📖

[논문리뷰] GAN [2014]

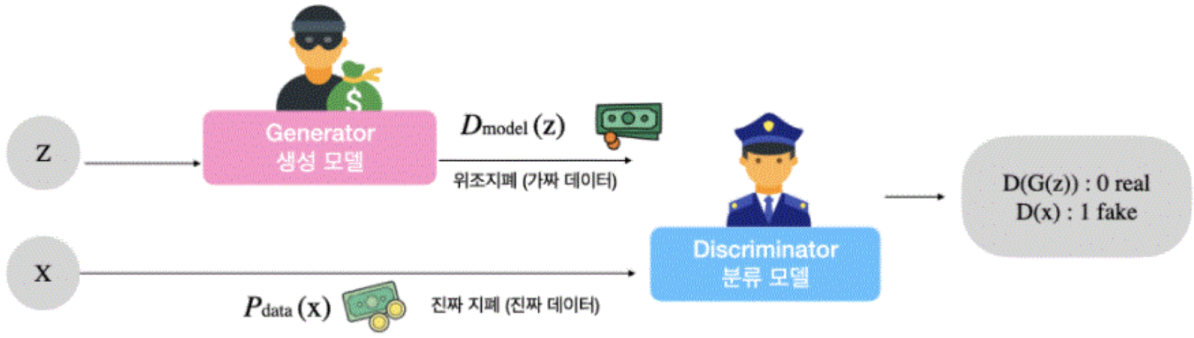
Introduction

🌟 해당 논문은 **적대적인(Adversarial) 과정**을 통하여,
평가하는 생성 모델(Genrative models)에 대한 새로운 Framework를 제시합니다.

두개의 모델을 동시에 활용합니다.

- Generative Model (G) : 데이터의 분포를 capture합니다.
- Discriminative Model (D) : 훈련 데이터로부터 온 데이터의 probability를 평가합니다.

> 💡 **Model G의 훈련 과정은 D가 실수할 확률을 최대화**하는 것입니다.
> **minimax two-player game**이라고 부르기도 합니다.
>
> 임의의 G,D에 의한 함수 공간에서 G는 훈련 데이터의 분포를 학습하며,
> 임의의 노이즈를 입력 받아 훈련 데이터와 같은 분포를 recovering하고, D는 해당 인풋이 생성된 이미지인지 훈련 데이터로부터 나온 이미지인지 평가합니다.



Generative Model (G)는 Discriminative Model (D)를 속이는 것이 가능하도록, 데이터의 분포를 학습하고, D는 G로 부터 나온 데이터와 진짜 데이터와 분류하는 방법을 학습합니다.
이러한 경쟁구도를 적대적(Adversarial)이라고 부릅니다. **[논문에서는 예시로 경찰 과 위조 지폐를 사용]**

Generator Model은 다층 퍼셉트론(Multilayer perceptrons)으로 구성되어 Random Noise를 전달하여, data를 생성합니다. 또한, Discriminator Model도 다층 퍼셉트론(Multilayer perceptrons)으로 구성됩니다.

이러한 구조를 적대적(Adversarial) 부릅니다. [강조]
G, D 모델들은 Backpropagation 과 Dropout 알고리즘으로부터 학습을 진행하며, G 모델은 오로지 forward propagation을 활용합니다.

이전의 생성기법인 Markov chains 기법들이 불필요합니다.

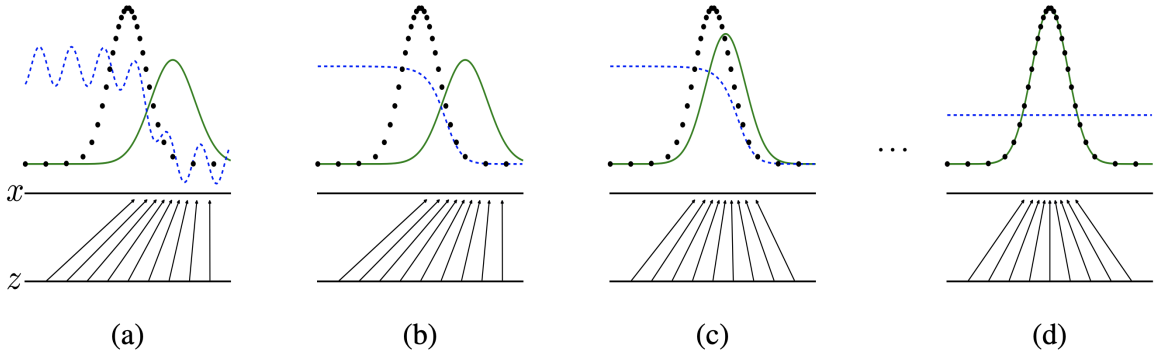
Adversarial Nets

Adversarial Modeling framework는 Model이 Multilayer Perceptrons일 때, 더 적용하기가 쉽다.

