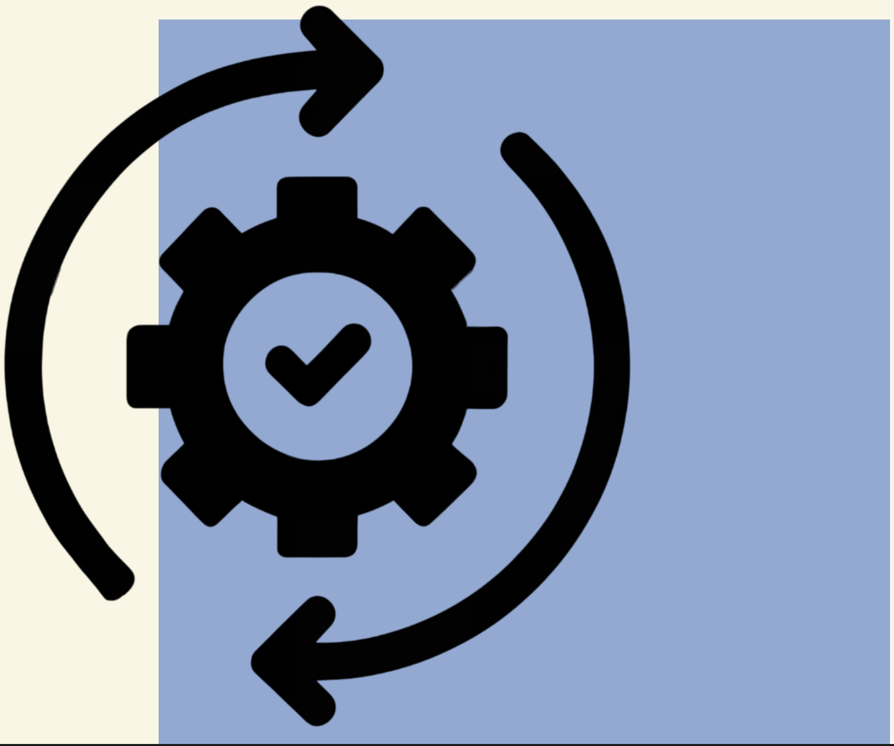
Эффективные системы машинного обучения



Лекция 10

Представление диффузионных моделей. Решения ОДУ

Преподаватель

Оганов Александр

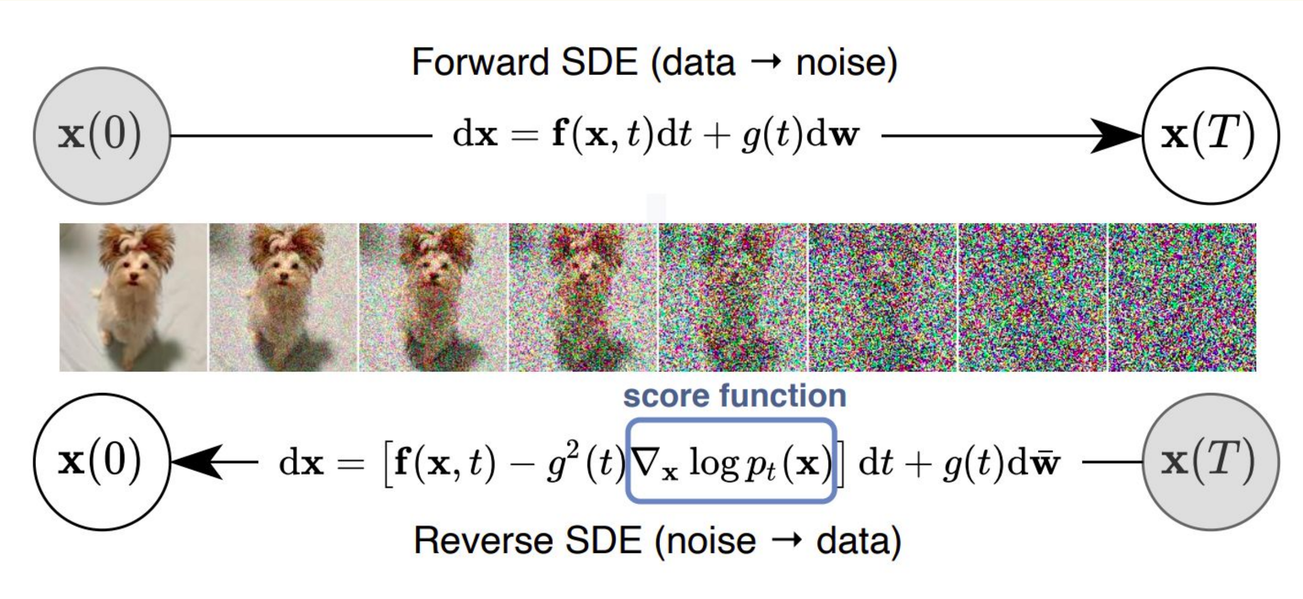
29 ноября 2024

ВМК МГУ

Напоминание

Из чего стоят диффузионные модели?

- Процесс разрушения (прямой)
- Процесс генерации (обратный)



Непрерывные модели

Далее мы будем рассматривать только модели с непрерывным временем. В качестве процесса зашумления мы можем взять VP-SDE или VE-SDE

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_t \Big\{ \lambda(t) \mathbb{E}_{\mathbf{x}(0)} \mathbb{E}_{\mathbf{x}(t)|\mathbf{x}(0)} \Big[\big\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0)) \big\|_2^2 \Big] \Big\}$$

$$p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0)) = \begin{cases} \mathcal{N}\big(\mathbf{x}(t); \mathbf{x}(0), [\sigma^2(t) - \sigma^2(0)]\mathbf{I}\big), & \text{(VE SDE)} \\ \mathcal{N}\big(\mathbf{x}(t); \mathbf{x}(0) e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, \mathbf{I} - \mathbf{I} e^{-\int_0^t \beta(s)\mathrm{d}s}\big) & \text{(VP SDE)} \\ \mathcal{N}\big(\mathbf{x}(t); \mathbf{x}(0) e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, [1 - e^{-\int_0^t \beta(s)\mathrm{d}s}]^2 \mathbf{I}\big) & \text{(sub-VP SDE)} \end{cases}$$

Параметризация

Можно учить:

- предсказание шума
- score функцию
- предсказание чистого объекта x_0

$$p_{0t}(x(t) \mid x(0)) := \mathcal{N}(x(t) \mid x(0)\sqrt{\bar{\alpha}_t}, (1 - \bar{\alpha}_t)I)$$

