[Generative Adversarial Nets]

0. Abstract

- 2개의 모델을 동시에 train 하는 적대적인 과정(adversarial process)을 통해 생성 모델을 평가하는 새로운 프레임워크를 제시
  - 생성 모델 (generative model G) : 데이터 분포를 capture하는..
  - 판별 모델 (discriminative model D) : G 모델에서가 아닌 실제 training 데이터에서 나온 데이터라고 확률을 추정하는..
- 이 프레임워크는 minimax two-player game으로 볼 수 있음
  - G 모델은 D 모델이 실수할 확률을 maxmize
  - D 모델은 자신이 실수할 확률을 minimize

1. Introduction

- 지금까지 성공적인 DL에서는 고차원이며 방대한 sensory input을 class label과 맵핑해서 판별하는 모델들과 연관이 있었으며, 이러한 성능은 well-behaved gradient를 갖는 각각의 linear units를 사용하는 back-propagation과 dropout 알고리즘에 기반되어 있음

- 그러나 Deep generative model에서는 다음과 같은 어려움때문에 성공적인 성능를 얻지 못했음
  - 최대 우도 측정 및 그에 연관된 strategies에서 증가하는 확률 연살들을 근사화하는 어려움
  - generative context에서 각 linear unit들의 이점을 가져오는 어려움

- 제안된 adversarial nets 프레임워크는 적대적인 두 모델을 대립하는 것임
  - D 모델은 데이터 input이 G 모델의 생성 데이터 인지 train 데이터 인지를 구분하면서 학습함
  - G 모델은 위조지폐를 생산하고 탐지되지 않도록 하는 '위조자'라고 볼 수 있으며, D 모델은 위조지폐를 감별하려고 하는 '경찰'이라고 볼 수 있음
- 이러한 경쟁은 두 모델의 성능향상을 수반하며, 위조지폐가 실제 지폐와 구분이 불가능할 정도로 수준이 향상됨

- 이 논문에서는, G 모델이 multilayer perceptron을 통해서 랜덤 노이즈가 통과된 샘플 데이터를 생성하고, D 모델 또한 multilayer perceptron인 특수한 경우를 adversarial nets라고 함
- 즉, forward와 back-propagation과 dropout만을 사용해서 학습이 가능함

2. Related work

- DBMs(Deep Boltzmann machines)
- GSN(Generative stochastic networks)
- VAEs(Varational Autoencoders)
- NCE(Noise-contrastive estimation)
- PM(Predictability Minimization)
- advarsarial examples

3. Adversarial nets

- adversarial modeling 프레임워크는 모델들이 모두 multilayer perceptron일 때, 적용할 수 있는 가장 쉬운 방법임
  - 먼저, 데이터 x에 대한 G 모델의 분포 P_g를 학습하기 위해서는 노이즈 변수인 $P_Z(Z)$를 사전 정의해야함
  - G 모델은 파라미터 θ_g를 가지는 multilayer perceptron에 의해 표현되는 미분 가능한 함수이며, data space와 맵핑되어 $G(z;\theta_g)$로 표현됨
  - 또한 단일 스칼라를 출력하는 두번째 multilayer perceptron, $D(x;\theta_d)$를 정의함 $D(x)$는 x가 P_g가 아닌 원본 데이터에서 나왔을 확률을 나타냄

기호	설명
x	데이터
P_g	x에 대한 생성자(G 모델)의 분포
$P_Z(Z)$	input noise 변수
θ_g	multilayer perceptrions의 parameters
G	θ_g에 의해 표현되는 미분가능한 함수
$G(z;\theta_g)$	data sapce와 맵핑되어 표현
$D(x)$	x가 P_g가 아닌 원본 데이터에서 나왔을 확률
$D(x;\theta_d)$	두 번째 multilayer perceptron