적대신경망은 모델이 모두 다층퍼셉트론일때 가장 적용하기 쉽다. 원본 x에 대해 생성된 p_g를 학습하기 위해 입력에서 더해질 noise 변수를 p_z(Z)라 정의하고 데이터 공간에 대한 매핑을 G(z; θ_g)로 표현하는데, 여기서 G는 매개 변수를 가진 다층 퍼셉트론에 의해 표현되는 미분 가능한 함수이다. 또한 단일 스칼라를 출력하는 두 번째 다층 퍼셉트론을 D(x; \(θ_d\))라고 정의한다. D(x)는 x가 p_g가 아닌 원본일 확률을 나타낸다. 우리는 D가 훈련 샘플과 G의 샘플 모두 올바르게 구별하도록 훈련시킨다. 동시에 log(1 – D(G(z)))를 최소화하도록 G를 훈련시킨다. 즉, D와 G는 가치 함수 V(G, D)를 가진 다음과 같은 2인용 미니맥스 게임을 한다.

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

🌟 D는 진짜 데이터가 들어올 경우, 1을 Output하며, 가짜 데이터를 출력하는 경우 0을 Output합니다.



학습 진행과정은은 위의 그림과 같습니다.

초록 선은 Generator 결과값의 분포이며, 검은 점들은 Dataset의 분포, 파란 점선은 Discriminator의 경계선을 나타냅니다.

(a) ~ (d)의 학습 진행과정을 보면, **처음에는 노이즈로부터 생성된 결과물들이 학습이 덜 된 G Model의 분포를 따르기에, 실제 데이터셋과 거리가 있다는 것을 알 수 있습니다.** 그러나, **학습이 진행될수록, G Model의 분포가 데이터셋의 분포를 따라가게 되며, 이상적인 결과 (d)에서는 Discriminator가 실제 데이터와 생성 데이터의 차이를 찾지 못하여, 어떠한 경우에도 반반의 확률을 나타내게 됩니다.**

Theoretical Results

G는 확률 분포 p_g를 z ~ p_z일 때 얻은 표본 G(z)의 분포로 암묵적으로 정의한다. 따라서, 우리는 충분한 양과 훈련 시간이 주어진다면 알고리즘 1이 p_{data}의 좋은 추정치로 수렴되기를 바란다. 이 섹션의 결과는 non-parametric하게 수행된다. 예를 들어 확률 밀도 함수의 공간에서의 수렴을 연구하여 무한 용량을 가진 모델을 나타낸다.

섹션 4.1에서 이 미니맥스 게임이 $p_g = p_{data}$에 대한 전역 최적점임을 보여줄 것이다. 섹션 4.2에서 알고리즘 1이 방정식 1을 최적화하여 원하는 결과를 얻는다는 것을 보여줄 것이다.

Algorithm 1 :

적대신경망의 훈련을 위한 미니배치 확률적 경사 하강법. 하이퍼파라미터 k는 구별모델에 적용하기 위한 스텝을 나타낸다. 우리는 실험에서 가장 저렴한 k=1을 사용하였다.

---

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

---

1. noise prior $p_g(z)$ 로부터 noise 시킨 m개의 미니배치 $\{z^{(1)}, \ldots, z^{(m)}\}$ 샘플을 만든다.
2. 데이터 생성분포 $p_{data}(x)$ 로부터 미니배치 m개의 샘플 $\{x^{(1)}, \ldots, x^{(m)}\}$을 평가한다.
3. 확률적 경사 상승법을 이용하여 D를 업데이트한다.

k번의 스텝이후

k-1. noise prior $p_g(z)$ 로부터 noise 시킨 m개의 미니배치 $\{z^{(1)}, \ldots, z^{(m)}\}$ 샘플을 만든다.

k-2. 확률적 경사 하강법을 이용하여 G를 업데이트한다.

우리의 실험은 momentum을 이용한 옵티마이저를 사용했다.

4.1 Global Optimality of $p_g = p_{data}$

우선 주어진 G에 대해 최적의 D를 고려해보자.

