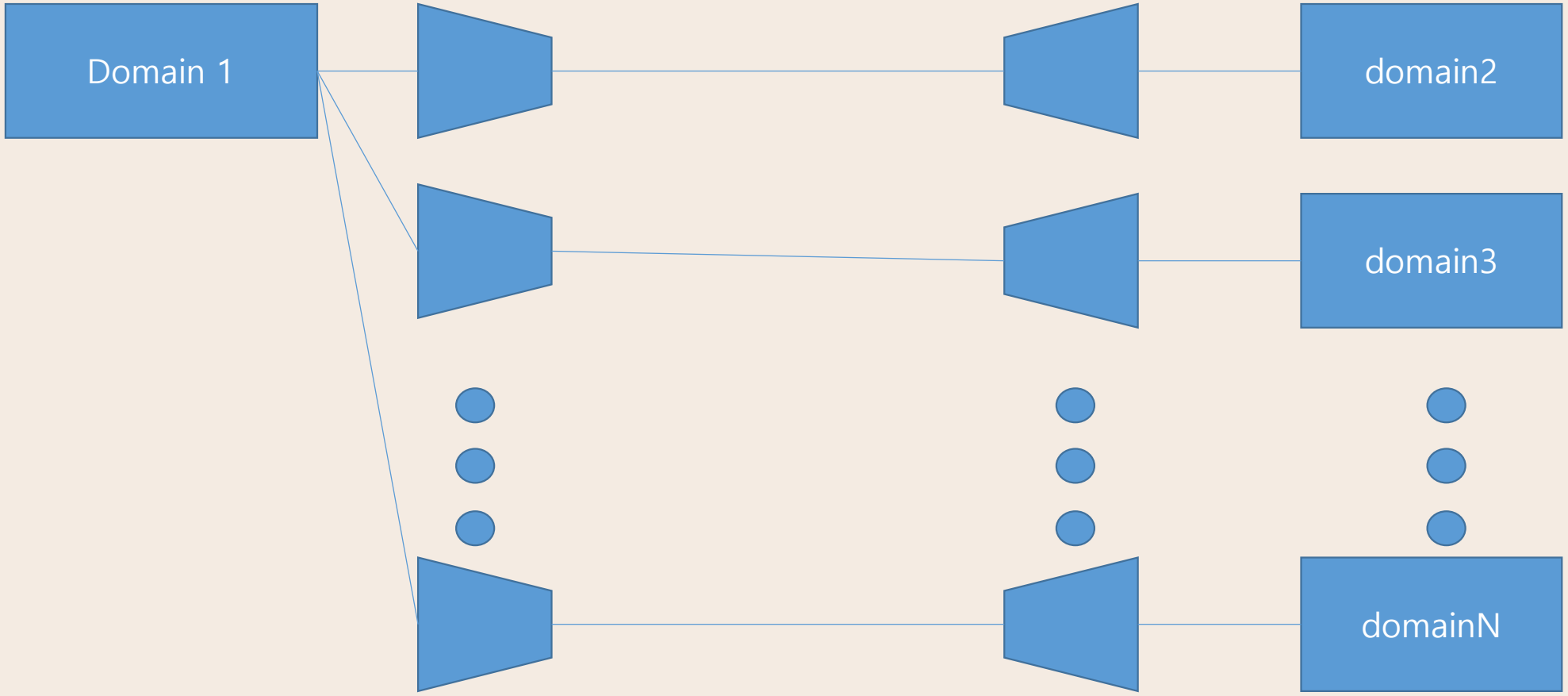
Image translation

한 도메인에 해당하는 이미지를 다른 도메인의 이미지로 변환하자!

Image translation

단순하게 생각해 본다면 domain이 n개인 경우 (n combination 2) 의 모델이 필요할 것이라고 생각할 수 있다.

