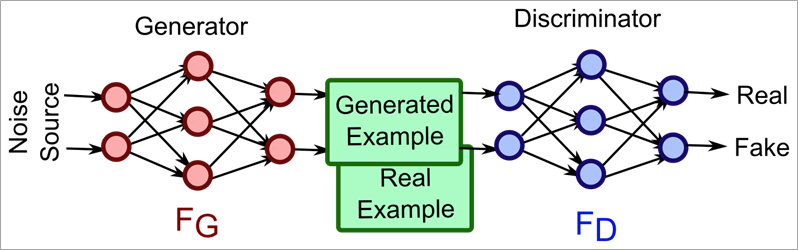
GAN



GAN의 architecture. Fully-Connected 구조이다.



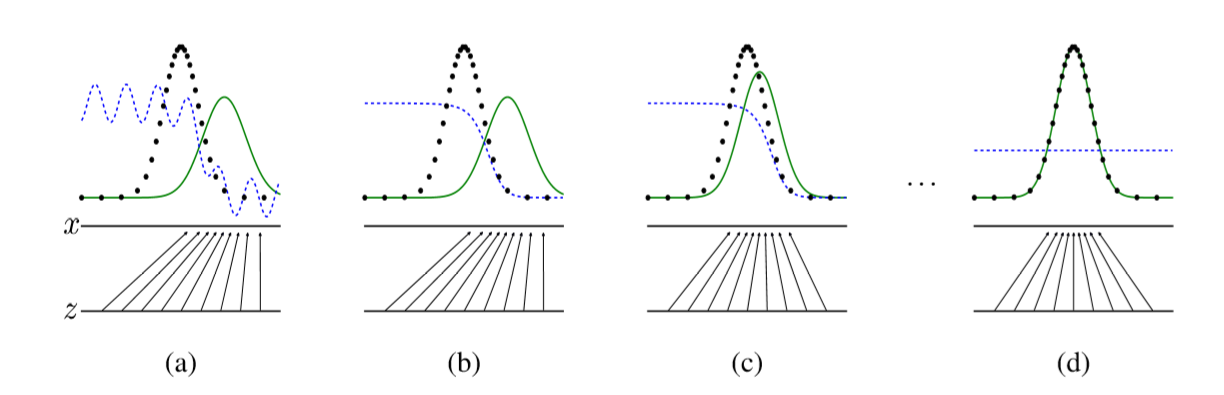
* 우항 첫 번째: reat data x를 판별자(D)에 넣었을 때 나오는 결과에 log를 취해 얻는 기대값
* 우항 두 번째: fake data z를 생성자(G)에 넣어서 나오는 결과를 D에 넣었을 때 나오는 결과를 log(1-결과)해서 얻는 기대값

학습 초반에는 생성자(G)가 생성하는 image가 D가 바로 구분할 수 있을 정도로 형편이 없어서 D(G(z))가 0에 가깝다. 학습이 진행될 수록 D가 fake data를 진짜로 판별하게 되어 D(G(z))가 1에 수렴할 것이다.

* D와 G의 입장에서 각각 살펴보면 D의 성능이 뛰어나면 D(x)는 1에 수렴하고 D(G(z))는 0에 수렴하므로 log1 + log(1-0) = 0으로 V(D, G) = 0이 된다. 즉, 가장 이상적인 D의 결과(최대값)는 0이다.
* G는 D(G(x))가 1이 되어야 진짜 같은 가짜를 만들어내는 것이고 D(x)는 D의 영역이기 때문에 log1 + log(1-1) = -무한대이다. 즉, 가장 이상적인 G의 결과(최솟값)은 -무한대이다.

따라서 D는 최대값을 위한, G는 최소값을 위한 방향으로 model이 학습되어야 한다. Two-player minmax game이라고 표현한다.

