GAN

2024-10-22

박준형

GAN?

Generative Adversarial Nets

Ian J. Goodfellow, Jean Pouget-Abadie[*], Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair[†], Aaron Courville, Yoshua Bengio[‡]
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

Abstract

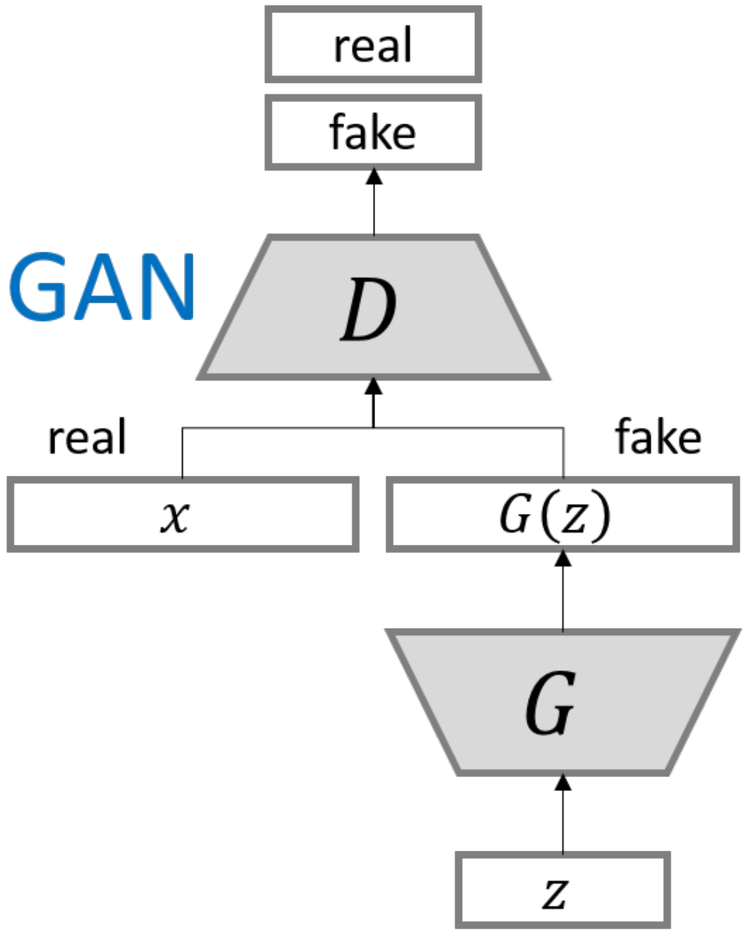
We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to $\frac{1}{2}$ everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples. Experiments demonstrate the potential of the framework through qualitative and quantitative evaluation of the generated samples.