Image translation



GAN

Generative adversarial networks

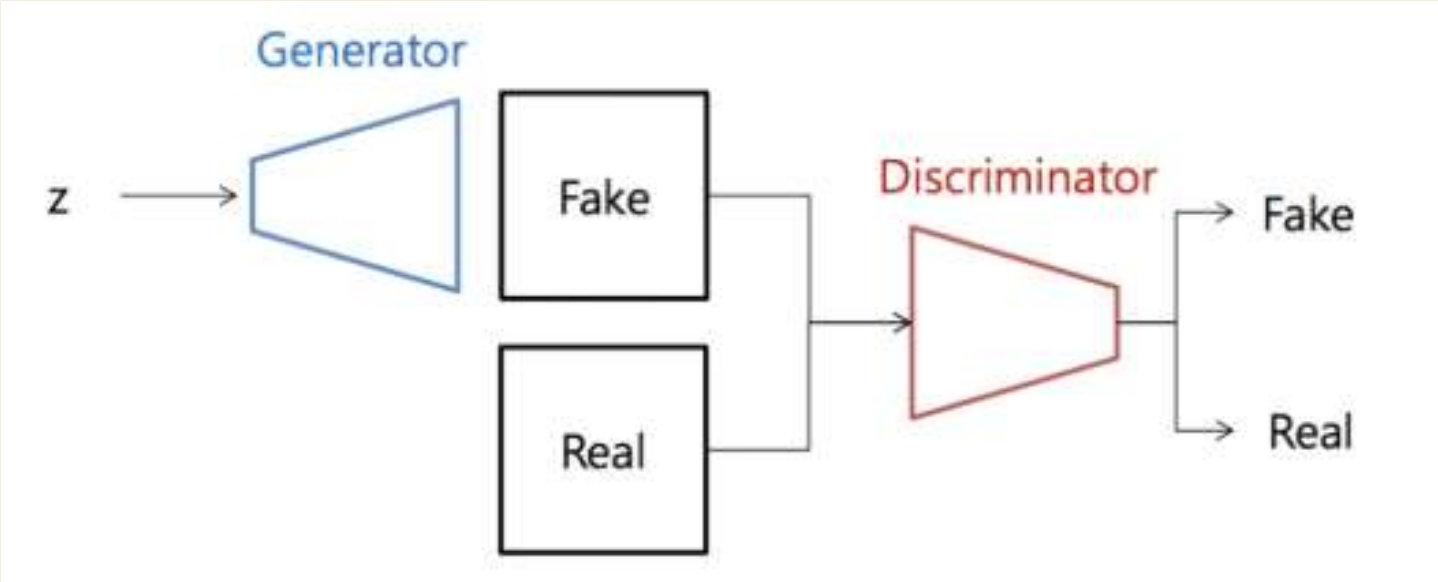
이미지 변환에서 자연스러운 이미지를 생성한다는 것은 어려운 일(challenging task)이다!

Generative adversarial networks

이미지 변환에서 자연스러운 이미지를 생성한다는 것은 어려운 일(challenging task)이다!

GAN

Generative adversarial networks



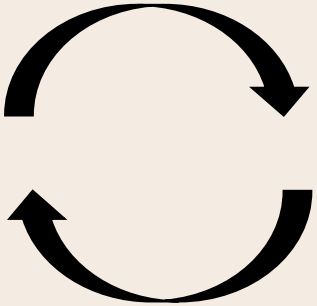
Discriminator은 sample x가 실제 데이터의 분포에서 나온 것인지 확률을 추정
Gernerator은 이미지 생성(이미지 변환)을 해서 discriminator가 속을 정도로 실제 사진과 유사한 사진을 만들고 싶어!!

Generative adversarial networks

Generator VS discriminator

Generator

나는 실사에 가까운 이미지를 만들어 너를 속일거야!



Discriminator

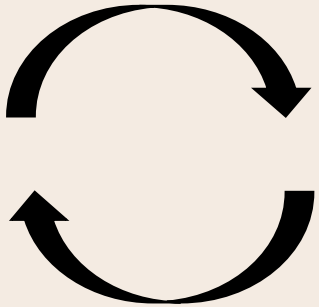
너가 어떤 이미지를 만들더라도 실사가 아니라고 판단해줄게.

Generative adversarial networks

Generator VS discriminator

Generator

나는 실사에 가까운 이미지를 만들어 너를 속일거야!



Discriminator

너가 어떤 이미지를 만들더라도 실사가 아니라고 판단해줄게.

서로 경쟁하며 서로의 성능을 높인다.

Generative adversarial networks

$$\min_G \max_D \mathbb{E}_x[logD(x)] + \mathbb{E}_z[log(1 - D(G(z))]$$

D(x)는 Discriminator가 x가 실사라고 판단할 확률 (1~0)
G(z)는 Generator가 생성한 이미지

Generator은 이걸 최소로 Discriminator은 이걸 최대화하고 싶어한다

Generative adversarial networks

컴퓨터 비전과 그래픽스의 많은 업무들은
A 도메인의 이미지 a를 B도메인 이미지 b로 변경하는 것.