■ GAN의 loss function 수식은 다음과 같음

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

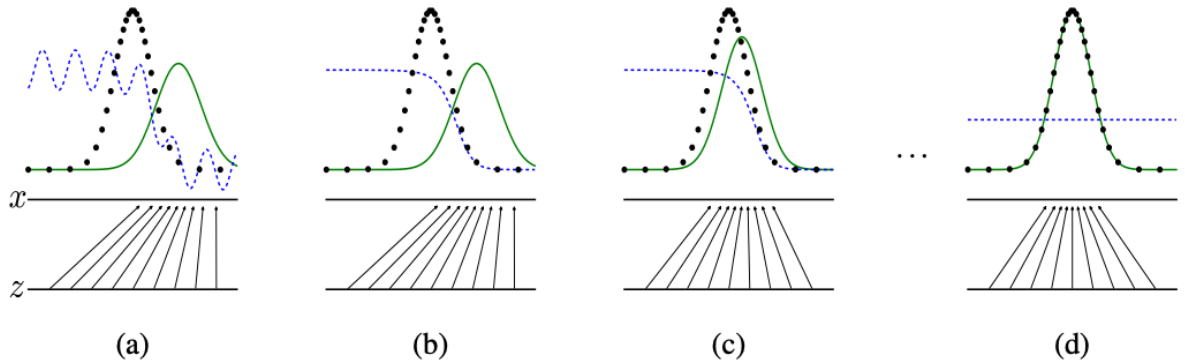
- 모델 G는 V(D,G)를 최소로, 모델 D는 V(D,G)를 최대로 만들려는 과정은 two-player minimax game으로 볼 수 있음
- training 데이터와 G 모델에서 생성되는 데이터로부터 올바른 라벨로 분류할 확률을 최대화 하도록 D 모델을 학습함
- 동시에, 모델 G는 $\log(1 - D(G(z))$ 을 최소화 하도록 학습함(식의 두번째 항)
  - ✓ D 모델이 옳게 분류했을 경우는 $D(G(z)) = 0$ 이며, 반대로 잘못 분류했을 경우에는 $D(G(z)) = 1$ 이며, $\log(1 - D(G(z))$는 $-\infty$ 값을 가지게 됨
- 결과적으로, 모델 D의 이상적인 V(D,G)의 값은 0 이며, 모델 G의 이상적인 V(D,G)의 값은 $-\infty$ 이라고 할 수 있음

■ 위 GAN의 loss function 수식에서는 G 모델이 잘 학습되도록 충분한 Gradient를 제공하지 못할 수도 있음. 즉, 학습 초기에는 D 모델은 높은 정확도로 G 모델의 생성 데이터와 원본데이터를 구별할 수 있음

■ 이럴 경우에, $\log(1 - D(G(z))$는 포화상태이며, G 모델은 $\log(1 - D(G(z))$을 최소화하는 것보다 $\log(D(G(z))$를 최대화하도록 할 수 있음

■ adversarial nets이 data의 분포를 학습하는 과정은 다음과 같음

- ✓ 파란 점선은 discriminative distribution(D)
- ✓ 검정 점선은 원본 데이터(P_x)
- ✓ 초록 실선은 생성된 데이터(P_g, G)



- (a) : 초기 상태, 즉 D 모델은 정확한 분류기라고 할 수 있음
- (b) : D 모델 학습 하며, $D = \frac{P_{data}(X)}{P_{data}(x) + P_g(x)}$ 로 수렴함
- (c) : G 모델이 학습되면서 원본 데이터와 유사해지고 있음
- (d) : 여러번 학습 후, G 모델은 원본 데이터와 똑같은, $P_g = P_x$ 상태까지 향상되며 D 모델은 두 분포를 구별하지 못하고 $D(x) = 1/2$ 이 됨

4. Theoretical Results

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.

- 전체 iterations 과정
  - D 모델을 학습하는 과정 (k_steps)
    - ✓ noise가 적용된 m개의 생성 데이터
    - ✓ m개의 원본 데이터
    - ✓ D 모델 업데이트
  - G 모델을 학습하는 과정
    - ✓ noise가 적용된 m개의 생성 데이터
    - ✓ G 모델 업데이트

- D 모델을 먼저 일정 수준 이상으로 학습시킨 이후에 G 모델을 학습하는 과정을 가지게 됨. 학습 초반 G 모델의 낮은 성능은 D 모델의 판별 성능을 향상시키고 이 후, D 모델의 판별 성능은 G 모델의 높은 성능을 보장함