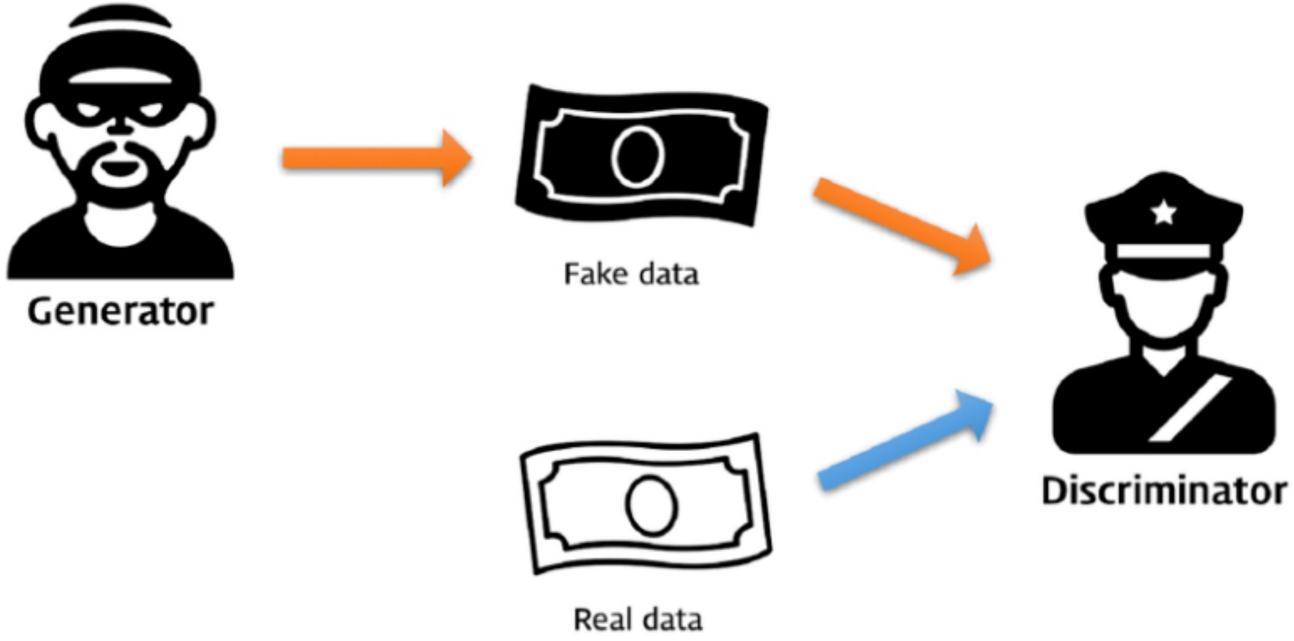
GAN

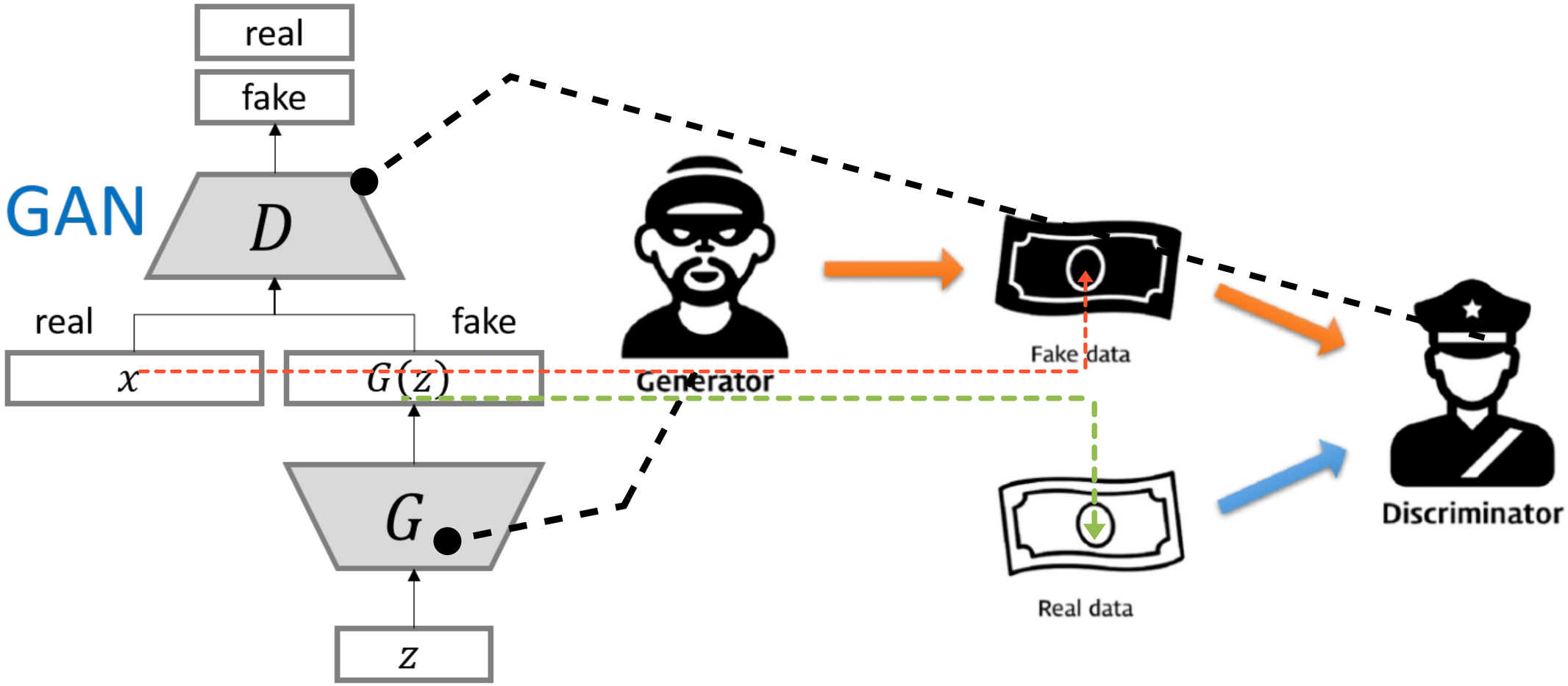
Generative

Adversarial

Networks


real
fake
GAN
D
real
fake
x
$G(z)$
G
z


Generator
Fake data
Real data
Discriminator


real
fake
GAN
D
real
fake
x
G(z)
G
z
Generator
Fake data
Real data
Discriminator

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

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

real data x를 discriminator 에 넣었을 때 나오는 결과를
log취했을 때 얻는 기댓값

fake data z를 generator에 넣었을 때 나오는 결과를 discriminator에 넣었을 때