Model을 training시킬 때 D를 최적화하는 것은 많은 계산을 필요로 하고 적은 data로는 overfitting을 초래한다. 그렇기에 D는 k step만큼(많은 step으로 overfitting 방지), G는 1 step만큼 최적화한다. 학습 초반에는 log(1-D(G(z)))가 0에 수렴하므로 gradient가 너무 작기 때문에 log(D(G(z)))를 최대화되게끔 training시키는 것이 더 좋다.

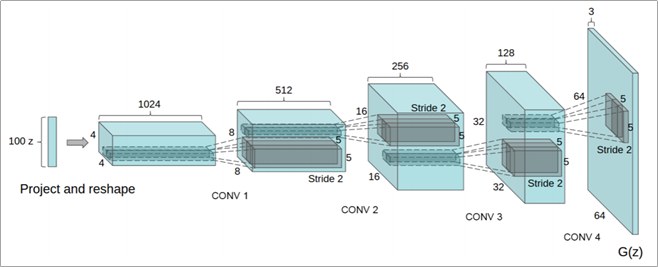


* 파란선: D / 검은색: data generation(real) / 녹색: generative(fake)

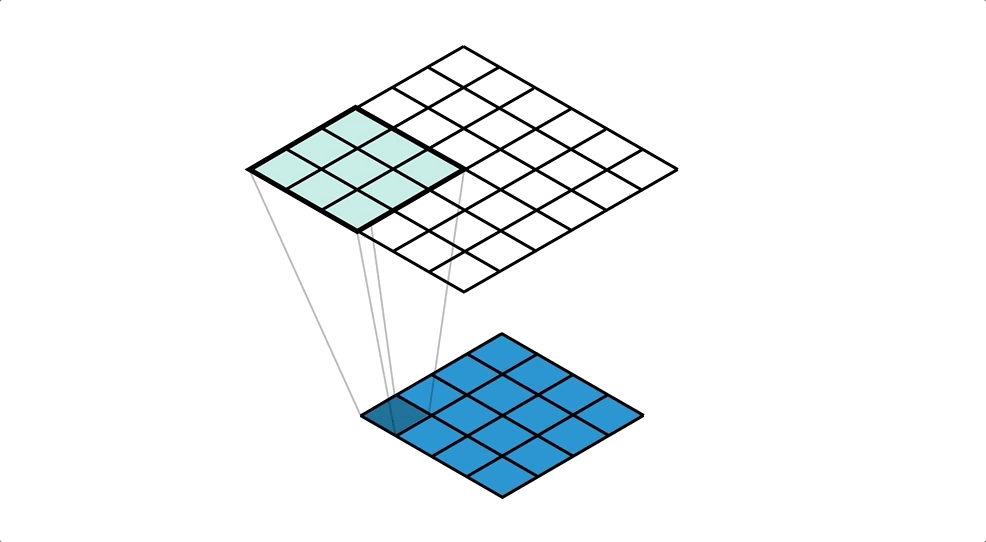
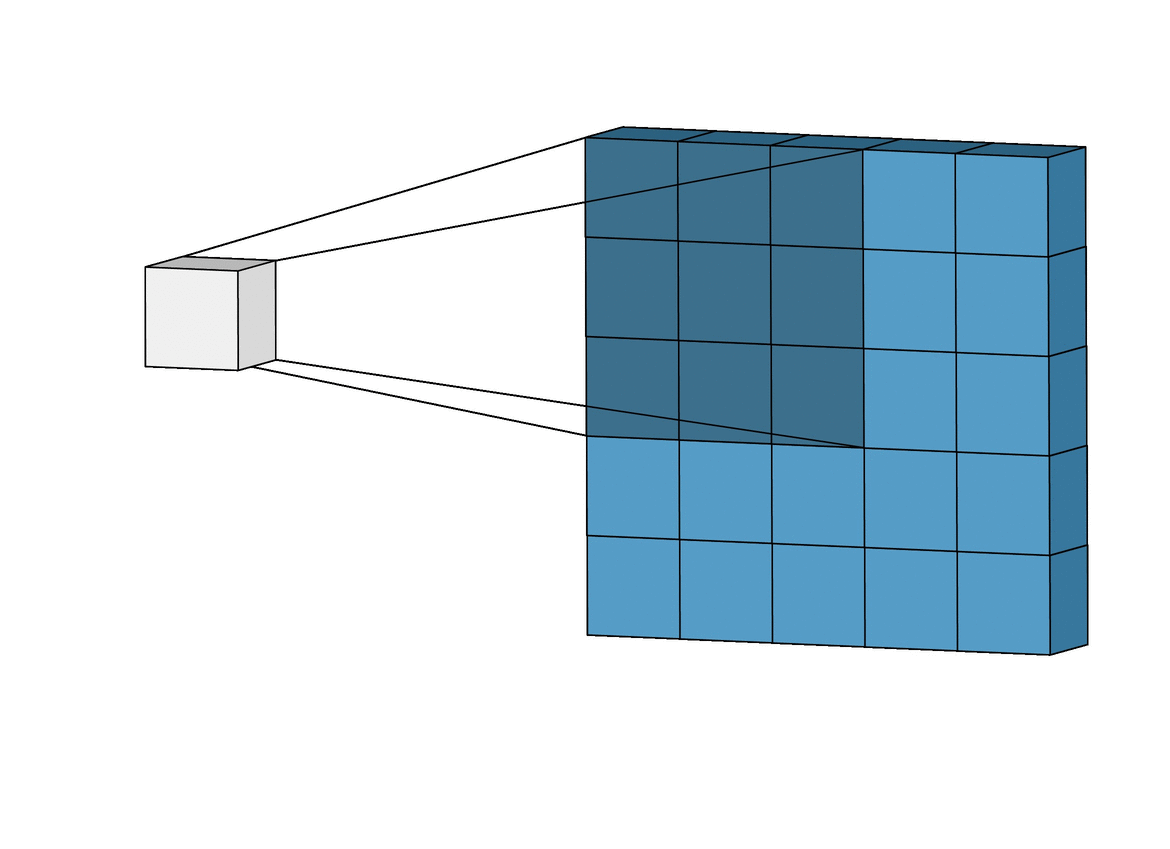
a에서 d로 갈 수록 real과 fake의 차이가 줄어들고 결국 둘이 수렴하여 D가 제대로 판별하지 못하게 되어 확률을 1/2로 계산하게 된다. 이러한 목적으로 training을 진행하는 것이 GAN의 최종 목적이라고 볼 수 있다.

