



PyTorch로 가짜 이미지 생성하기

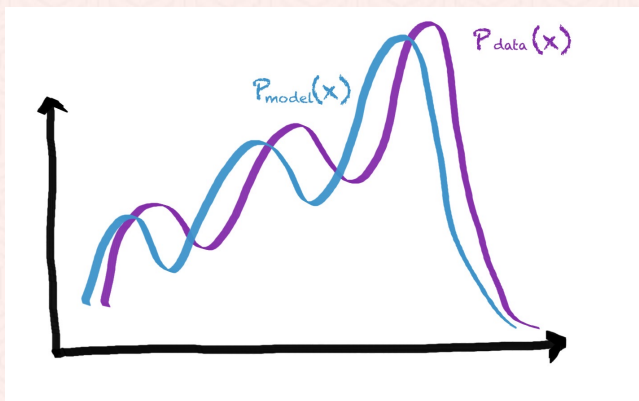
- GAN

오늘의 학습내용

- 생성모델(Generative Model)
- 잠재 변수(Latent Variable)
- Generative Adversarial Networks(GAN)
- PyTorch를 이용한 GAN 구현

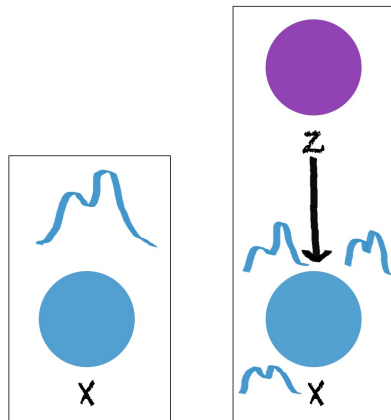
◆ 생성 모델(Generative Model)의 개념

- ❖ 지금까지 주로 살펴본 모델들은 지도 학습 방법론으로 어떤 값을 분류하거나 예측하는데 초점이 맞추어져 있었습니다. 이런 모델을 **구분 모델** Discriminative Model 이라고 합니다. 이번 시간에 배울 **생성 모델** Generative Model 은 조금 다른 목적을 가지고 있습니다. 생성 모델은 주어진 트레이닝 데이터의 특징을 학습하여 트레이닝 데이터와 유사한 새로운 데이터를 생성 Generate 하는데 그 목적이 있습니다.
- ❖ 이런 과정을 좀 더 수학적인 용어로 표현하면 생성 모델의 **목적은 트레이닝 데이터의 분포를 학습**하여 트레이닝 데이터의 분포와 유사한 데이터를 샘플링을 통해 새로 생성하는 것입니다. 그림은 생성 모델의 학습 과정을 보여줍니다. 생성 모델은 트레이닝 데이터의 분포인 $p_{data}(x)$ 를 통해 유사한 분포인 $p_{model}(x)$ 를 학습합니다.



◆ 잠재 변수(Latent Variable)

- ❖ 생성 모델의 개념을 자세히 이해하기 위해서는 몇 가지 확률 통계학의 개념들을 이해해야만 합니다. 먼저 **잠재 변수** Latent Variable 라는 개념을 살펴봅시다.
- ❖ 잠재 변수는 이름에서 알 수 있듯이 숨겨진 변수로써 데이터에 직접적으로 나타나지 않지만 현재 **데이터 분포를 만드는데 영향을 끼치는 변수**입니다. 따라서 어떤 데이터의 잠재 변수를 알아내면 잠재 변수를 이용해서 해당 데이터와 유사한 데이터를 생성해낼 수 있습니다.
- ❖ 즉, 잠재 변수는 데이터의 형태를 결정하는 특징으로 생각할 수 있습니다. 잠재 변수는 보통 잠재 변수로부터 생성하는 데이터보다 적은 차원을 갖습니다. 그림은 우리가 가지고 있는 데이터 x 와 잠재 변수 z 로부터 생성된 다양한 데이터 x 를 보여줍니다.