그 결과를 log(1-결과)했을 때 얻는 기댓값

D and G play the following two-player minimax game

학습 과정

데이터 준비

학습데이터 (진짜 데이터)준비.
생성자는 랜덤 노이즈 벡터를 입력 받음

판별자 학습 (D 업데이트)

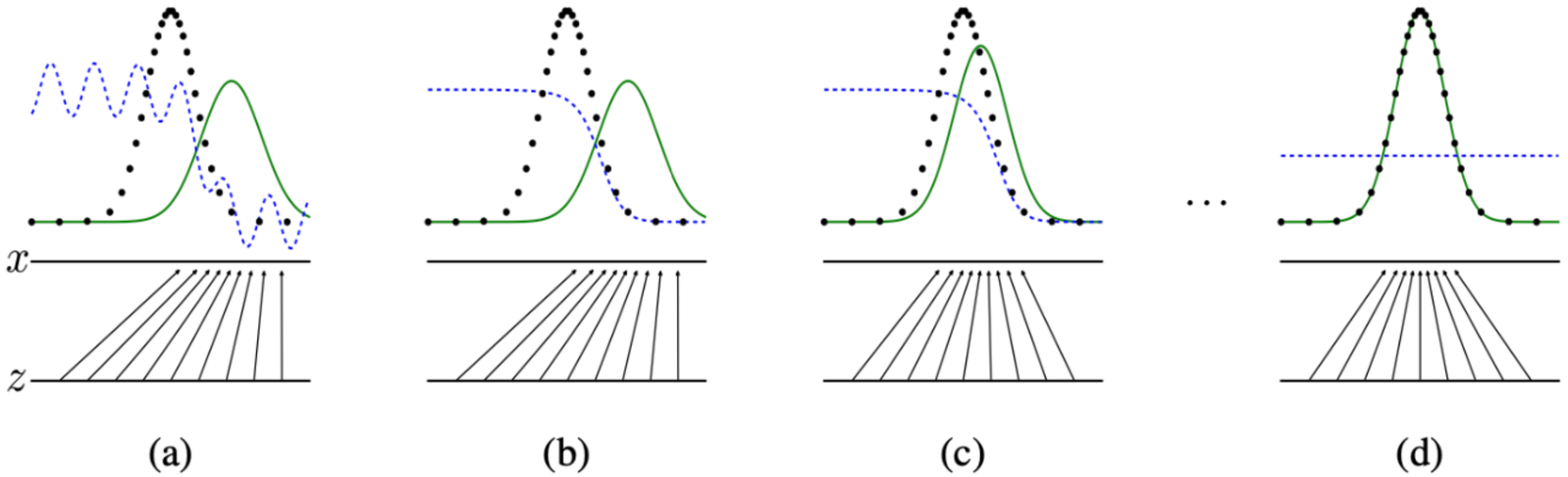
판별자는 이 실제 데이터와 가짜 데이터를 정확하게 구분하도록 학습.
판별자의 손실을 최소화하기 위해 파라미터를 업데이트.

생성자 학습 (G 업데이트)

생성자가 만들어낸 가짜 데이터를 판별자가 실제 데이터라고 판단하게끔 학습.
생성자는 판별자의 파라미터는 고정한 채 자신의 파라미터만 업데이트하여 학습

반복

이상적으로는, 생성자가 만든 가짜 데이터가 실제 데이터와 매우 유사해져 판별자가 더 이상 구분할 수 없게 되는 상태로 수렴



파란 점선: 판별자가 예측한 진짜 데이터일 확률. 값이 높을수록 진짜로 판단하는 확률이 큼.