DCGAN

기존 GAN은 Fully-Connected 구조로 성능이 잘 나오지 않았으나 이후 DCGAN(Deep Convolutional GAN)을 통해 성능을 향상시켰다.



G의 model에는 transposed convolutional network를 사용하여 up-sampling하는데 사용하였다. D의 model에 단순 convolutional network를 사용하였다. Transposed convolutional network의 경우 기존의 convolutional network처럼 줄어드는 것이 아닌 확대되는 것이다.



왼쪽이 기존의 CN. 기존 CN으로는 좋은 성능을 내지 못하여 5가지 방법을 고안하였다.

* 미분 불가한 pooling layer를 제거하고 미분 가능한 convolution layer로 대체함(upsampling시 매우 이상한 사진을 생성한다고 함)
* Batch normalization layer를 추가함
* Fully-connected hidden layer를 삭제함
* 생성자 모델에 ReLU 함수를 적용하고 출력의 activation function은 Tanh로 설정함
* D의 모든 layer에 LeakyReLU를 적용하여 유연성을 더함

LSGAN

D와 G의 loss function을 cross entropy에서 least square loss로 변경하였더니 더 좋은 성능을 보인다.

외에 PGGAN, 저화질 이미지를 고화질로 바꾸는 SRGAN, CycleGAN, StarGAN 등이 있다.