◆ 생성 모델(Generative Model)의 개념

- ❖ 예를 들어서, 우리가 학습하고 생성하고자 하는 데이터가 **사람 얼굴 이미지**라면 적절한 잠재 변수는 사람의 성별이 될 수 있습니다. 잠재 변수를 **사람의 성별**로 간주할 경우, 이 사람이 남자인지 여자인지를 나타내는 **1차원 Boolean 특징값**만을 가지고도 생성하고자 하는 데이터의 형태(예를 들어, 남자라면 짧은 머리의 얼굴, 여자라면 긴머리의 얼굴)를 어느 정도 결정할 수 있습니다. 이에 더해서 **사람의 표정, 촬영한 카메라의 각도** 등이 적절한 잠재 변수가 될 수 있을 것입니다.
- ❖ 다른 예로 우리가 학습하고 생성하고자하는 데이터가 **MNIST 필기체 데이터**라면 **필기획의 기울기, 이미지를 나타내는 레이블(1,2,3,...)** 등이 적절한 잠재 변수가 될 수 있습니다.
- ❖ 다음 장에서 배우는 GAN은 **임의의(노이즈의) 잠재 변수로부터 적절한 데이터를 생성해내는 함수**를 학습합니다. 즉, GAN의 동작 과정을 수식으로 나타내면 아래와 같습니다.

$$x_{data} = f_{GAN}(z_{noise})$$

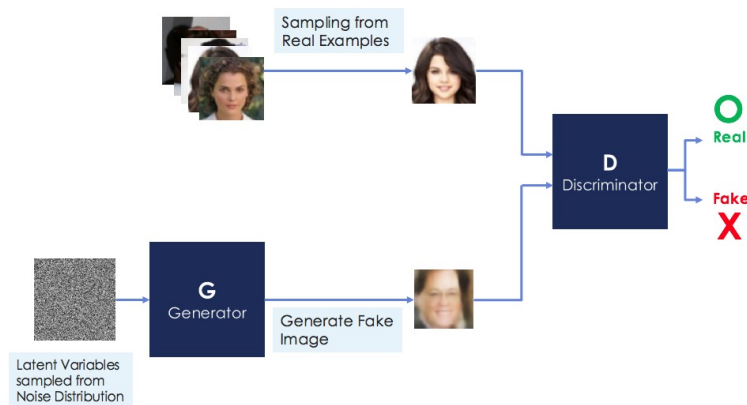
◆ GAN(Generative Adversarial Networks)의 개념

- ❖ **GAN** Generative Adversarial Networks은 게임 이론 Game Theory의 minimax two-player 게임의 구조를 이용해서 생성 모델을 구현한 구조입니다. GAN 모델은 Goodfellow, Ian, et al의 "Generative adversarial nets."라는 제목의 2014년도 논문에서 최초로 제안되었습니다.
- ❖ 구체적으로 GAN은 **생성자** Generator와 **구분자** Discriminator라는 2가지 부분으로 구성되어 있습니다.
- ❖ GAN의 개념을 직관적으로 이해하기 위해서 많이 사용하는 예시는 **경찰(구분자)**과 **위조지폐 생성범(생성자)**입니다. 위조지폐 생성범은 경찰을 속이기 위해서 최대한 진짜 지폐와 구분이 되지않는 위조지폐를 생성하려고 노력할 것입니다. 이에 반해 경찰은 위조지폐 생성범이 생성한 위조지폐와 진짜 지폐를 최대한 정확하게 구분할 수 있도록 노력할 것입니다.
- ❖ 경찰과 위조지폐 생성범이 서로 노력해서 계속 학습을 진행하면 경찰이 위조지폐 생성범이 생성한 위조지폐와 진짜 지폐를 50% 확률로 구분할 수 있게 되는 균형점에서 학습이 종료됩니다. 결과적으로 **생성자는 원본 데이터와 유사한 데이터 분포를 학습**하게됩니다.