Generative adversarial networks

컴퓨터 비전과 그래픽스의 많은 업무들은
A 도메인의 이미지 a를 B도메인 이미지 b로 변경하는 것.

Isola et al은 조건적인 Gan을 이용해 이것을 달성
(image to image translation framework)

Generator가 존재하는 이미지 a에 조건화된 이미지를 변환
이미지를 만들기 위해 vector 분포 p(z)에서 샘플링하는 것 대신 주어진 입력 이미지를 변경!

Generative adversarial networks

컴퓨터 비전과 그래픽스의 많은 업무들은
A 도메인의 이미지 a를 B도메인 이미지 b로 변경하는 것.

Isola et al은 조건적인 Gan을 이용해 이것을 달성
(image to image translation framework)

Generator가 존재하는 이미지 a에 조건화된 이미지를 변환
이미지를 만들기 위해 vector 분포 p(z)에서 샘플링하는 것 대신 주어진 입력 이미지를 변경!

하지만 이를 위해서는 두 도메인의 이미지 데이터가 필요함.
데이터들은 또한 해당하는 쌍으로 정렬되어 있어야 한다.
따라서 unsupervised한 이미지 변환이 불가!

Cycle GAN

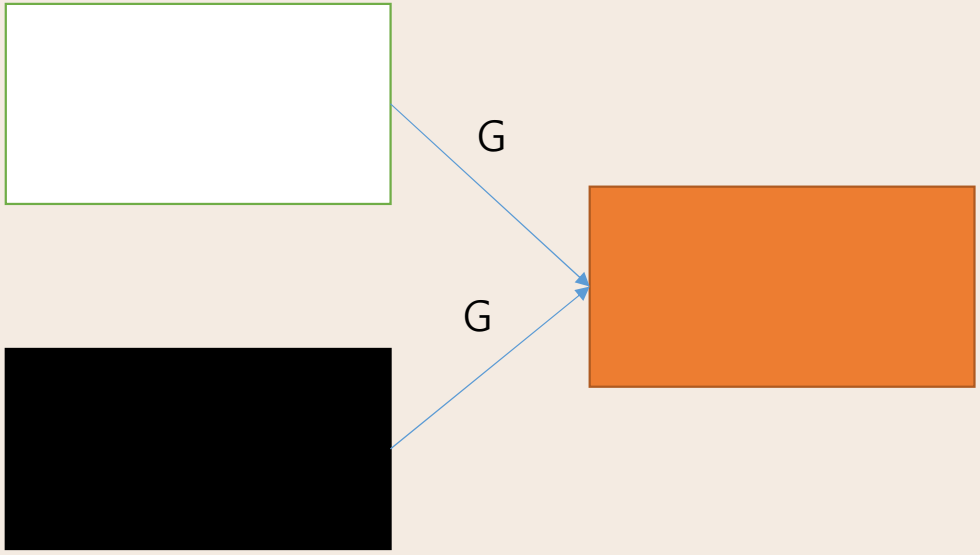
Cycle GAN

Gan을 응용한 것으로 unsupervised한 이미지 변환이 가능하다!
혁신적인 모델

Cycle GAN

기존의 GAN의 문제점

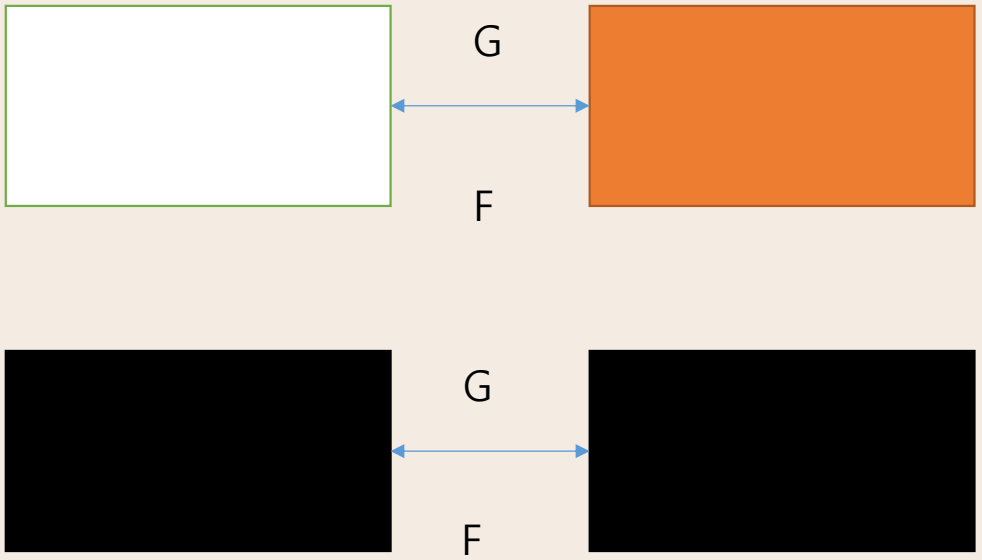
다른 사진이 같은 출력 이미지로 generation 될 수 있다!
→ input 이미지의 특성이 무시될 수 있다



