[논문리뷰] Generative Adversarial Networks

🌟 Introduction

Adversarial Nets Framework?

- 제안 배경: probabilistic computations(최대 확률 추정 등의 분야에서 사용됨)을 추론하는 것에 대한 어려움, 생성 모델에서 부분적 선형 유닛의 이점을 이용하는 것의 어려움 ⇒ **이를 해결하는 생성모델의 새로운 측정 과정!**

▼ 기존 Gernerative Model의 한계

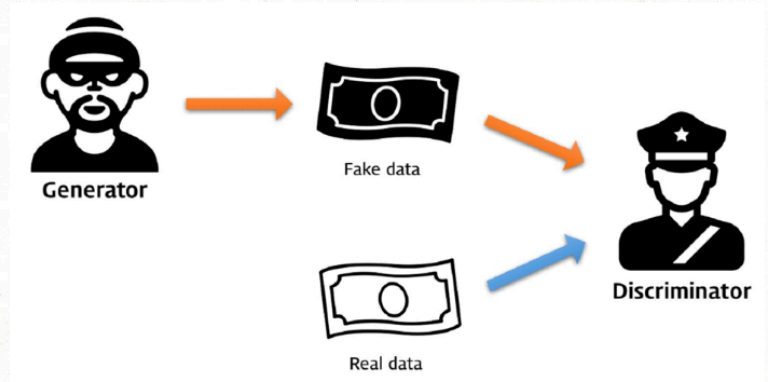
1. 확률 그래프 모델 기반 생성기: RBM, DBM 같은 undirected 그래프 모델은 정규화 상수의 계산이 어렵고, MCMC 기반 추론은 느리고 불안정
2. Deep Belief Networks(DBNs): 계층 별로 빠른 근사학습은 가능하나, 양쪽(비·방향성)의 계산 단점을 모두 가짐
3. Score Matching, Noise-contrastive Estimation(NCE) : 로그우도 대신 다른 기준으로 학습하려 하지만, 여전히 정규화 상수가 포함된 확률 밀도 함수가 명시되어야 함. 특히 NCE는 고정된 노이즈 분포를 사용하기 때문에, 학습이 느리고 부분적인 변수에만 잘 맞는 경향.
4. Generative Stochastic Network(GSN), Denoising AE : 분포 명시 없이 마르코프 테인을 사용하여 샘플링하기 때문에, 느리고 feedback loop 존재

1. **Generative Model(G)**: D를 속이는 모델
2. **Discriminative Model(D)**: 실제 데이터와 G가 만들어낸 데이터를 구별

⇒ 1과 2는 **경쟁하면서 발전**
(G는 랜덤 노이즈를 multi-layer perceptron에 넣어 모방 데이터를 만들고,
D는 multi-layer perceptron에 데이터를 넣어 모방인지 진짜인지 구분을 반복)

= Adversarial nets

- GAN은 생성자 G와 판별자 D가 경쟁하며 학습하는 구조로, 추론 네트워크나 마르코프 체인 없이 학습 가능하다.



Adversarial Nets

[기호 정의]

- $p_z(z)$ = 노이즈 변수 z의 사전 분포(정규분포, 균일분포 등등)
- $G(z;\theta_g)$ = 파라미터 θ_g를 가지는 MLP에서 구한 미분가능 함수 (g로 이루어진 multilayer perceptron)
- $D(x;\theta_d)$ = 하나의 스칼라 값을 출력하는 파라미터 θ_d를 가지는 함수 (입력 데이터가 p_g가 아닌 진짜 데이터 분포일 확률을 single scalar로 나타내는 함수)

[Training Method]

"D와 G의 minimax game with value function V (G, D)"

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

1. **D의 학습목표**

   : 실제 데이터 분포 p_data(x)에서 x를 뽑았을 때, log(D(x))(=판별자가 진짜라고 판단할 확률)의 기댓값
   → 학습 데이터셋과 G에서 만든 데이터에 정답을 할당할 확률을 최대로 만들자!

$$E_{x\sim p_{data}(x)}[logD(x)]$$

2. **G의 학습목표**

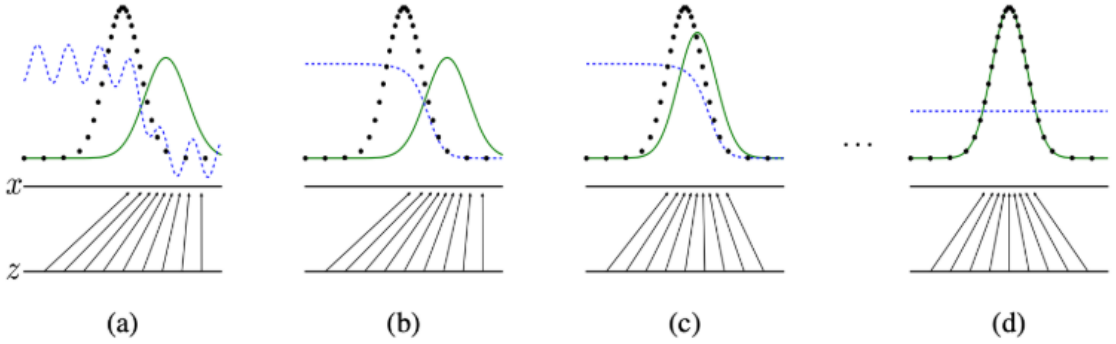
   : 노이즈의 사전분포 p_z(z)로부터 z를 뽑아서 만든 가짜이미지 G(z)를 판별자가 가짜라고 판단할 확률의 기댓값

→ 정답을 제대로 할당할 수 없게 학습 데이터셋과 유사한 데이터를 만들자!

$$E_{z\sim p_z(z)}[log(1-D(G(z)))]$$

⇒ log(D(x)) = D 최대, log(1-D(G(z))) = G가 최소로 만드려는 대상

Theoretical analysis of adversarial nets



- 파란색 점선 : discriminative distribution(데이터 구분 = **D**),
  검은색 점선 : data generating distribution **p_data**(실제 데이터),
  녹색 실선 : generative distribution **p_g**(생성한 데이터)
- (a) : 학습이 안된 상태. p_g가 p_data와 형태가 비슷하고, discriminative distribution이 부분적으로 정확한 모습.
- b : D가 D*(x) = p_data(x)/((p_data(x) + p_g(x))로 수렴하여, 실제 데이터와 생성된 데이터를 구분할 수 있게 학습됨.
- c : G 학습. 이 때 D의 그래디언트는 G(z)가 실제 데이터와 같은 데이터로 분류하게끔.
- d : G와 D가 충분한 능력을 가질 때까지 a~c를 여러번 반복. G는 p_g = p_data라고 할정도로 데이터를 잘 생성해내고, D는 두 데이터의 차이를 구분할 수 없다(D(x) = 0.5).

Theoretical Results

(증명 나중에 추가....)