검정색 선: 원본(진짜) 데이터

초록색 선: 생성된(가짜) 데이터

x 축: 데이터 공간. 실제 데이터나 생성된 데이터가 위치하는 공간

z 축: 노이즈 공간을 의미. 이 노이즈는 생성자G의 입력으로 사용

화살표: x=G(z)mapping

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

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:

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

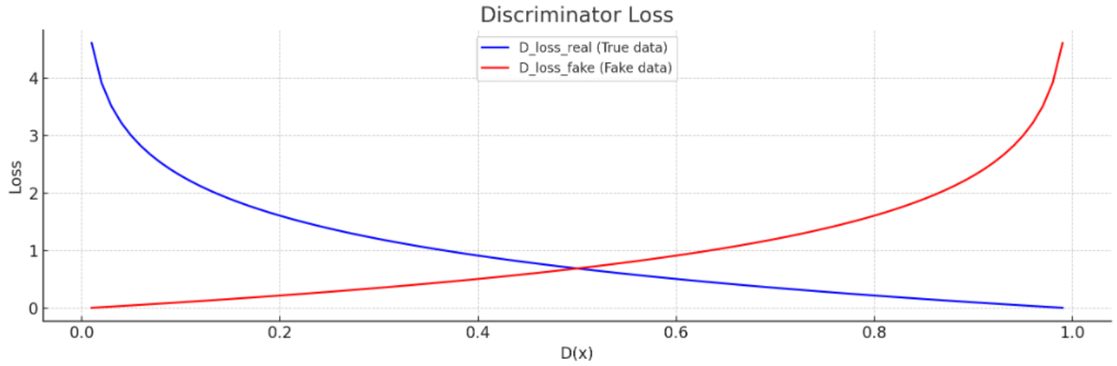
end for

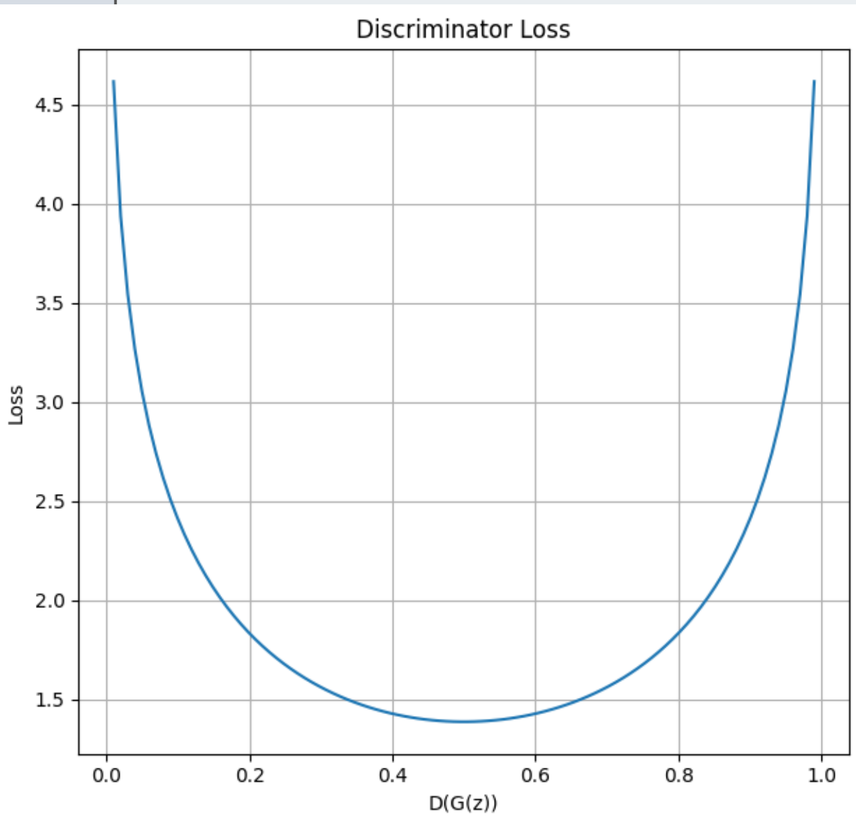
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