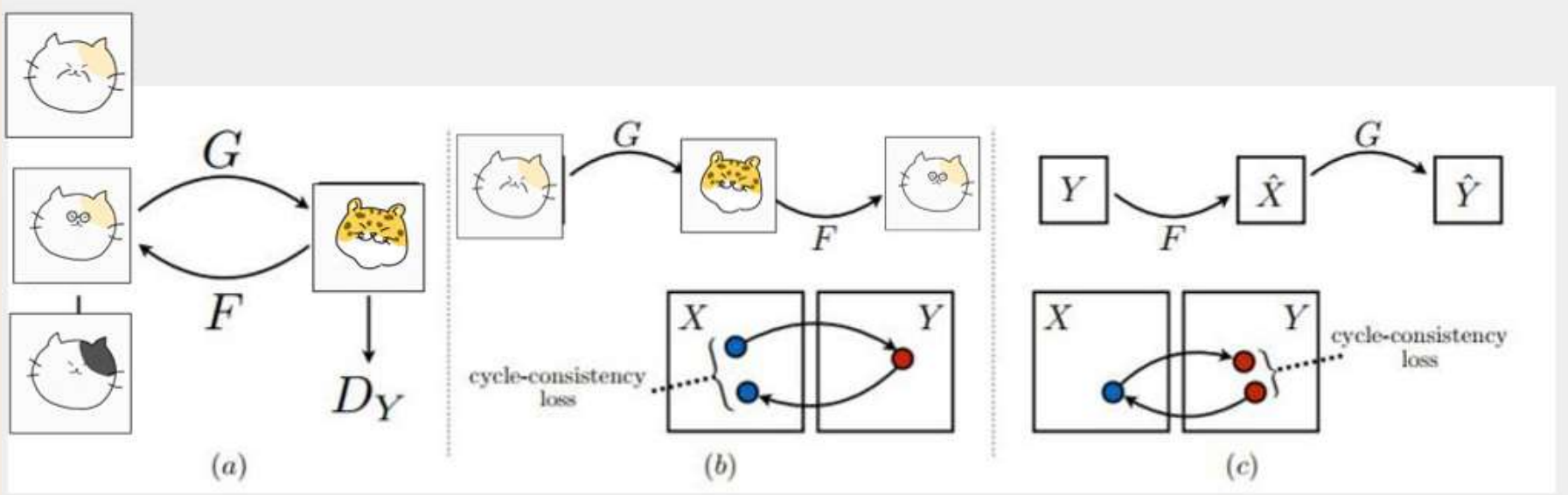
Cycle GAN

Cycle GAN의 컨셉

이미지가 generating된 이미지는
다시 원본 이미지로 reconstruct되도록 하자



Cycle GAN



$$\mathcal{L}_{cycle}(G, F) = \mathbb{E}_a[||F(G(a)) - a||_1] + \mathbb{E}_b[||G(F(b)) - b||_1]$$

Cycle GAN

$$\mathcal{L}_{GAN}(G, D_B, A, B) =$$
$$\mathbb{E}_b[(D_B(b) - 1)^2] + \mathbb{E}_a[D_B(G(a))^2]$$

GAN loss의 log부분을 squared하게 바꾸어 안정성을 높인다!

Cycle GAN

$$\mathcal{L}(G, F, D_A, D_B) = \lambda \mathcal{L}_{cycle}(G, F) + \mathcal{L}_{GAN}(G, D_B, A, B) + \mathcal{L}_{GAN}(F, D_A, B, A)$$

Cycle loss로 같은 target 이미지로 변환되는 것을 막고
Log 대신 squared를 사용해 안정성을 높이자

Cycle GAN

그러나 CycleGan의 이런 확장성은 두 네트워크가 두 도메인에 공동으로 연결되어 있어 좋지 않다!