$$\log p_{0t}(x(t) \mid x(0)) = \text{const} - \frac{1}{2}\frac{(x(t) - \sqrt{\bar{\alpha}_t}x(0))^2}{1 - \bar{\alpha}_t}$$

$$\nabla_x \log p_{0t}(x(t) \mid x(0)) = -\frac{x(t) - \sqrt{\bar{\alpha}_t}x(0)}{1 - \bar{\alpha}_t}$$

$$\nabla_x \log p_{0t}(x(t) \mid x(0)) = -\frac{\epsilon(x(t), x(0))}{\sqrt{1 - \bar{\alpha}_t}}$$

Как генерировать?

Для генерации необходимо решить либо обратный СДУ, либо соответствующее ОДУ

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$$

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]\mathrm{d}t$$

Открытые вопросы

1) Как решать ОДУ/СДУ учитывая условия задачи?
2) Как же правильно задавать лосс?
3) Как выбрать f(x,t) и g(t)?
4) Как не сломать себе голову, когда читаешь статьи и увидеть единый формализм

Как повысить качество?

Обычно, новые модели учат достаточно плохо и неправильно: выбирают не ту архитектуру, оптимизируют не тот лосс и тд.

Какие гипер-параметры у нас есть?

- Зашумление
- Вес в лоссе
- Архитектура

Как их выбирать?

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_t \Big\{ \lambda(t) \mathbb{E}_{\mathbf{x}(0)} \mathbb{E}_{\mathbf{x}(t)|\mathbf{x}(0)} \Big[\big\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0)) \big\|_2^2 \Big] \Big\}$$

Ответ на все вопросы

Для начала, стоит подумать, как записать все в едином виде, поэтому обратимся к статье Elucidating the Design Space of Diffusion-Based Generative Models (EDM) (https://arxiv.org/abs/2206.00364).