D의 파라미터에 대한 기울기

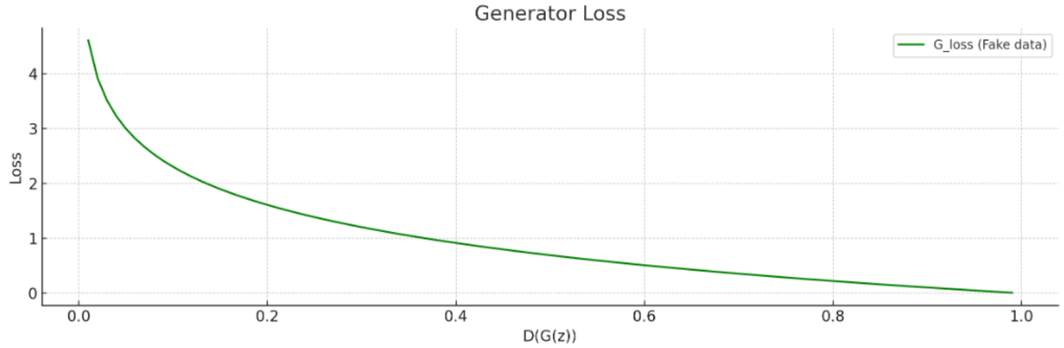
미니 배치 크기.
한 번의 업데이트에 사용되는 데이터 샘플의 수

$$L_D = -[\log D(x) + \log(1 - D(G(z)))]$$





$$L_G = -\log D(G(z))$$

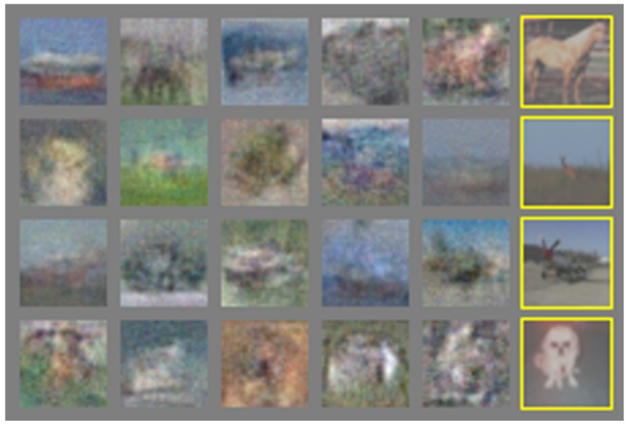


MNIST

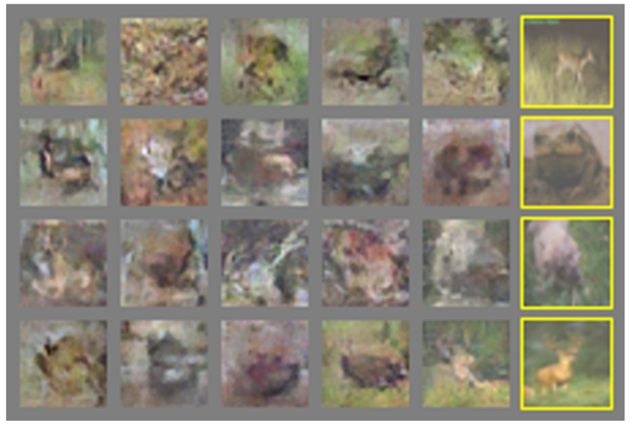
a)

Toronto Face Database

b)

