GANs 리뷰

1. 생성모델

1. GAN은 그럴 싸한 데이터를 만들 수 있는 생성 모델의 한 종류
2. 이미지 데이터에 대한 확률 분포

   이미지데이터도 벡터나 행렬과 같은 데이터 형태로서 컴퓨터가 가지고 있을 수 있으므로 컴퓨터로 모델링 할 수 있다.

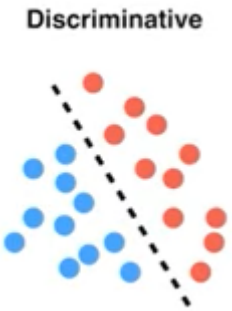
   이미지에서의 다양한 특징들이 각각의 확률 변수가 되는 분포를 의미한다.

   1. 이미지 데이터는 다차원 특징 공간의 하나의 점으로 표현될 수 있다.
      - 이미지는 다양한 픽셀로 구성이 되어 있다.
      - 이미지는 색상을 나타내는 RGB의 채널을 가지고 있으므로 고차원 상의 한 점으로 표현 가능하다.
      - 고차원 공간상에 존재하는 분포를 학습하는 방식으로 이미지 분포를 학습할 수 있다.
   2. 사람의 얼굴에도 어느정도 통계적인 평균치가 존재할 수 있다. 즉, 이를 수치로 표현할 수 있다.

생성모델은 실존하진 않지만 있을 법한 이미지 데이터를 생성할 수 있는 모델을 의미한다. 이 때 데이터는 이미지, 자연어 등이 될 수 있다.

c. **분류 모델**과 **생성모델**의 차이점

- **분류모델**
  데이터(X)가 주어졌을 때 label(y)이 나타날 조건부 확률 P(X|y)를 반환하는 모델. label 정보가 있어야 하므로 지도학습 모델이다.



- **생성모델**
  - 지도적 생성모델

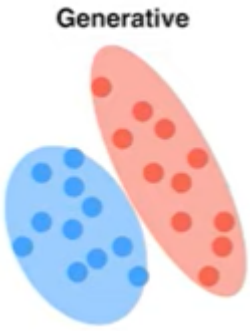
    label이 있는 데이터에 대해 각 클래스별 feature의 확률 분포 P(X|y)를 추정한 다음 베이즈 정리를 이용하여 P(y|X)를 계산하는 방법.

    - y에 대한 x의 조건부확률인 가능도 계산 방법

      1. P(x|y=k)가 특정한 확률 분포 모형을 따른다고 가정한다.
      2. k번째 클래스에 속하는 학습 데이터 {x_1, …, x_N|을 사용하여 이 모형의 모수값을 구한다.
      3. 모수값을 알고 있으므로 P(x|y=k)의 확률 밀도 함수를 구하였다. 즉, 새로운 독립변수 값 x가 어떤 값이더라도 P(x|y=k)의 값을 계산할 수 있다.

  - 비지도적 생성모델

    대부분의 생성모델. label이 없어서, 데이터 X 자체의 분포를 학습해서 X의 모분포를 추정하는 학습 데이터의 분포를 학습하는 모델.



    - 통계적 생성 모델

      밀도 추정. 관측된 데이터들의 분포로부터 원래 변수의 확률 분포를 추정하고자 하는 것.

- 딥러닝을 이용한 생성 모델

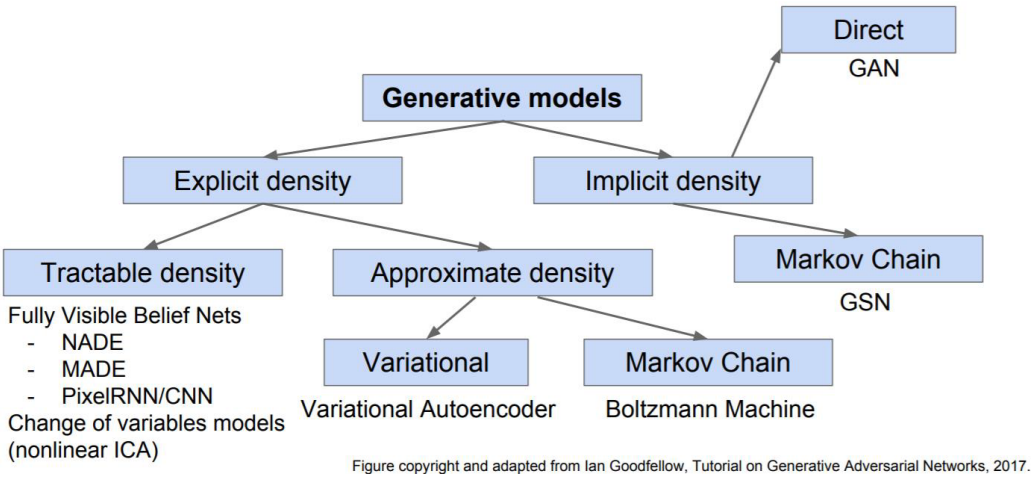


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

- Tractable Density : 데이터 X를 보고 확률분포를 '직접' 구하는 방법
- Approximate Density: 데이터 X를 보고 확률분포를 '추정'하는 방법
- Implicit Density : 데이터 X의 분포를 몰라도 되는 방법

GAN은 Implicit Density model에 속함.