Cycle GAN

그러나 CycleGan의 이런 확장성은 두 네트워크가 두 도메인에 공동으로 연결되어 있어 좋지 않다!

Ex) 도메인 A,B가 있는 Cycle GAN에서 도메인 C를 추가하려면??
→ 4개의 새로운 네트워크가 필요하다
A to C, C to A, B to C, C to B

Cycle GAN

그러나 CycleGan의 이런 확장성은 두 네트워크가 두 도메인에 공동으로 연결되어 있어 좋지 않다!

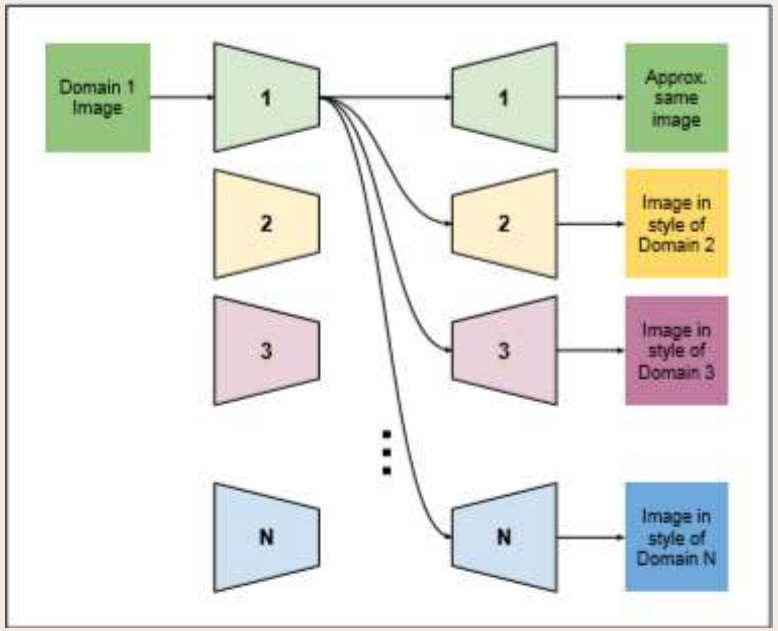
Ex) 도메인 A,B가 있는 Cycle GAN에서 도메인 C를 추가하려면??
→ 4개의 새로운 네트워크가 필요하다
A to C, C to A, B to C, C to B

도메인이 증가함에 따라 네트워크(모델)이 폭발적으로 증가!!