Proposition 1. G가 고정 되어 있을 때 최적의 D는 아래와 같다.

$$D_G^*(\boldsymbol{x}) = \frac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$$

증명 : 주어진 어떤 G에 대해 D의 훈련법은 V(G,D)의 양을 최대화 시키는 것이다.

$$\begin{aligned} V(G,D) &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x}))dx + \int_{z} p_{\boldsymbol{z}}(\boldsymbol{z}) \log(1 - D(g(\boldsymbol{z})))dz \\ &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x})) + p_g(\boldsymbol{x}) \log(1 - D(\boldsymbol{x}))dx \end{aligned}$$

임의의 $(a, b) \in R^2 \setminus \{0,0\}$ 에 대하여 함수 y → a log(y) + b log(1 - y) 는 $\frac{a}{a+b}$로 [0,1]에서 최대치를 달성한다. D를 $Supp(p_{data}) \cup Supp(p_g)$ 외부에서 정의할 필요가 없으며 증명은 마무리된다.

D에 대한 훈련 목표는 조건부 확률 P(Y = y|x)를 추정하기 위한 로그 우도를 최대화하는 것으로 해석될 수 있다. 여기서 Y는 x가 p_{data} (with y = 1) 또는 p_g(with y = 0)로부터 오는지를 나타낸다. 방정식 1에서의 minimax 게임을 다음과 같이 재구성할 수 있다.

$$\begin{aligned} C(G) &= \max_D V(G,D) \\ &= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}}[\log(1 - D_G^*(G(\boldsymbol{z})))] \\ &= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))] \\ &= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x} \sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] \end{aligned}$$

Theorem 1. 가상 훈련 기준 C(G)의 전역 최소값은 $p_g = p_{data}$인 경우에만 달성된다. 이 시점에서 C(G)는 다음과 같은 값을 달성합니다. - log 4.

증명 : (방정식 2에 따르면) $p_g = p_{data}$ 에 대해 $D^*_G(x) = \frac{1}{2}$ 이다. 따라서 $D^*_G(x) = \frac{1}{2}$에서의 방정식 4를 보면 C(G) = $\log\frac{1}{2}$ + $\log\frac{1}{2}$ = -log4 인것을 확인할 수 있다. 이 식이 $p_g = p_{data}$에 대해 최적의 C(G)값인지는 다음 식을 확인하시오.

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}[-\log 2] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[-\log 2] = -\log 4$$

그리고 이 식을 C(G) = V(D^*_G, G)에서 빼면 다음과 같은 값을 얻을 수 있다.

$$C(G) = -\log(4) + KL\left(p_{\text{data}} \middle\| \frac{p_{\text{data}} + p_g}{2}\right) + KL\left(p_g \middle\| \frac{p_{\text{data}} + p_g}{2}\right)$$

KL은 Kullback-Leibler 발산이다. 우리는 이전 표현에서 모델의 분포와 데이터 생성과정에서 Jensen-Shannon발산을 발견했다.

$$C(G) = -\log(4) + 2 \cdot JSD\left(p_{\text{data}} \| p_g\right)$$

두 분포 사이의 Jensen-Shannon 발산은 항상 음이 아니고 그것들이 같을 때만 0이기 때문에, 우리는 C^* = -log(4)가 C(G)의 전역 최소값이며, 유일한 값은 $p_g = p_{data}$이다. 이는 생성모델이 완벽하게 데이터를 생성했다고 본다.

4.2 Convergence of Algorithm 1

Proposition 2. G와 D가 충분한 능력을 가지고 있고 알고리즘 1의 각 단계에서 D는 최적의 G에 도달할 수 있으며 p_g는 정책을 개선하기 위해 갱신된다.

$$\mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D^*_G(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D^*_G(\boldsymbol{x}))]$$

그리고 p_g는 p_{data}로 수렴한다.

증명 : 위에서 쓰인 기준과 같이 V(G, D) = U(p_g, D)를 p_g의 함수로 간주한다. U(p_g, D)는 p_g 에서 convex 하다. 볼록함수 우위의 하위 도함수는 최대치에 도달하는 지점에서 함수의 도함수를 포함한다. 이말은, 만약 f(x)=$sup_{\alpha\in\mathcal{A}} f_\alpha(x)$ 이고 모든 α 에 대해 $f_\alpha(x)$ 가 x 에 convex 하면 $\beta = arg sup_{\alpha\in\mathcal{A}} f\alpha(x)$ 일때 $\partial f_\beta(x) \in \partial f$ 이다.
이는 해당 G가 주어진 최적 D에서 p_g에 대한 경사 하강법을 계산하는 것과 같다. p_g 에 대해 $sup_D U(p_g, D)$ 가 유일한 전역최적점을 가지며 convex 하는것이 Thm1 에서 증명되었기 때문에 p_g 는 충분히 작게 갱신되고, p_g 가 p_x 로 수렴하므로 증명을 마무리한다.