PixcelCNN은 다루기 쉬운 밀도 함수를 정의하고 training data의 likehood를 최적화했다. VAE는 대조적으로 잠재변수 z를 가진 다루기 힘든 함수를 정의한다. 이 때, z는 좋은 property를 많이 가지고 있다. 하지만 "명시적으로 밀도를 모델링하는 것을 포기하고 sample을 얻으면 어떨까?" 하는 아이디어에서 GAN이 나오게 되었다.

2. GAN

GAN은 명시적인 밀도 함수를 적용하지 않고 게임이론으로 접근한다. 2 player game의 training 분포를 사용해서 생성하는 것을 학습하게 된다. 복잡하고 고차원의 샘플을 원하지만 이것을 직접적으로 할 방법이 없다는 점이 문제였는데, 이를 간단한 분포의 샘플을 얻어서 간단한 분포로부터 변형을 학습함으로써 해결하고자 하였다.

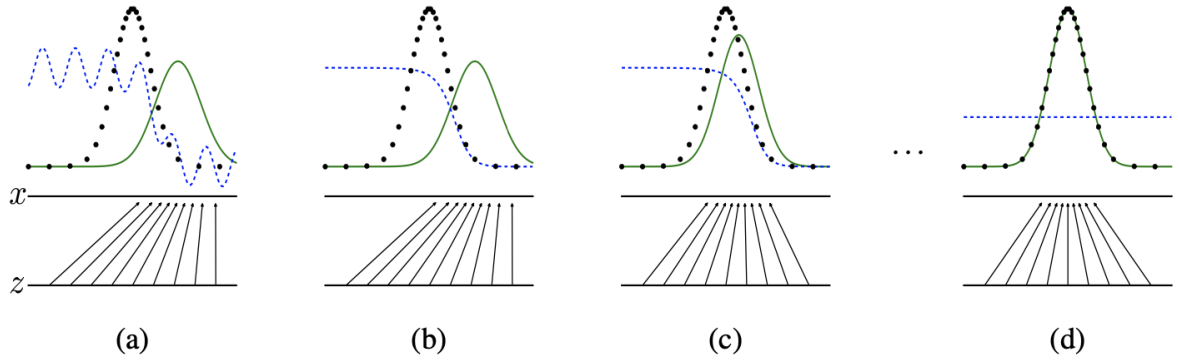
GAN 모델은 위의 그림과 같이 discriminator와 generator 네트워크로 구성되는데, 여기서 generator는 random noise에서 fake 이미지를 만들어서 discriminator를 속이는 것을 목표로 하고, discriminator는 입력 이미지가 'real' 또는 'fake'인지 구별하는 것이 목표이다.

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

위의 수식이 GAN의 학습과정을 의미하는데, Disciminator는 Objective function을 최대화하는 것이 목표이므로 logD(x)가 커야 하므로 D(x)가 1에 가까워지도록, D(g(z))는 0에 가까워지도록 최적화하게 된다. Generator는 Objective function을 최소화하는 것이 목표이므로, log(1-D(g(z)))가 0에 가까워지도록, 즉, D(G(z))가 1에 가까워지도록 최적화하게 된다. 여기서 1-D(G(z))가 0에 가깝다의 의미는 discrimator가 가짜 이미지를 진짜로 분류하고 있음을 의미하며, generator가 이미지를 realistic하게 생성하고 있음을 의미하게 된다. 즉, 1-D(g(z))가 0에 가까워질수록, genertor가 일을 잘 하고 있음을 의미하게 되므로 generator의 object function은 최소화하게 되는 것이다. 모델 학습이 끝나면 generator를 사용해서 새로운 fake 이미지를 생성하게 된다.

3. 이론적 분석

GAN의 이론적 분석은 Training Criterion이 데이터의 생성 분포를 G로 복구하고 D가 non-parametric limit에서 충분한 기능을 가지게끔 한다는 것을 보여준다.



위의 그림은 학습과정을 그림으로 나타낸 것이다.

여기에서 파란색 점선이 의미하는 것은 discriminative distribution(데이터 구분), 검은색 점선이 의미하는 것은 data generating distribution p_x(실제 데이터), 녹색 실선이 의미하는 것은 generative distribution p_g(생성한 데이터)이다.

그리고 수평선 z는 z가 샘플되는 영역(domain)을 의미한다. 여기에서는 균일하게 추출되게끔 설정하였다. 이 때 x는 x의 영역 중 일부이며 z에서 x로 향하는 화살표는 mapping x= G(z)가 non-uniform한 분포 p_g를 transformed samples로 향하게 하는 것을 나타낸다.

학습 과정의 inner loop에서 D를 완성시키기 위해 최적화하는 것은 계산적으로 불가능하다고 한다. 또한 학습을 위한 데이터셋이 유한하기 때문에 과적합에 빠질 가능성이 높다. D의 가중치를 계속 업데이트 하는 방법이 아닌 다른 방법을 택했는데,

1. D를 k번 학습
2. G를 한번 학습

1,2의 과정을 반복하여 최적의 solution에 가까운 D,G를 만들어내는 알고리즘을 채택했다.