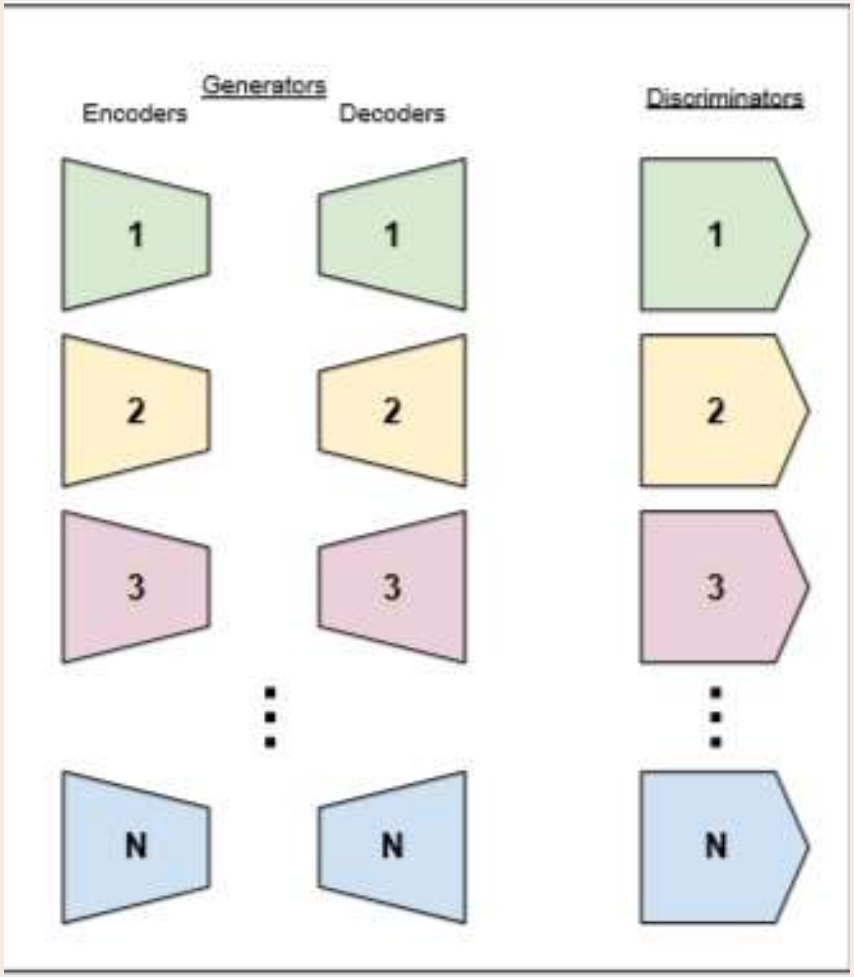
Combo GAN

Combo GAN

도메인과 네트워크를 서로 분리하자(**Decoupling the Generators**)
→ 도메인 수 증가 시 모델의 폭발적인 증가를 막을 수 있다.



Combo GAN



- Generator를 반으로 나누어 Encoder, Decoder 라고 한다.
- 이름에서 알 수 있듯이 인코더와 디코더를 빌딩 블록처럼 결합 가능.
- **도메인이 증가함에 따라 generator도 선형적으로 증가!!!!**

Combo GAN

기존 GAN의 generator의 개수 변화

도메인이 n개라고 했을 때 generator의 개수는
(N combination 2) * 2
= n(n-1)
도메인이 증가하면 폭발적으로 증가!

Combo GAN

기존 GAN의 generator의 개수 변화

도메인이 n개라고 했을 때 generator의 개수는
(N combination 2) * 2
= n(n-1)
도메인이 증가하면 폭발적으로 증가!

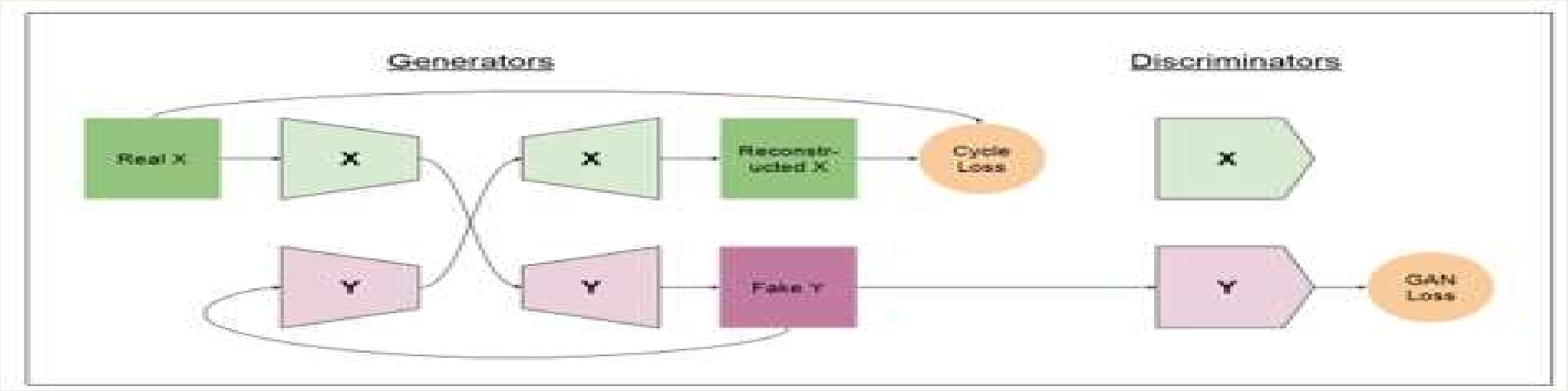
Combo GAN의 generator개수 변화

도메인 하나당 generator 한 개!(generator=encoder+decoder)
도메인이 증가함에 따라 선형적으로 증가!

Combo GAN

Combo GAN의 학습과정

각 iteration의 시작에서 n개의 도메인중 2개의 도메인을 랜덤하게 고른다!
고른 두개의 도메인에 대해 cycle GAN에서 했던 것처럼 학습을 진행!



Experiments and results (Datasets)

6000 images of the Alps mountain

14-paintes' painting

nVidia Titan X GPU

Experiments and results

N개의 도메인을 학습할 때 100*N에폭으로 진행.