실제로, 적대신경망은 함수 G(z; θ_g)를 통해 p_g 분포의 제한된 부분을 나타내며, 우리는 p_g 자체보다는 θ_g를 최적화한다. 다층 퍼셉트론을 사용하여 G를 정의하면 파라미터 공간에 여러 임계점이 도입된다. 그러나 실제로 다층 퍼셉트론의 뛰어난 성능은 이론적인 보장이 없음에도 불구하고 사용하기에 합리적인 모델임을 보여준다.

Experiments

MNIST, TFD, CIFAR Dataset을 이용하여, Adversarial nets을 훈련시켰습니다.
G Model의 경우, ractifier linear, sigmoid 활성화 함수를 활용하였으며,
D Model의 경우, maxout 활성화 함수와 Dropout을 사용하였습니다.

G로 생성된 **sample에 Gaussian Parzen Window를 fitting**하고,
log-distributtion으로부터 testdata의 확률을 추정하였습니다.

Gaussian의 σ parameter의 경우, cross validation을 통하여, 얻었습니다.

Model	MNIST	TFD
DBN [3]	138 ± 2	1909 ± 66
Stacked CAE [3]	121 ± 1.6	$\mathbf{2110 \pm 50}$
Deep GSN [5]	214 ± 1.1	1890 ± 29
Adversarial nets	$\mathbf{225 \pm 2}$	$\mathbf{2057 \pm 26}$

Table 1: Parzen window-based log-likelihood estimates. The reported numbers on MNIST are the mean log-likelihood of samples on test set, with the standard error of the mean computed across examples. On TFD, we computed the standard error across folds of the dataset, with a different σ chosen using the validation set of each fold. On TFD, σ was cross validated on each fold and mean log-likelihood on each fold were computed. For MNIST we compare against other models of the real-valued (rather than binary) version of dataset.

확률을 추정하는 해당 방식은 high variance을가지고 있으며, high dimensional space에서 잘 수행되지 않지만, It's the best method available to our knowledge.