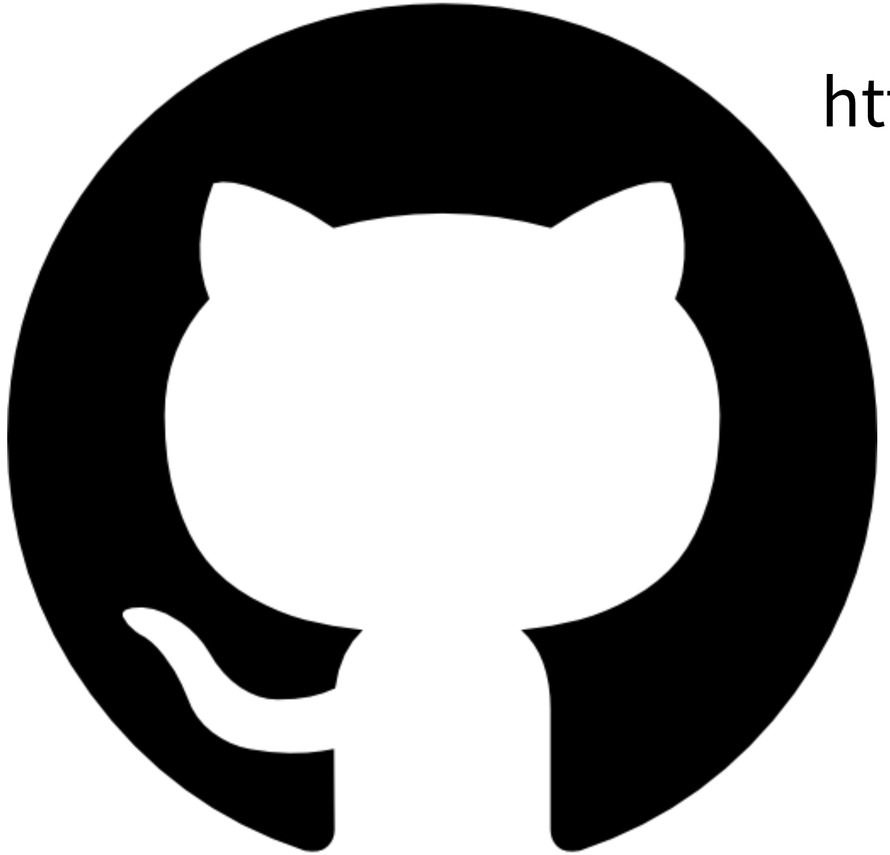


c)



d)

CIFAR-10



https://github.com/eriklindernoren/PyTorch-GAN

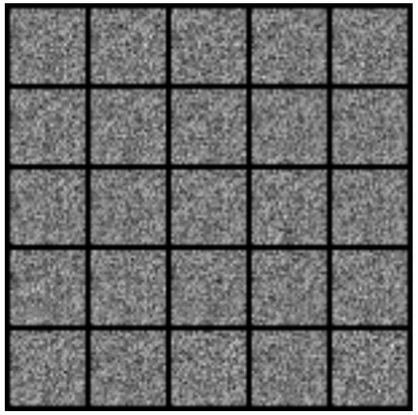
- Implementations
  - Auxiliary Classifier GAN
  - Adversarial Autoencoder
  - BEGAN
  - BicycleGAN
  - Boundary-Seeking GAN
  - Cluster GAN
  - Conditional GAN
  - Context-Conditional GAN
  - Context Encoder
  - Coupled GAN
  - CycleGAN
  - Deep Convolutional GAN
  - DiscoGAN
  - DRAGAN
  - DualGAN
  - Energy-Based GAN
  - Enhanced Super-Resolution GAN
  - GAN
  - InfoGAN
  - Least Squares GAN
  - MUNIT
  - Pix2Pix
  - PixelDA
  - Relativistic GAN
  - Semi-Supervised GAN
  - Softmax GAN
  - StarGAN
  - Super-Resolution GAN
  - UNIT
  - Wasserstein GAN
  - Wasserstein GAN GP
  - Wasserstein GAN DIV

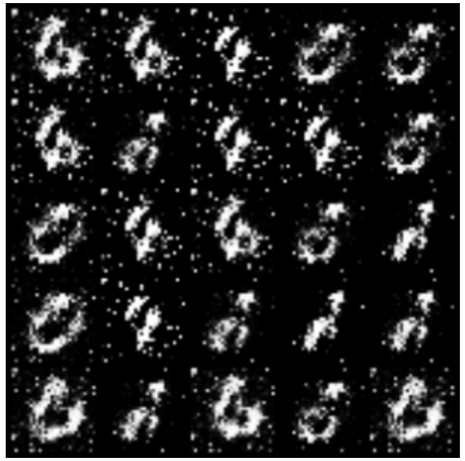
Installation

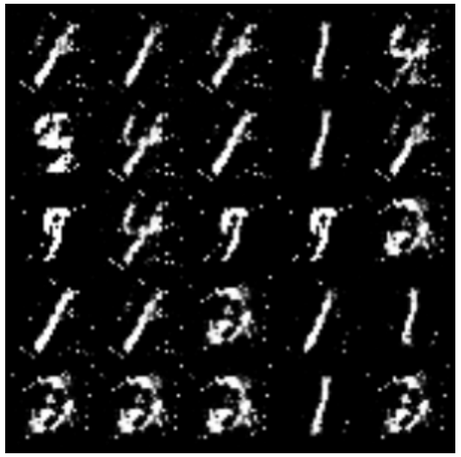
```
$ git clone https://github.com/eriklindernoren/PyTorch-GAN
$ cd PyTorch-GAN/
$ sudo pip3 install -r requirements.txt
```