→ 이는 기본 Cycle GAN훈련이 두개의 도메인을 학습할 때 200에폭이 필요한 것을 따른 것.

→ generator하나당 100에폭이 필요하다

Experiments and results

Experiments and results

Combo GAN

Cycle GAN

Experiments and results

Combo GAN

Cycle GAN

Both are excellent results

Experiments and results

Combo GAN	**4(domain)*1(generator per domain)*100 = 400 epoch**
Cycle GAN	**(4 combination 2) * 2 * 100 =1200 epoch**

Experiments and results

각 도메인(화가)의 특성에 맞게 그림이 훌륭하게 변환되었다!!!!

Experiments and results

Combo GAN	**14(domain)*1(generator per domain)*100 = 1400 epoch 220시간 소요**
Cycle GAN	**(14 combination 2) * 2 * 100 =18200 epoch 2860시간(4달) 소요**

Experiments and results

Cycle GAN의 결과와 비교할 수 있는 좋은 결과

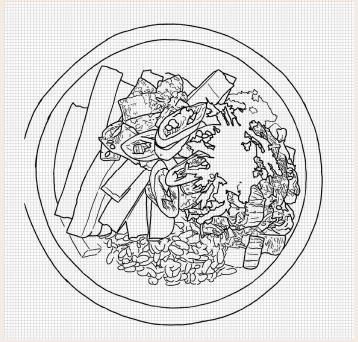
+

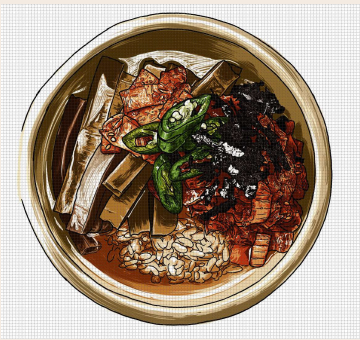
Cycle GAN보다 훨씬 빠른 속도

Experiments and results

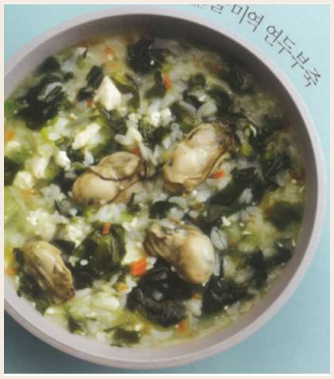
Experiments and results

Experiments and results

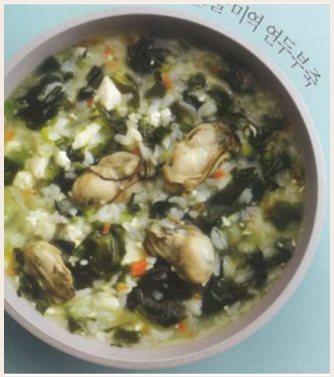




Experiments and results



Experiments and results



Experiments and results

해본 실험 내용들

- Epoch 최대 1400까지 학습 (논문에 따르면 domain 당 100epoch 즉 우리 데이터에서는 500epoch이면 충분하다 나와있다. 에폭이 늘어날 수록 결과가 나빠지기도 하였다.)

- Dropout 사용/미사용(dropout rate=0.5)

- Weight for cycle loss 조절 (default : 10.0 -> 5.0, 7.0, 8.0 등 여러 시도
  (결과물을 보면 cycle loss를 너무 신경써서 generate에 소심해 보인다고 판단하여 weight를 줄여서 시도.)

- Weight for forward loss 조절 (default : 0.0 -> 0.5등 다양한 시도.
  (cycle loss의 weight를 줄이고 forward loss의 weight를 증가시켜 reconstruct 보다 forward 작업에 좀 더 집중하도록 시도하였다.)

- 초기 Learning rate조절과 Learning rate decay 사용.