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.

---

4. 이론적 결과

G는 p_g를 G(z)의 분포로 정의했다. 따라서 위에 있는 알고리즘이 G(z)를 p_data에 수렴하게끔 하길 원했다.

논문에서는 이론적 결과가 non-parametric setting에서 수행된다고 말했다. 다시말해서 확률밀도 함수에 수렴하는 것을 학습함으로써 무한한 능력을 가진 모델을 표현할 것이라고 말했다.

5. 실험

G,D를 MNIST, TFD(Toronto Face Database), CIFAR-10에서 학습시키고 테스트했다.

G는 활성화 함수로 ReLU, 시그모이드 함수를 혼합하여 사용하였고,

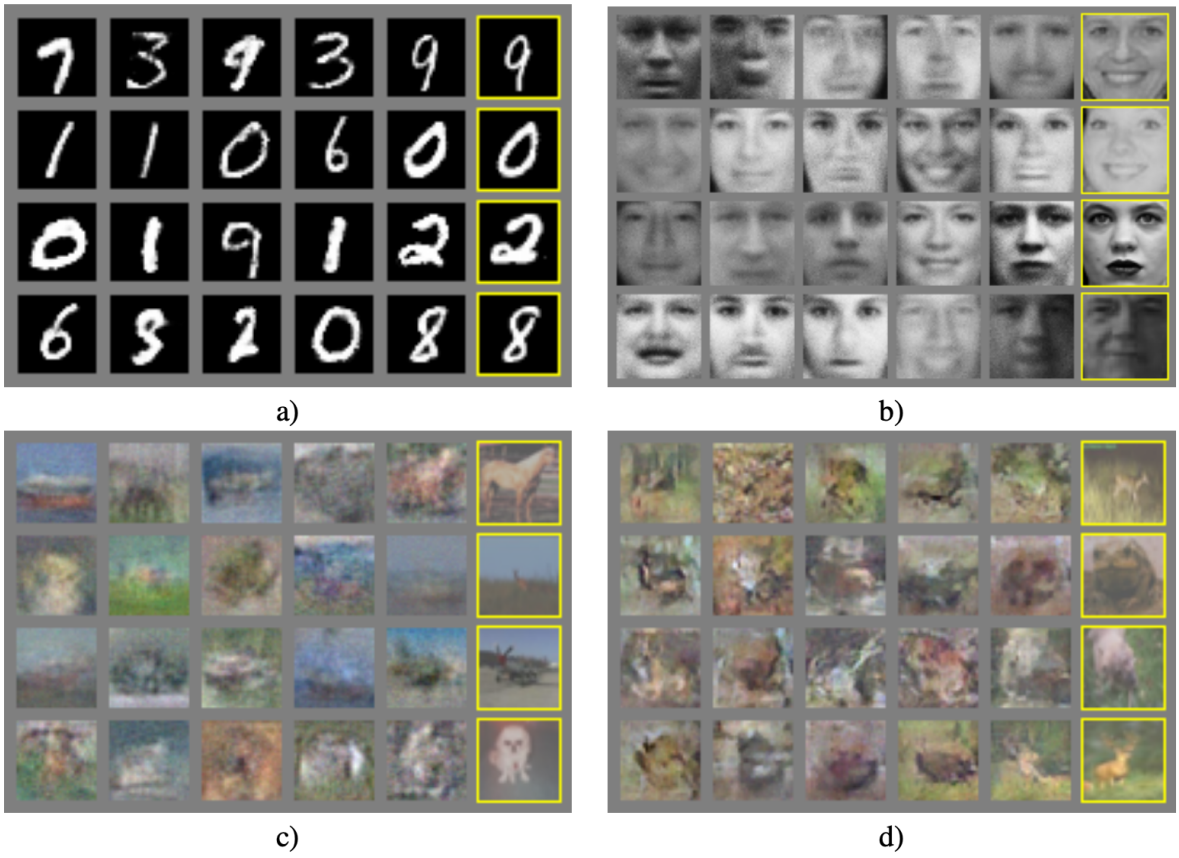
D는 maxout를 활성화 함수로 사용했다. 이 때 G는 noise를 입력데이터로 받아 데이터를 생성하였다.

p_g에 있는 test set data들의 확률은 G로 생성된 데이터들에 Gaussian Parzen window를 맞추는 방식으로 측정하며 측정된 확률은 p_g에 속한 log 확률로 변환한다. 이 때 Gaussian의 표준편차는 검증 데이터셋의 교차 검증을 통해 값을 정한다.

이 과정은 실제 확률을 다루기 힘든 생성모델들을 평가할 때 사용하는 방식이라고 한다.

평가 결과는 아래와 같다.

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	$\mathbf{225 \pm 2}$	$\mathbf{2057 \pm 26}$



노란색 박스가 생성된 이미지인데 그림을 보면 아주 잘 생성됨을 확인할 수 있다.