About GAN

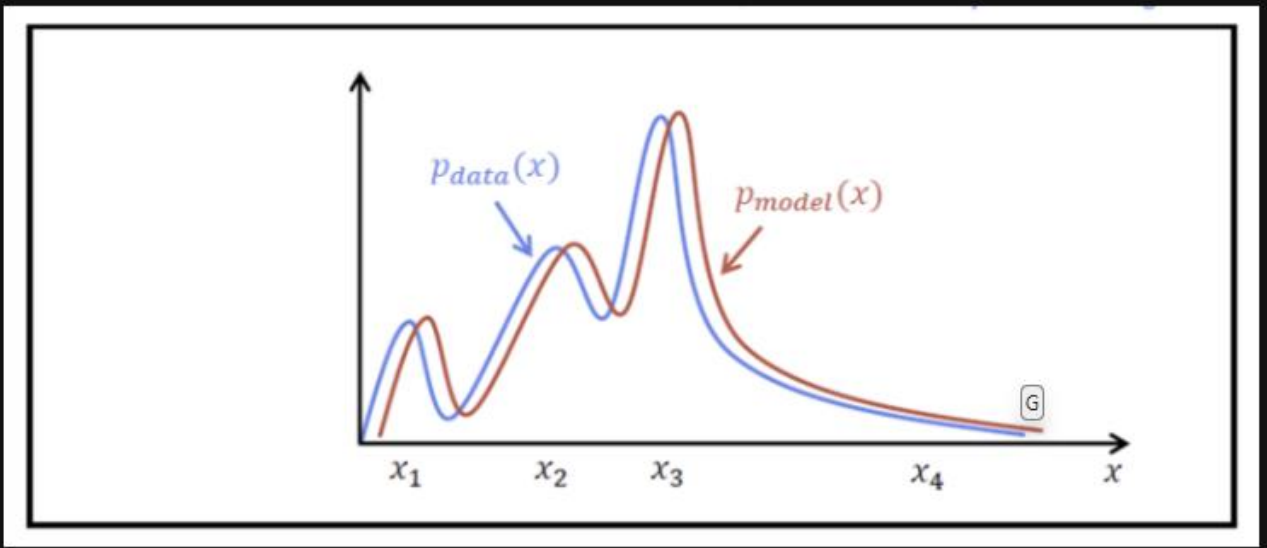
GAN Loss function의 그래디언트를 이해하기 위한 수식적 배경.
introduction에서 소개하는 다양한 문제를 이해할 수 있게 한다.

generative method

- generative method: 고차원 데이터 분포(예: image)로부터 새로운 샘플을 생성하는 생성 방식. (=**샘플링**)
- 대표적인 generative method
  - autoregressive model
    이미지 구성 픽셀을 순차적으로 조건부확률 분포 모델링
  - VAE(변분 오토인코더)
    z=Encode(x) 로 잠재벡터 뽑아 decoder에서 생성
  - **GAN**(Generative Adversarial Network)
    generator와 discriminaor가 경쟁하며 학습 (**SOTA)**

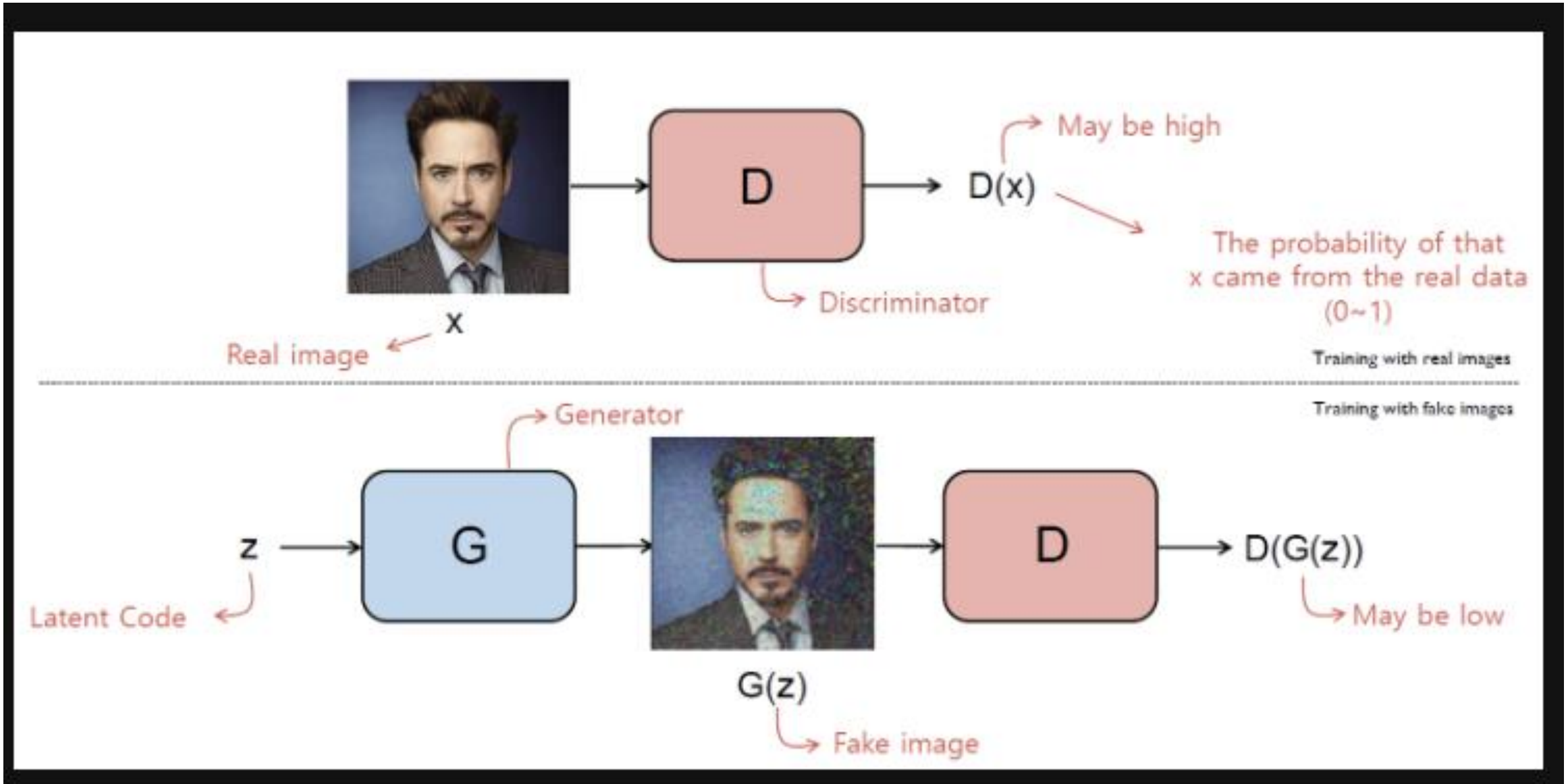
generative method

Generative Model의 Goal



- $p_{\text{data}}(x)$에 근사하는 $p_{\text{model}}(x)$들 찾기
- $p_{\text{data}}(x)$: 실제 학습 데이터의 분포
- $p_{\text{model}}(x)$: 모델이 생성한 데이터의 분포
- 두 분포의 차이를 최소화하기

Vanilla GAN


D
May be high
D(x)
The probability of that
x came from the real data
(0~1)
Discriminator
x
Real image
Training with real images
Training with fake images
Generator
z
G
D
D(G(z))
Latent Code
May be low
G(z)
Fake image

Vanila GAN Loss function

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

GAN (Goodfellow et al.,2014)

목적: D(x) = D(G(z)) = 0.5

진짜와 가짜 분포 구분 못하게 차이 줄이기.

(목적함수)

Vanila GAN loss function

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

$$D(x) = \sigma(f(x)) = \frac{1}{1 + e^{-f(x)}}$$

D(x): discriminator의 이미지 구분 함수.
최종적으로 sigmoid 함수로 확률적 해석.

- z: 잠재벡터
- z~p(z): latent variable 샘플링 정규분포.
- x : 입력 이미지 벡터.
- $x \sim p_{data}$: 이미지 픽셀 분포. 실제 데이터 샘플링

Vanila GAN loss function

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

앞 항: 실제 data가 들어올 때
Discriminator가 진짜로 보도록 함

뒷 항: 생성된 가짜 data G(z) 들어올 때
Discriminator가 가짜로 보도록 함

Vanila GAN loss function

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

1. z~p(z) : 아무 잠재벡터나 랜덤
2. x′ = G(z) : 잠재벡터에서 generating
3. discriminator에 $x_{real} \sim p_{data}$ 와 x_{fake} 섞어 fowarding (x′)
4. z′ = D(x′) : x′가 real일 확률 계산. (binary classification)
5. backpropagation:

D(x), G(z)의 내부 파라미터 θ_D, θ_G로 편미분

Vanila GAN loss function

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

G/D간 경쟁
D: log probability 최대화
real D(x)는 1에 가깝게 만들고 싶음. fake D(G(z))는 0으로 만들고 싶음.

G: log probability 최소화
G(z)를 조정해, D(G(z)) 1로 만들어 최대화

최종 목적: D(G(z)) = 0.5 (구분 못하겠다!)

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

[NOTE]
Loss Function의 한계와 개선

(1. Introduction 3문단과 연관)

위 Loss Function의 한계가 밝혀지고 다양한 해결 방법이 논의되었다.

논문에서 주로 다루지는 않는다고 하였으나, gradient와 연관된 문제의 분석에 있어 중요하게 다뤄지므로 정리해보았다.

Vanila GAN loss function's Gradient

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

➤ D에 대한 gradient:

(실제 x, 생성 $G(z)$에 대해 별도로 미분)

▸ 실데이터 쪽 미분:

$$\frac{\partial}{\partial D} \log D(x) = \frac{1}{D(x)}$$

▸ 가상데이터 쪽 미분:

$$\frac{\partial}{\partial D} \log(1 - D(G(z))) = \frac{-1}{1 - D(G(z))}$$

> 전체적으로 D는 **진짜에 대해서는 D(x)가 커지도록 (→ 1),**
> **가짜에 대해서는 D(G(z))가 작아지도록 (→ 0) 유도됨**

➤ G에 대한 gradient:

$$\frac{\partial}{\partial G} \log(1 - D(G(z))) = \frac{-1}{1 - D(G(z))} \cdot \frac{\partial D(G(z))}{\partial G(z)}$$

즉:

- $\frac{\partial D(G(z))}{\partial G(z)}$ → discriminator의 gradient
- $\frac{\partial G(z)}{\partial G}$ → generator 내부 gradient (backpropagation 통해 계산)

Jensen-Shannen Divergence와 Gradient vanishing/exploding problem

$$\frac{\partial}{\partial D}\log D(x) = \frac{1}{D(x)}$$

$$\frac{\partial}{\partial D}\log(1 - D(G(z))) = \frac{-1}{1 - D(G(z))}$$

$$\frac{\partial}{\partial G}\log(1 - D(G(z))) = \frac{-1}{1 - D(G(z))} \cdot \frac{\partial D(G(z))}{\partial G(z)} \cdot \frac{\partial G(z)}{\partial G}$$

JSD 기반의 Discriminator는 다음을 최적화합니다:

$$\max_D \mathbb{E}_{x \sim P_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim P_z}[\log(1 - D(G(z)))]$$

하지만 $D(G(z)) \approx 0$, 즉 Discriminator가 너무 잘할 경우:

- Generator의 gradient:

$$\nabla_G \log(1 - D(G(z))) \approx 0$$

→ **gradient vanishing**

[NOTE] KL Divergence

p, q: 확률분포에서 특정 상태가 나타날 확률.
KL Divergence: 두 확률분포 간 정보 손실량의 기댓값.
각 확률분포의 로그우도 값에 대해 차이를 기댓값을 취한 것.

$$D_{KL}(p||q)=E[log(p_i)-log(q_i)]=\sum_i p_i log\frac{p_i}{q_i}$$

문제: 1. symmetric 하지 않다.
둘 중 어느쪽이 분포 기준인가에 따라 값이 달라진다.
낮을수록 두 확률분포가 유사한 건 맞지만,
척도가 되기에는 부적절.

$$D_{KL}(P||Q)\neq D_{KL}(Q||P)$$

2. Q = 0 & P > 0 인 경우, KL Divergence값이 무한대.
따라서 P > 0인 모든 부분에서 Q > 0 이되어야 한다. (support가 포함관계)

[NOTE] Jensen-Shannon Divergence

P, Q의 중간값 M과의 KL divergence를 하면서 대칭성을 띄도록 설계

$$JSD(P,Q) = \frac{1}{2}D(P||M) + \frac{1}{2}D(Q||M)$$

$$where \ M = \frac{1}{2}(P+Q)$$

1. JSD(P||Q) = JSD(Q||P)
2. 항상 $0 \leq JSD(P||Q) \leq \log 2$
3. 분포가 겹치지 않아도(support가 일부 달라도) 정의됨
4. 정보이론적인 해석이 여전히 가능.
   두 분포를 평균 낸 혼합 분포와 얼마나 다른가?

GoodFellow et al. 2014 GAN에서 증명

최적의 Discriminator 식을 Loss function에 대입할 때, Generator를 최적화시키는 것은 JSD를 최소화하는것과 같다!

(설계 해본 결과 그렇게 유도가 되더라...)

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

그런데 이걸 최적의 D를 대입하면 (Goodfellow, 2014 논문 참고):

$$\min_G V(G) = -\log 4 + 2 \cdot \text{JSD}(P_{\text{data}} \parallel P_g)$$

즉, **GAN의 이상적인 최적화는 결국 JSD를 최소화하는 것과 동일합니다.**

[NOTE] 증명 (생략)

🎯 목표

GAN에서:

- **Generator $G(z)$**는 잠재 분포 $p_z(z)$에서 샘플을 받아 가짜 샘플을 생성하고
- **Discriminator $D(x)$**는 입력이 진짜인지 가짜인지 구분하는 확률을 출력함

GAN은 다음 **minimax game**을 학습합니다:

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

---

① Discriminator가 최적일 때의 형태

Discriminator는 x에 대해 $D(x) \in [0,1]$ 값을 출력합니다.

이 때, Generator G는 고정해두고, **Discriminator D**만 최적화해보면?

▶ Discriminator의 목적: 진짜일 확률을 최대화

$$\max_D V(D,G) = \int_x p_{\text{data}}(x)\log D(x) + p_g(x)\log(1-D(x))\,dx$$

이걸 $D(x)$에 대해 최적화하면, calculus(미적분)적으로 해를 찾을 수 있고,
최적 Discriminator는:

$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)}$$

② 최적 $D^*(x)$을 넣은 후의 Value Function

위에서 구한 최적의 $D^*(x)$를 원래의 value function $V(D,G)$에 대입하면:

$$V(D^*,G) = \int_x p_{\text{data}}(x)\log\left(\frac{p_{\text{data}}(x)}{p_{\text{data}}(x)+p_g(x)}\right) + p_g(x)\log\left(\frac{p_g(x)}{p_{\text{data}}(x)+p_g(x)}\right)dx$$

정리하면,

$$V(D^*,G) = -\log 4 + 2\cdot \text{JSD}(p_{\text{data}} \,\|\, p_g)$$

> 👉 이게 핵심입니다. GAN의 loss function을 최적의 D로 평가한 값은 **Jensen-Shannon Divergence의 2배 – log 4** 와 같습니다.

Jensen-Shannon 기반 Divergence 한계 : Gradient Vanishing

최적의 Discriminator를 사용

$$L(\theta) := V(G(z;\theta)) = \max_D V(G,D) = 2JSD(P_{data} \| P_g(\theta)) - \log 4$$

Generator를 학습한다는 의미:
자신이 만들어내는 확률분포 Pg가
실제 데이터 분포와 가까워지도록,
G의 파라미터로 편미분 해가며 backpropagation

Jensen-Shannon 기반 Divergence 한계 : Gradient Vanishing

$$L(\theta) := V(G(z;\theta)) = \max_D V(G,D) = 2JSD(P_{data} \| P_g(\theta)) - \log 4$$

θ가 분포가 거의 안 겹치는 구간에 위치할 경우

$P(x) > 0$인 영역에서 $Q(x) \approx 0 \Rightarrow M(x) \approx \frac{1}{2}P(x) \Rightarrow \frac{P(x)}{M(x)} \approx 2 \Rightarrow \log \frac{P(x)}{M(x)} \approx \log 2$

즉, 전 영역에서 거의 일정한 값으로 KL이 채워지고 있고,
→ Q가 조금 움직이는 정도로는 $M(x)$이 거의 안 바뀝니다.

> 그 결과:
> JSD는 분포가 겹치기 시작하기 전까지 거의 변하지 않음 → flat

Jensen-Shannon 기반 Divergence 한계 : Gradient Vanishing

- 원래는 P와 Q가 정의된 부분에서 어느 정도 겹치는 부분이 있다면 괜찮음 (support 교집합 존재)
- 하지만 학습 초기:

P_{data}는 진짜 이미지 주변에만 밀도가 있고

$P_{generator}$는 이상한 노이즈 이미지 주변에만 밀도가 있음.

(안 겹치는 support가 상당히 많다.)

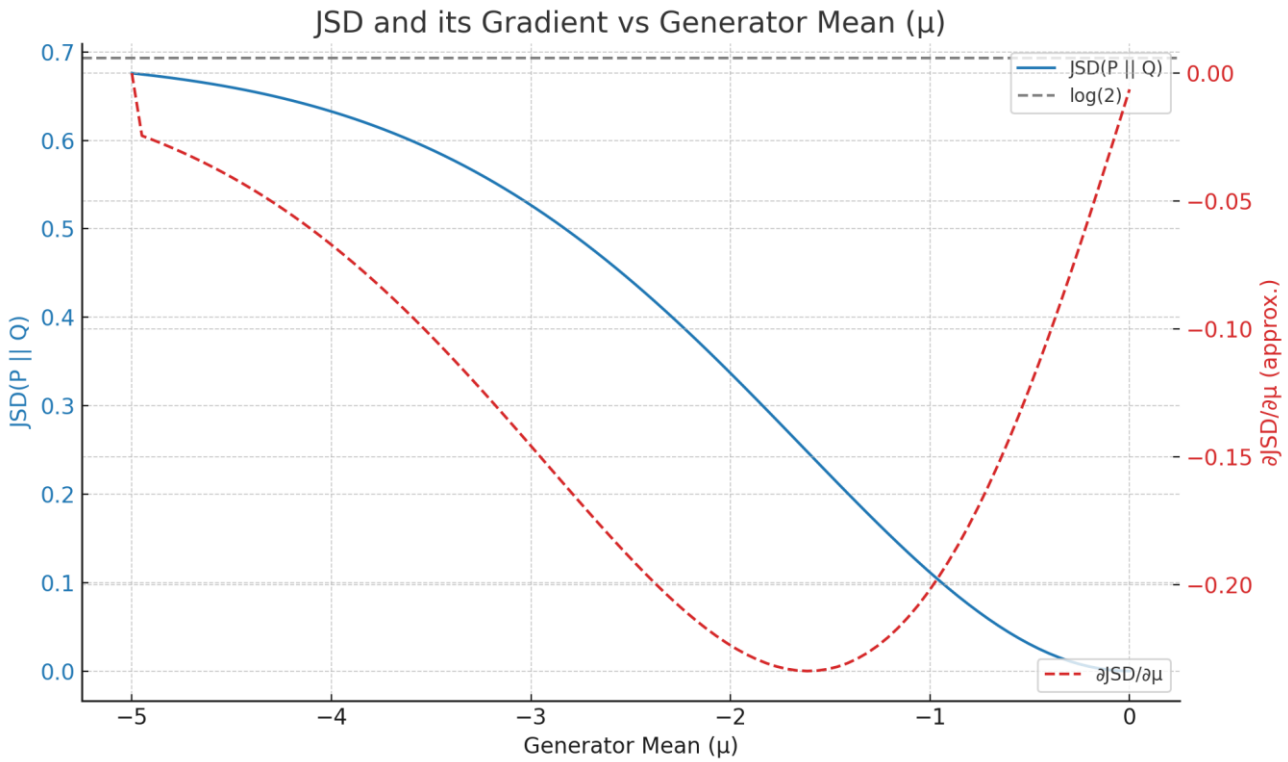
loss function& 그래디언트 예시

파란 선: JSD(P||Q)

- $\mu = -5$ 근처에서 JSD는 $\log 2 \approx 0.693$에 수렴합니다.
- 이 시점은 P와 Q의 지지집합이 거의 안 겹치는 구간입니다.
- 이때는 손실 함수가 **상수처럼 flat**합니다.

빨간 점선: $\frac{d}{d\mu} \mathrm{JSD}(P \parallel Q)$

- μ가 -5일 땐 gradient도 거의 **0**
  → 즉, Generator가 μ를 움직여도 **손실이 변하지 않음**
- μ가 0에 가까워질수록 → **gradient가 살아납니다**
  → 두 분포가 겹치기 시작하면 **JSD가 민감하게 변하고**, 학습 가능



해결점: Wasserstein 기반 거리함수

- JSD 정의 구조가 Q가 support 외부에 있을 때는 거리 정보를 거의 반영하지 못함.
- GAN에서는 JSD 대신 Wasserstein 거리(지구이동 거리) 를 사용하게 됨

결론 (요약)

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

- loss function이 Jensen-Shannen Divergence로 최적화됨이 증명되는데,
- 분포가 너무 멀리 떨어지면, JSD는 변하지 않는 평평한 구간을 형성하고,
- 이때는 G의 파라미터 θ_g를 아무리 바꿔도 gradient가 너무 작아서, 학습이 정체되거나, "gradient가 무작위 방향으로 튄다". (논문 1. introduction 3문단)
- 그래서 이 loss function 대신 Wasserstein 기반 함수를 쓴다.

기존 GAN의 특징

- 선명한 이미지 (픽셀 분포가 blurry하지 않다)
- 낮은 해상도
- 제한된 variation(다양성; 평균적인 pattern만 주로 생성한다)
- 훈련의 불안정성 문제

gradient instability(방향 무작위성),

escalation(신호 폭주: explode / vanish)

mode collapse(몇가지 pattern만 생성)

고해상도 이미지 생성에 대해

- 고해상도 이미지 생성시, 훈련 이미지와 생성 이미지 간 구분을 하기 쉬워짐
- => Discriminator가 먼저 빠르게 학습을 해버림(gradient problem)
- 메모리 제약 => mini batch size 저하 => mode collapse 발생.

- 2장에서 그 해결 방안을 제시

GAN의 다양성 저하에 대해

다양성: 생성 모델이 얼마나 다양한 샘플을 보존하고 있는가?

앞선 원인 말고도, 초기 GAN은 가장 평균적으로 많이 나타나는 특징만을 생성하려고 하는 경향이 컸음.

⇒다양성 감소

추후 3장에서 다양성 평가 방법 제시

부록

학습 초기 두 분포가 겹치지 않아 발생하는 문제

- https://chatgpt.com/share/686300b3-5f0c-8008-b254-ac015bb724af