- 그나마 제일 괜찮은 결과는 800epoch, no learning rate decay, no dropout, weight for cycle loss : 10.0 , weight for forward loss : 0.0 초기 learning rate : 0.002

Experiments and results

```
loss_G = self.loss_G[self.DA] + self.loss_G[self.DB] + \
         (self.loss_cycle[self.DA] + self.loss_cycle[self.DB]) * self.lambda_cyc + \
         (loss_idt_A + loss_idt_B) * self.lambda_idt + \
         (loss_enc_A + loss_enc_B) * self.lambda_enc + \
         (loss_fwd_A + loss_fwd_B) * self.lambda_fwd
loss_G.backward()
```

```
self.parser.add_argument('--lambda_cycle', type=float, default=10.0, help='weight for cycle loss (A -> B -> A)')
self.parser.add_argument('--lambda_identity', type=float, default=0.0, help='weight for identity "autoencode" mapping (A -> A)')
self.parser.add_argument('--lambda_latent', type=float, default=0.0, help='weight for latent-space loss (A -> z -> B -> z)')
self.parser.add_argument('--lambda_forward', type=float, default=0.0, help='weight for forward loss (A -> B; try 0.2)')
```

Experiments and results

예상되는 문제점

1. ComboGan 모델이 우리 데이터와는 맞지 않는다??
   → (X) 위 모델은 input 이미지와 출력하고자 하는 것의 visual content가 유사한 경우 적합하다고 함. 따라서 우리 모델은 음식사진 -> 음식그림 으로 변환을 원하므로 이 모델은 적합하다.

Experiments and results

예상되는 문제점

1. ComboGan 모델이 우리 데이터와는 맞지 않는다??
   → (X) 위 모델은 input 이미지와 출력하고자 하는 것의 visual content가 유사한 경우 적합하다고 함. 따라서 우리 모델은 음식사진 -> 음식그림 으로 변환을 원하므로 이 모델은 적합하다.

2. 데이터 부족??
   → 논문에서의 실험결과는 변환이 잘된 결과를 보여주었고 이때 alps_seasons의 데이터의 경우 4개의 도메인, 6000장의 이미지를 사용하였고 14_painting의 데이터의 경우 14개의 도메인, 대략 1만장의 이미지로 학습을 진행. 반면 내가 직접 훈련에 사용한 것은 5개의 도메인, 대략 350장의 이미지(data augmentation random crop 사용) 가능성이 있는 문제점.

Experiments and results

예상되는 문제점

1. ComboGan 모델이 우리 데이터와는 맞지 않는다??
   → (X) 위 모델은 input 이미지와 출력하고자 하는 것의 visual content가 유사한 경우 적합하다고 함. 따라서 우리 모델은 음식사진 -> 음식그림 으로 변환을 원하므로 이 모델은 적합하다.

2. 데이터 부족??
   → 논문에서의 실험결과는 변환이 잘된 결과를 보여주었고 이때 alps_seasons의 데이터의 경우 4개의 도메인, 6000장의 이미지를 사용하였고 14_painting의 데이터의 경우 14개의 도메인, 대략 1만장의 이미지로 학습을 진행. 반면 내가 직접 훈련에 사용한 것은 5개의 도메인, 대략 350장의 이미지(data augmentation random crop 사용) 가능성이 있는 문제점.

3. Cycle loss에 집중하는 것 같다.
   → 변환된 이미지를 보면 기존 이미지에서 style은 잘 바꾸지 않고 색깔과 명도 채도 등 만 바꾼 것 같음. -> reconstruct 연산을 위해 style transfer을 사리는 것 같다??

Experiments and results

논문 리뷰를 진행하며 들었던 의문점.

이 논문이 쓰여진 것은 2017년 12월.
논문에 따르면 CycleGan과 비슷한 성능의 이미지 변환이 가능하며 그 훈련시간은 엄청나게 감소시켜준다!!!!!!!
→ 와 엄청 좋아보여

Experiments and results

논문 리뷰를 진행하며 들었던 의문점.

이 논문이 쓰여진 것은 2017년 12월.
논문에 따르면 CycleGan과 비슷한 성능의 이미지 변환이 가능하며 그 훈련시간은 엄청나게 감소시켜준다!!!!!!!
→ 와 엄청 좋아보여

그런데 도대체 왜 ComboGan에을 구글링 해보았을때 관련 다른 다른 글은 하나도 찾아볼 수 없고 이 모델을 사용했다~ 이 것을 이용해서 실험을 해보았다 라는 말은 어디에도 찾아볼 수가 없었다...
분명 논문에 따르면 이렇다할 단점이 없고 기존에 있는 Cycle Gan의 놀라운 개선 버전이라고 할 수 있는데 왜 그 누구도 안 쓸까...?

Experiments and results

궁금한 점

Sgd+momentum을 사용하면 batch size에 따라 test 결과가 달라지는가??

제가 생각하는 이 모델의 결과가 안 좋은 이유가 맞는 것인지