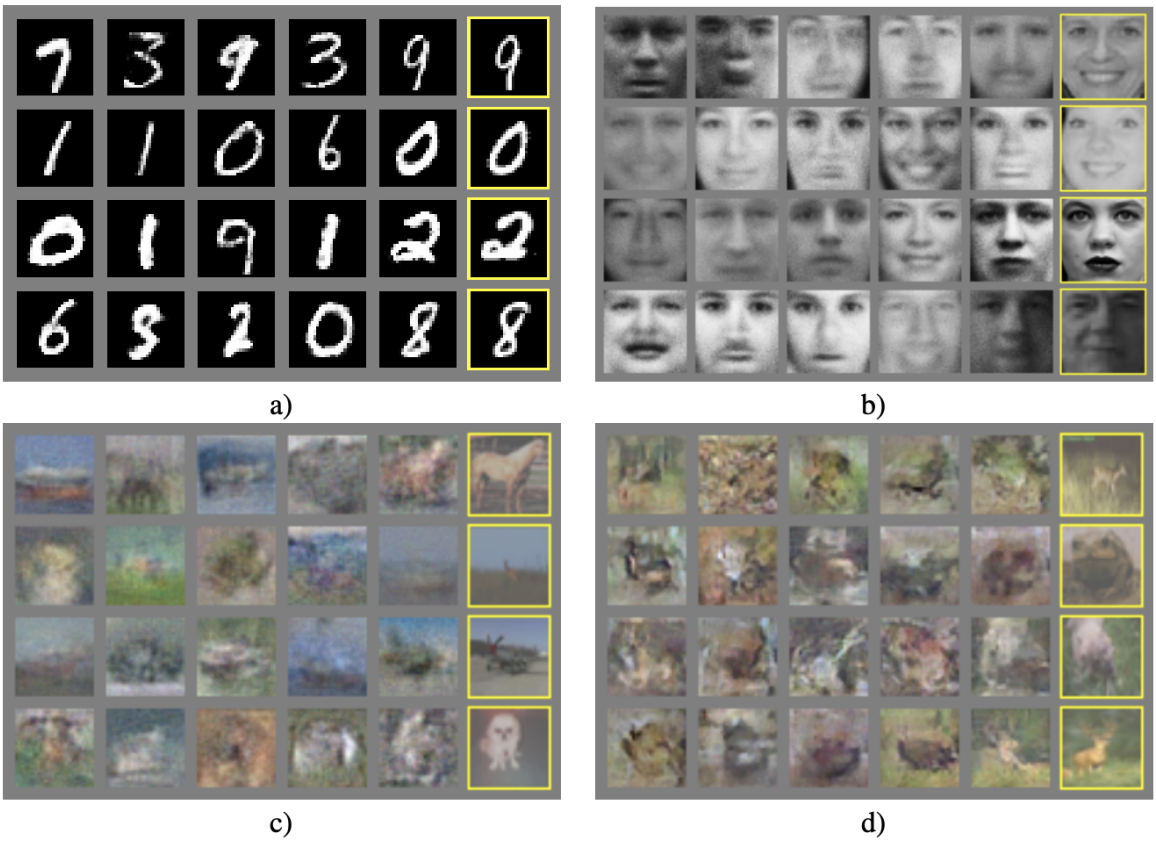


Figure 2: Visualization of samples from the model. Rightmost column shows the nearest training example of the neighboring sample, in order to demonstrate that the model has not memorized the training set. Samples are fair random draws, not cherry-picked. Unlike most other visualizations of deep generative models, these images show actual samples from the model distributions, not conditional means given samples of hidden units. Moreover, these samples are uncorrelated because the sampling process does not depend on Markov chain mixing. a) MNIST b) TFD c) CIFAR-10 (fully connected model) d) CIFAR-10 (convolutional discriminator and "deconvolutional" generator)

위의 그림과 같이, G Model을 학습한 후 출력된 이미지가 있습니다.
Adversarial Net의 Sample이 기존의 방법에 의해 생성된 Sample보다 낫다고 주장하지는 않지만,
이전의 Model과 경쟁하고 있으며, **Adversarial Net의 가능성을 강조**합니다.

Advantages and disadvantages

Adeversarial Network는 기존의 framework와 다르게 상대적으로 장점과 단점이 존재합니다.

먼저, **단점으로는 일차적으로 G Model이 생성한 data의 확률에 대한 명시적 표현이 없으며**,
훈련 과정 중에서 D, G 두 가지 모델을 동기화해야 한다는 것입니다.

장점으로는 주로 Computational 측면에 있습니다. D의 기울기 조정만으로도 G에서의 이점을 얻을 수 있습니다. 또한 Backpropagation을 사용하며, 다양한 기능의 모델이 통합될 수 있다는 것입니다.

Conclusions and future work

Adversarial은 간단한 확장을 허용합니다.

- G, D Model에 입력 c를 더하면, 조건부 생성 모델 P(X | C)를 얻을 수 있습니다.
- 학습된 근사 추론은 주어진 x를 예측하기 위해, 보조 네트워크를 훈련함으로써 수행할 수 있습니다.
  이것은 wake-sleep 알고리즘에 의해 훈련되는 inferene net과 유사하지만,
  inference net이 훈련된다는 장점이 있습니다.
- Semi Supervised Learning :
  inference Net 혹은 D의 feature는 제한된 레이블 데이터를 사용하는 것이 가능할 때,
  분류기의 성능을 향상시키는 것이 가능합니다.
- Efficiency improvements :
  훈련 중에 G와 D를 조정하는 더 나은 방법을 찾거나 z 표본을 추출하는 더 나은 분포를 결정함으로써, 훈련을 크게 가속화할 수 있습니다.

Reference

Generative Adversarial Nets

Generative Adversarial Nets Part of Advances in Neural Information Processing Systems 27 (NIPS 2014) Authors Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio Abstract We propose a new framework for estimating generative models via adversarial nets, in which we

https://papers.nips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html

https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

GAN (Generative Adversarial Nets) 생성적 적대 신경망 논문 리뷰/번역 - 2014

우리는 2가지 모델을 훈련시켜 적대적인 프로세스를 통해 생성모델을 평가하는 새로운 프레임워크를 제시한다 : 데이터를 제공하는 생성모델 G와 해당 데이터가 G가 만든것인지, 원본인지 확률을 평가하는 구별모델 D이다. G는 D가 실수할 확률을 높이는 방향으로 학습된다.

https://bestkcs1234.tistory.com/53