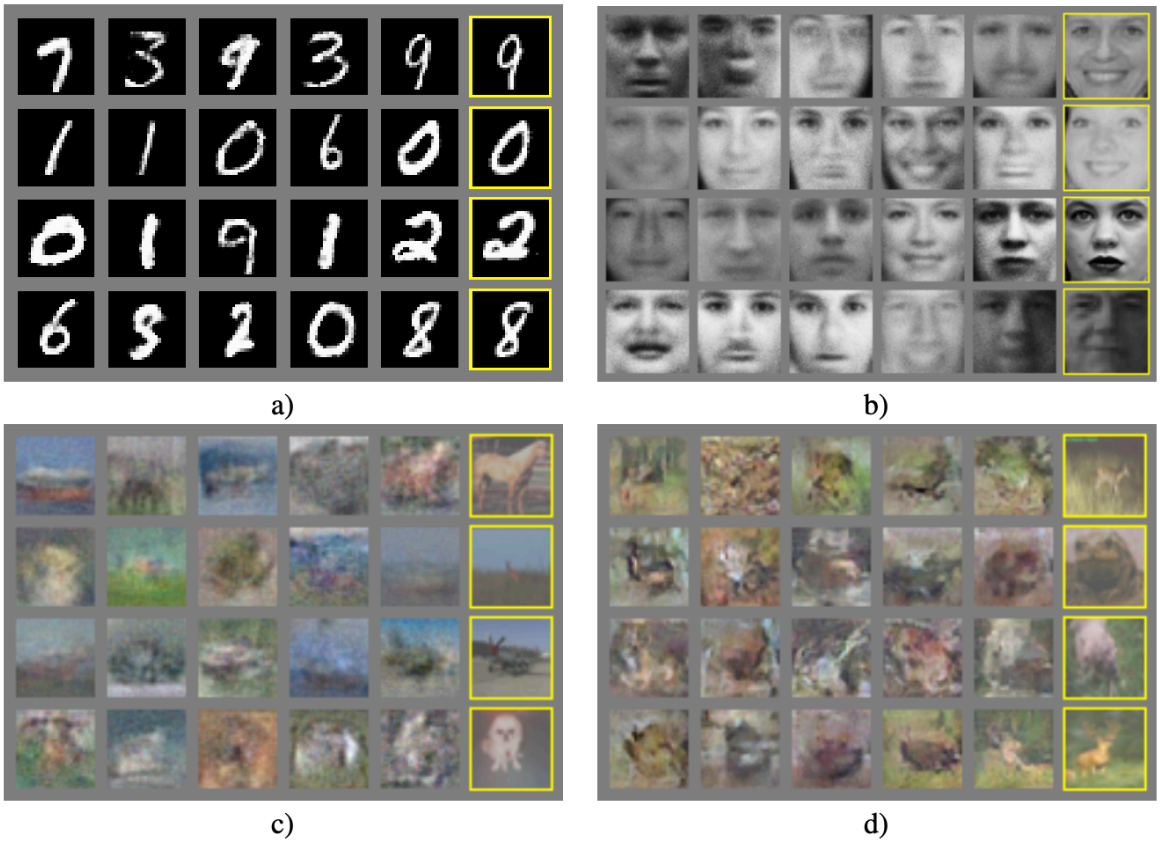
- proposition 1

- proposition 2

5. Experiments

- adversarial nets를 MNIST, Toronto Face Database(TFD), CIFAR-10에 적용함
  - G 모델에는 rectifier linear activation과 sigmoid activation을 혼합해서 사용하였으며, D 모델에는 maxout activation을 사용함
  - D 모델을 학습할 때만 dropout을 사용함
  - Input noise는 G 모델의 가장 아래 layer에 적용함

- 학습이 진행된 이후 G 모델에서 생성한 데이터는 다음과 같음(가장 오른쪽 열)



a) b)

c) d)

6. Advantages and Disadvantages

- 단점
  - $P_g(x)$에 대한 명시적인 표현이 없음
  - D 모델은 G 모델은 균형을 잘 맞추어 학습되어야함. G 모델은 D 모델이 향상되기 전에 너무 학습되는 것을 피해야함)
- 단점
  - Markov chains이 필요 없으며 오직 back-porpagation만으로 gradients를 얻을 수 있음
  - 학습 중 추론이 필요 없음
  - 다양한 function들과 모델이 결합될 수 있음
  - 퇴화분포에서도 선명한 결과를 얻을 수 있음

7. Conclusions and future work

- G 모델과 D 모델로 c를 추가함으로써, conditional generative model로 발전할 수 있음
- x가 주어졌을 때, Learned approximate inference는 z를 예측하는 보조 network가 될 수 있음
-
- semi-supervised learning으로, 판별과 추로 net의 feature는 정해져있는 레이블 데이터를 사용할 때, 분류의 성능을 향상시키킬 수 있음
-