GAN(Generative Adversarial Networks)의 구조

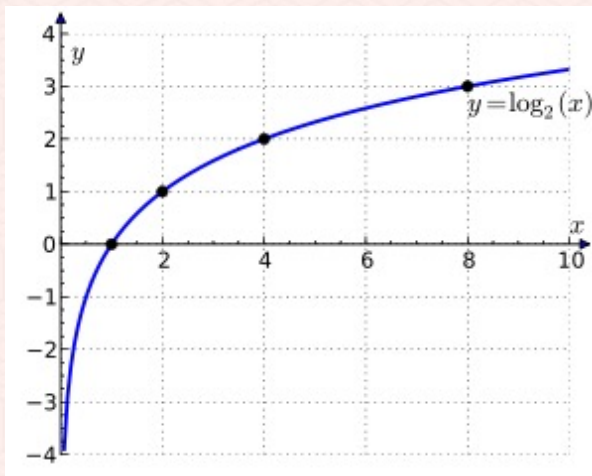
- ❖ 사람 얼굴 이미지를 생성하는 GAN을 구성할 경우, 생성자 G는 임의의 잠재 변수(노이즈값)을 입력 받아서 가짜 이미지(Fake Image)를 생성합니다.
- ❖ 구분자 D는 진짜 사람 얼굴 이미지인 진짜 이미지와 가짜 이미지를 입력 받을 수 있고, 만약 입력 받은 이미지가 진짜 이미지이면 1, 가짜 이미지이면 0의 값을 출력하는 것을 목표로 합니다.
- ❖ 생성자 G와 구분자 D는 임의의 머신 러닝 모델(예를 들면, 소프트맥스 회귀, SVM 등)을 사용하여 구현할 수 있지만 일반적으로는 인공신경망을 이용해서 구현합니다.

Generative Adversarial Networks(GAN)



◆ GAN의 Loss Function

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



GAN Learning Algorithm

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

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

end for

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

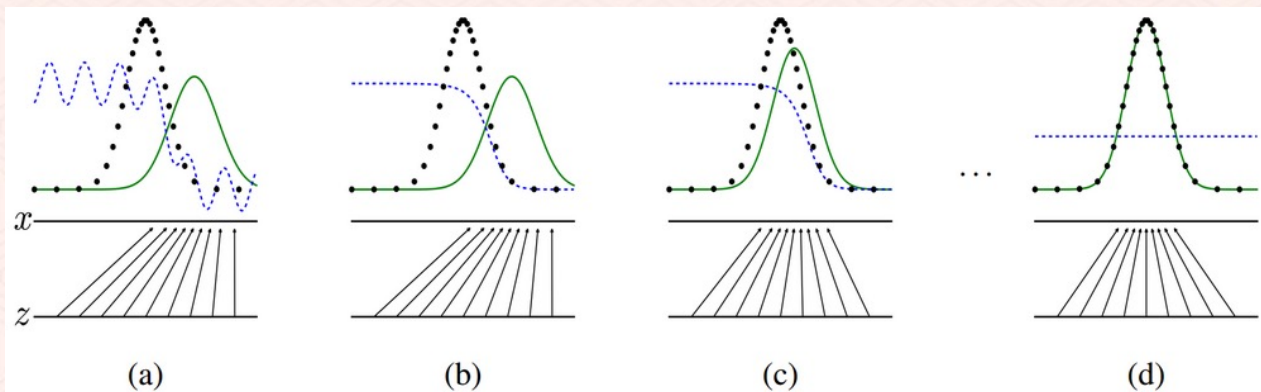
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D(G(z^{(i)})) \right).$$

end for

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

GAN(Generative Adversarial Networks)의 동작과정

- ❖ 그림은 GAN의 학습 과정을 단계별로 나타냅니다.
- ❖ 학습 초기에는 트레이닝 데이터의 분포(검은색 점)와 다른 형태로 생성자가 분포를 생성합니다. 또한 구분자(파란색 라인)는 임의의 확률로 트레이닝 데이터와 생성한 데이터를 구분합니다.
- ❖ 그리고 **잠재 변수 z 의 값은 임의의 데이터 값 x 로 맵핑**됩니다. 하지만 학습이 진행됨에 따라 생성자가 생성한 분포(초록색 라인)는 트레이닝 데이터의 분포와 점점 유사해지다가 **학습이 완료되면 트레이닝 데이터의 분포와 일치**하게 됩니다.
- ❖ 또한 구분자는 50% 확률로 트레이닝 데이터와 생성자가 생성한 가짜 데이터를 구분하게 되고, 잠재 변수 z 의 값은 트레이닝 데이터의 분포값 x 로 맵핑되게 됩니다.



▲ 파이토치(PyTorch)를 이용해서 가짜 MNIST 이미지 만들기

- ❖ 파이토치(PyTorch)를 이용해서 가짜 MNIST 이미지를 만들어봅시다.
- ❖ 8강_PyTorch를_이용해서_가짜이미지_만들기_gan.ipynb
- ❖ https://colab.research.google.com/drive/19sxhURLJVDqirX2ueroo6sHq4ZcX6_co?usp=sharing