Run Example

```
$ cd implementations/gan/
$ python3 gan.py
```







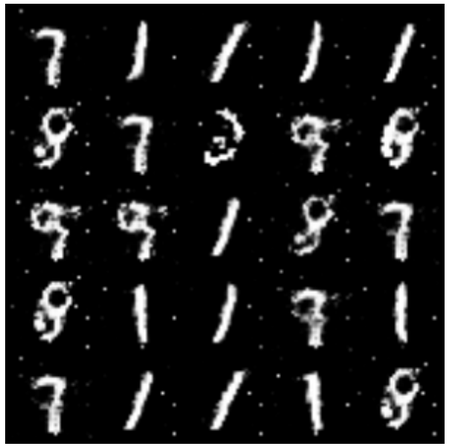
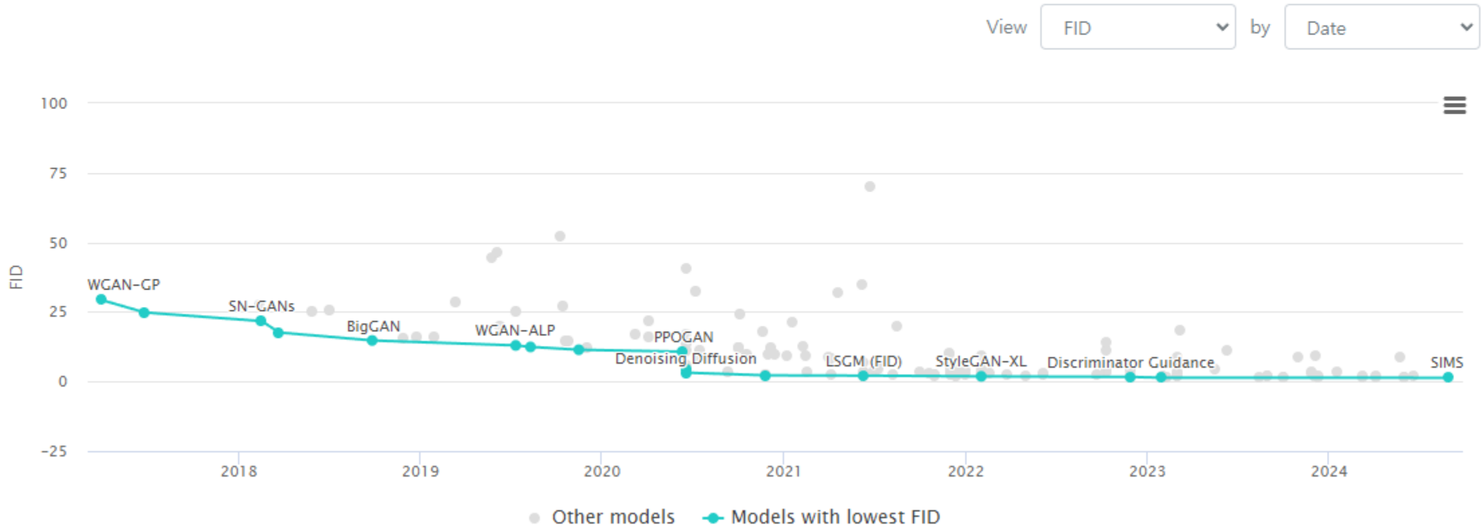


Image Generation on CIFAR-10

Leaderboard Dataset



Rank	Model	Paper	Code	Result	Year	Tags
1	**SIMS**	Self-Improving Diffusion Models with Synthetic Data			2024	Diffusion
2	**StyleSAN-XL**	SAN: Inducing Metrizability of GAN with Discriminative Normalized Linear Layer			2023	GAN
3	**PFGM++ + CS**	Compensation Sampling for Improved Convergence in Diffusion Models			2023	PFGM, Diffusion
4	**GDD-I**	Diffusion Models Are Innate One-Step Generators			2024	Diffusion, GAN
5	**CTM**	Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion			2023	Diffusion

Here are the classes in the dataset, as well as 10 random images from each: