Paper summary

AI VISION Lab

1. 공부한 논문의 제목, 게재된 학회 혹은 저널 등 논문 기본 정보를 적으세요.
   - **A. 이름: EDM: Elucidating the Design Space of Diffusion-Based Generative Models**
   - **B. 저널: NeurIPS**
   - **C. 도메인: Diffusion**
   - **D. 출판연도: 2022**
   - **E. 저자: NVIDIA (Tero Karras et al)**

2. 논문에서 제안한 알고리즘 및 프레임워크에 대해 본인이 이해한대로 다이어그램을 그려보세요. **논문 Figure를 그대로 따라 그리면 안됩니다.**

   A. [선행연구] SDE, ODE를 제안한 Yang song의 논문을 바탕으로 전개함.

   B. 본 저자는, 기존의 Diffusion Model (VE, VP)의 Framework의 경우는 서로 의존성이 존재하다 보니, Model의 전체구조를 수정하는 것이 아니라면 변화를 주기 어렵다고 판단함.

   C. 따라서, Diffusion Model의 구성요소들에 대해서, 발전된 구조와 함께, 각 요소 사이에 존재하던 의존성을 제거하고자 함.

   D. SDE, ODE의 경우 1 Step의 변화를 dx로 설명하고 있음. 그러나 해당 dx의 경우, f(x, t) - drift항과, g(t) - diffusion항으로 설명하고 있으며 이는 DDPM 기준, β에 대한 식으로 설명 가능함.

   E. 그러나 실제 Diffusion을 적용하는 과정에서는, dx가 아닌, p_{0t}를 계산해서 사용하고 있다는 점에서, 계산이 여러 번 필요하며 설명이 복잡하다는 한계가 있었음

   F. 따라서, dx에 대한 식에서 출발하는 것이 아닌, p_{0t}를 s(t)와, $\sigma(t)$로 정의한 후, dx를 정의함. 즉, Diffusion의 모든 요소들을 s(t)와, $\sigma(t)$로만 설명할 수 있게 됨.

G. ODE의 경우, Stochasticity 가 없는 구조다 보니, 정확하고, Sampling을 효과적으로 할 수 있다는 장점이 있음. 따라서, 논문에서는 우선 ODE 기반의 Sampler를 제안함.

H. 기존의 Sampler로 가장 많이 사용되는 ODE Solver는 Euler's Method이며, 이는 1st 에 해당함. 즉, Trajectory를 예측해야 하는 Solver의 특성상, 예측 정확도(Truncation Error)가 낮을 수밖에 없음. 또한, Runge-Kutta's method의 경우 high order의 Solver로, 예측 정확도가 높으나, 연산요구량이 상당히 많다는 한계가 있었음. 따라서, 논문에서는 정확도를 높이되, Step에 정확히 비례하는(추가 계산이 거의 없는) Trade-off(quality – NFE: D_θ를 몇 번 호출했는가?) 가 가능한 Sampler를 제안하고자 함. 따라서, 2nd Heun's method를 제안함.

I. SDE의 경우, ODE 와 달리 Stochastic하다는 점 때문에 다루기 어렵다는 한계가 있으나, ODE 대비 성능이 높다는 장점이 있고, 논문에서는 ODE에서 제시한 구조를 기반으로 SDE Sampler를 정의함. (2nd Heun + Langevin Like)

J. 또한, Training과정에 대해서도 정의하는데, 기존의 Model은 Sigma Level에 영향을 직접적으로 받는 구조라는 한계가 있었음. 또한, 이 논문에서는, NN이 직접 D_θ(Score Function of P를 기반으로 Noise를 제거한 Data)를 예측하는 것이 아닌, Score Function of P를 예측하고, 이를 기반으로 D를 구하는 구조를 제안함. (안정적)

K. 결과적으로, 제안한 모든 Option을 결합했을 때, 가장 높은 성능을 보임.

3. 본인이 생각하는 이 논문의 장점이 무엇이라고 생각하나요? **논문 Contribution bullet을 그대로 따라 적으면 안됩니다.**
   A. Noise Scheduler, Mean, Variance 구조 간의 의존성에서 벗어나, 각 구성 요소를 제어할 수 있는 Framework를 제안했다는 점과, 기존의 Euler's method가 아닌, Heun's method를 제안함으로써, Sampling 성능을 향상시키거나, dx를 통합한 형태에서 정의함으로써, Parameter에 대한 통일성을 부여했다는 점 등, 결과적으로 Diffusion Model의 다양한 구성요소들에 대해서, 하나씩 기여했다는 점이 가장 큰 장점이라고 생각함.

4. 이 논문을 읽으면서 느낀 점, 혹은 배운 점이 있으면 적어보세요.

A. 선행연구와 비교과정을 상세히(식, 그래프) 설명해 둠. 이를 바탕으로 추후, 복습할 일이 있는 경우 논문을 다시 읽어보면 좋을 것으로 판단됨.

5. 이 논문의 한계점이 있다면 무엇이라고 생각하나요?
   A. Tuning할 hyperparameter의 수가 상당히 많은 반면, 논문에서 제안한 Optimal한 구조의 경우, ImageNet, CIFAR10 등의 데이터셋을 기준으로 성능을 뽑았다는 한계가 있음. 즉, 다른 Task에 대해서는 이 논문과 다른 Hyperparameter를 찾아야 한다는 한계가 있음.

6. 본인의 연구에 접목시켜볼 점이 있을지 생각하고 적어보세요.
   A. Diffusion 선행연구

7. 본 Summary를 작성하는 과정에서 생성형AI를 사용했나요?
   A. 아니요

날짜: 2025-07-13

이름: 신준원

Elucidating the Design Space of Diffusion-Based Generative models 07/13

Introduction

이론에 치우친 Diffusion model

↓

Design space 상, 구성 방법에 어려움으로 작용

본 논문은 해당 문제를 해결할 수 있는 방법을 제안함.

① 이론보다는 활용 측면에서, 명확한 구성 (forward, reverse, scheduling ...)
└ System을 부수지 않은채, option 변경 및 최적화 구조.

⇒ 각 구성간의 의존성 파악 + 의존X 상태인 경우 고려 시도.

② 너무 긴 sampling 속도.
⇒ 가장 많이 쓰인 sampler 분석 → sampling 속도의 비약.

③ Score 기반 modeling

①, ②, ③ 을 분석 ⇒ Diffusion 성능 발전에 기여.

denotation

$p_{data}(x)$: data distribution
σ_{data} : standard deviation → noise의 세기.
$p(x;\sigma)$: mollified distribution → 최종 분포 (forward) : $p(x;\sigma_{max}) \sim N(0, \sigma_{max}^2 I)$
(forward → $\sigma_0 < \sigma_1 < \cdots < \sigma_{max} \sim N(0, \sigma_{max}^2 I)$)

$$p_{0t}(x(t)\,|\,x(0)) = N(x(t);\ s(t)x(0),\ s(t)^2\sigma(t)^2 I)$$

$$s(t) = \exp\left(\int_0^t f(\xi)\,d\xi\right),\quad \sigma(t) = \sqrt{\int_0^t \frac{m(\xi)^2}{s(\xi)^2}\,d\xi}$$

→ $\boxed{x = s(t)\cdot x_0 + \sigma(t)}$

ODE formulation ① $f(x,t) = f(t)x$ 꼴, ② $x = x_t$.

↦ by Yang-Song 2021

original ODE (probability flow ODE)

$$dx = \left[f(t)x - \frac{1}{2} g(t)^2 \nabla_x \log p_t(x) \right] dt$$

→ 실제 적분할 때 ⇒ $dx(x)$, marginal distribution

⇒ 고정하면 dx 정의x 아닌, $S(t)$, $\sigma(t)$를 정의 (marginal 계산 위함) ⇒ 더 효율적!

$$p_t(x) = \int_{\mathbb{R}^d} R_{0t}(x|x_0) p_{data}(x_0) dx_0$$

$$= \int_{\mathbb{R}^d} p_{data}(x_0) \cdot \left[\mathcal{N}(x; S(t)x_0, S(t)^2 \cdot \sigma(t)^2 I) \right] dx_0$$

$$= \int_{\mathbb{R}^d} p_{data}(x_0) \left[S(t)^{-d} \mathcal{N}(x/S(t); x_0, \sigma(t)^2 I) \right] dx_0$$

$$= S(t)^{-d} \int_{\mathbb{R}^d} p_{data}(x_0) \mathcal{N}(x/S(t); x_0, \sigma(t)^2 I) \Big] dx_0$$

→ $u = S(t) \cdot x_0$, $x_0 = \frac{u}{S(t)}$, $dx_0 = S(t)^{-d} du$.

$$= S(t)^{-d} \left[p_{data} * \mathcal{N}(0, \sigma(t)^2 I) \right] (x/S(t))$$

⇒ $p(x;\sigma) = p_{data} * \mathcal{N}(0, \sigma(t)^2 I)$ / $p_t(x) = S(t)^{-d} p(x/S(t); \sigma(t))$

⇒ $$\boxed{dx = -\dot{\sigma}(t)\sigma(t) \nabla_x \log p(x; \sigma(t)) dt}$$

Denoising Score matching

$E_{y\sim data} E_{n\sim N(0,\sigma^2 I)}\left[\| D(y+n;\sigma) - y\|_2^2\right]$, $\nabla_x \log p(x;\sigma) = (D(x;\sigma)-x)/\sigma^2$

└▷ 각 noise step × data 에서. ⇒ $D_\theta(x;\sigma)$

① y = Data (image)

② n = noise.

③ $D(x;\sigma)$ = denoiser function

Time-dependent signal scaling

$x = \underbrace{s(t)}_{\text{Scale scheduler}} \cdot \hat{x}$ ⟶ PDF, ODE solution 에서

ODE - solution 구성

$$dx = \left[\frac{\dot{s}(t)}{s(t)} x - s(t)^2 \cdot \dot{\sigma}(t)\cdot\sigma(t) \nabla_x \log p\left(\frac{x}{s(t)};\sigma(t)\right)\right] dt$$

~~Time-dependent signal sca~~

Solution by discretization

각각 정의된 $\nabla_x \log p$ ⟶ ODE - solution 구하기 위함

⊕ 2nd order solver

deterministic sampling (improved)

sample quality 상승 = sampling 위한 연산량 감소.

가설 1) sampling 과정은 확률적이다.

⇒ 앞서 정의한 σ_t 은 DDIM, DDPM 등과 의존성 X.

model (score) 즉, 이로 black box 관계

실험) 3개의 모델 (DDPM++ : VP, NCSN++ : VE, ADM)

⊕ sampler 적용 (original or paper).

⇒ 의존성이 없다면, 성능↑

의존성이 없다면, 3모델 모두에서 성능향상

결과) 3모델의 sample 성능 상승,, ⇒ 증명

⊕ NFE 비교 : σ_t 몇번 호출? ⇒ Sampling 속도를 의미 (비례)

논문 ⇒ improved version 제안서,

3 model 모두에서 성능향상 확인가능

↳ NFE가 적은 경우에도 유의미한 FID 관찰가능

Discretization & high order integrators

Discretization → 연속 형의 SDE – ODE
그러나 실제로는 이산적인 구조.
즉, 이산화해줄 필요가 있음.

→ N↑ 증가 ⇒ error 감소.
error 감소 ⇒ N↑ 증가
trade-off 필요.

~~연속적인 형태~~.

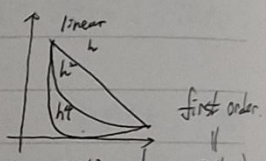
연속 → (근사) → 이산

N step 개념

보정 필요. ⇒ truncation error
(Step - error 축적)

(1 step 개념)
local error : Step 크기와 비례

즉, 근사과정에서는 차이(error)가 발생한다. 이는 보정해줄 필요가 있음.
그러나 해당 연산은 h와 비례 ⇒ 오래걸리더라도 문제X (근사 or 정확도)



→ error 최소.

General 1 : ⊕ Euler's method (first order) $O(h^2)$ 복잡도. (1step $O(h^2) \times \frac{1}{h} \rightarrow O(h)$)
General 2 : Runge-kutta method (high-order) : 매 step에 대한 추가 연산 필요 $O(h^4)$
→ 정확 ∥

proposed method

Heun's 2nd order method (improved Euler, trapezoidal rule)

$h = (t_{n+1} - t_n)$ ∥

Euler → $d_{n+1} = d_n + h \cdot f(d_n, t_n)$

Heun → $d_{n+1} = d_n + \frac{h}{2}[f(d_n, t_n) + f(d_{predict}, t_n + h)]$

⇓

$d_{predict} = d_n + h f(d_n, t_n)$

local → time-step 간격을 조절

(본 Contribution) time-step → noise level $\{\sigma_i\}$ 계획 ⇒ $t_i = \sigma^{-1}(\sigma_i)$

$\sigma_0 = \sigma_{max}$, $\sigma_{N-1} = \sigma_{min}$, $\sigma_{i<N} = (A i + B)^{\rho}$, $\sigma_N = 0$

~~(A) $\rho = 3$, 본 연구 방식~~

$5 < \rho < 10$, sampling quality ↑ ⇒ σ_{min}의 영향이 크다!

min, max
Controller.

Trajectory curvature & noise schedule

현재 ⇒ ODE 정의 with $b(t)$, $s(t)$
↳ 크기 예측 가능 ($\frac{dx}{dt}$).

Best option ⇒ $b(t) = t$ & $s(t) = 1$

$\frac{dx}{dt} = (x - D(x;t))/t$ → x, t 부분 → noise

Sampling (with Euler step $\to t = 0$)

Vε
Vp
$b(t) = t$, $s(t) = 1$
↳ linear 한 이동 = 성능우수.

→ ① trajectory 곡률 측정
⇒ noise 가능. (작은 차이 → 극복)
즉, 최대한 궤적 유지 = Best.

Stochastic sampling

ODE 장점 → forward-reverse 전환가능.
그러나 SDE 대비 성능이 낮음

Q) ODE, SDE 모두 같은 distribution을 따르는데, 그렇다면 stochasticity는 필요한가?

$$dx_{\pm} = -\dot{\sigma}(t)\,\sigma(t)\,\nabla_x \log p(x;\sigma(t))\,dt \pm \beta(t)\,\sigma(t)^2\,\nabla_x \log p(x;\sigma(t))\,dt + \sqrt{2\beta(t)}\,\sigma(t)\,d\omega_t$$

Probability flow ODE / deterministic noise decay / noise injection

Langevin diffusion SDE

denotation

dx_+ ⇒ forward, dx_- ⇒ reverse → $dx_\pm$

Langevin term.
⇒ deterministic score-based denoising term (reverse)
+
stochastic noise term.

$\beta(t) = \frac{\dot{\sigma}(t)}{\sigma(t)}$ ⇒ 최적의 $\beta(t)$ ⇒ 논의

paper → stochastic sampler

구성: 2^{nd} order deterministic ODE integrator + Langevin like

과정 1) ① $\gamma_i = \begin{cases} \min\left(\frac{S_{churn}}{N}, \sqrt{2}-1\right) & \text{if } t_i \in [S_{tmin}, S_{tmax}] \\ 0 & \text{otherwise} \end{cases}$

② $\hat{t}_i = t_i + \gamma_i t_i$ → ③ $\hat{x}_i \leftarrow x_i + \sqrt{\hat{t}_i^2 - t_i^2}\,\epsilon$

과정2) $\hat{x}_i$, $\hat{t}_i$ 기반 → dx 추정 (single step)

Euler step → $x_{i+1} = \hat{x}_i + (t_{i+1} - \hat{t}_i)\,d_i$

↓

2nd ~~step~~ order Correction → return x_N

training

input : x , data : y noise : n.

$x = y + n \Rightarrow n \sim N(0, \sigma^2 I)$ 즉, σ 크기에 영향을 받는 구조임.

따라서, D_θ direct $x \Rightarrow D_\theta$를 구할 수 있는 F_θ 학습.

$$D_\theta(x;\sigma) = x - \sigma F_\theta(\cdot)$$

↳ 문제) large σ인 경우, → control, 값 구하기가 매우 어려움.

본 논문에서 제안하는 구조. → scaler

$$D_\theta(x;\sigma) = c_{skip}(\sigma)x + c_{out}(\sigma)F_\theta(c_{in}(\sigma)x ; c_{noise}(\sigma))$$

- c_{skip} : skip Connection
- c_{out} : 학습 목표
- c_{noise} : noise level

$$E_{\sigma,y,n}[\lambda(\sigma)\| D(y+n;\sigma) - y\|_2^2]$$

$$F_\theta \Rightarrow E_{\sigma,y,n}\left[\underbrace{\lambda(\sigma)c_{out}(\sigma)^2}_{\text{effective weight}} \| \underbrace{F_\theta(c_{in}(\sigma)\cdot(y+n) ; c_{noise}(\sigma))}_{\text{network output}} - \underbrace{\frac{1}{c_{out}(\sigma)}(y - c_{skip}(\sigma)\cdot(y+n))}_{\text{training target}} \|_2^2\right]$$

- $\lambda(\sigma)$: $1/c_{out}(\sigma)^2$ $\Rightarrow$ effective weight = 1

- noise level $\sigma \sim p_{train}(\sigma)$ ※

↳ 너무 높은 경우 / 너무 낮은 경우] → 문제 가능 → normal distribution. 사용

Augmentation

→ overfitting 방지 · 성능 향상

본 프로젝트 GAN의 Augmentation pipeline 사용