1) Как влияет СДУ на переходные вероятности?

$$\mathrm{d}\boldsymbol{x} = f(t)\,\boldsymbol{x}\,\mathrm{d}t + g(t)\,\mathrm{d}\omega_t$$

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

$$p_{0t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(0)\big) = \mathcal{N}\big(\boldsymbol{x}(t);\; s(t)\,\boldsymbol{x}(0),\; s(t)^2\,\sigma(t)^2\,\mathbf{I}\big)$$

Обратный процесс

Зная прямой процесс, мы можем найти и обратный. Попробуем выписать обратный процесс в общем виде используя определения f(t) и g(t)

$$f(t) \quad = \quad \dot{s}(t)/s(t) \qquad g(t) \quad = \quad s(t)\,\sqrt{2\,\dot{\sigma}(t)\,\sigma(t)}$$

$$\begin{aligned}
\mathrm{d}\boldsymbol{x} \quad &= \quad \left[[f(t)]\,\boldsymbol{x} - \tfrac{1}{2}\,[g(t)]^2\,\nabla_{\boldsymbol{x}} \log p\big(\boldsymbol{x}/s(t);\sigma(t)\big)\right]\,\mathrm{d}t && (35)\\
&= \quad \left[\left[\dot{s}(t)/s(t)\right]\,\boldsymbol{x} - \tfrac{1}{2}\left[s(t)\sqrt{2\,\dot{\sigma}(t)\,\sigma(t)}\right]^2\,\nabla_{\boldsymbol{x}} \log p\big(\boldsymbol{x}/s(t);\sigma(t)\big)\right]\,\mathrm{d}t && (36)\\
&= \quad \left[\left[\dot{s}(t)/s(t)\right]\,\boldsymbol{x} - \tfrac{1}{2}\left[2\,s(t)^2\,\dot{\sigma}(t)\,\sigma(t)\right]\,\nabla_{\boldsymbol{x}} \log p\big(\boldsymbol{x}/s(t);\sigma(t)\big)\right]\,\mathrm{d}t && (37)\\
&= \quad \left[\frac{\dot{s}(t)}{s(t)}\,\boldsymbol{x} - s(t)^2\,\dot{\sigma}(t)\,\sigma(t)\,\nabla_{\boldsymbol{x}} \log p\left(\frac{\boldsymbol{x}}{s(t)};\sigma(t)\right)\right]\,\mathrm{d}t. && (38)
\end{aligned}$$

Итог

Нам достаточно задать s(t) и σ(t) и мы сможем получить рассчитать переходные вероятности и обратный процесс!

Если мы рассматриваем VE-SDE, то s(t) = 1 и получаем:

$$\mathrm{d}\boldsymbol{x} = -\dot{\sigma}(t)\,\sigma(t)\,\nabla_{\boldsymbol{x}} \log p\big(\boldsymbol{x};\sigma(t)\big)\,\mathrm{d}t$$

Как задать все процессы?

		VP [49]	VE [49]	iDDPM [37] + DDIM [47]	Ours (“EDM”)
Sampling (Section 3)					
ODE solver		Euler	Euler	Euler	2[nd] order Heun
Time steps	$t_{i<N}$	$1+\frac{i}{N-1}(\epsilon_s - 1)$	$\sigma_{\max}^2\left(\sigma_{\min}^2/\sigma_{\max}^2\right)^{\frac{i}{N-1}}$	$u_{\lfloor j_0+\frac{M-1-j_0}{N-1}i+\frac{1}{2}\rfloor}$, where $u_M = 0$, $u_{j-1} = \sqrt{\frac{u_j^2+1}{\max(\bar{\alpha}_{j-1}/\bar{\alpha}_j, C_1)}-1}$	$\left(\sigma_{\max}{}^{\frac{1}{\rho}} + \frac{i}{N-1}\left(\sigma_{\min}{}^{\frac{1}{\rho}} - \sigma_{\max}{}^{\frac{1}{\rho}}\right)\right)^{\rho}$
Schedule	$\sigma(t)$	$\sqrt{e^{\frac{1}{2}\beta_d t^2+\beta_{\min}t}-1}$	$\sqrt{t}$	t	t
Scaling	$s(t)$	$1/\sqrt{e^{\frac{1}{2}\beta_d t^2+\beta_{\min}t}}$	1	1	1
Network and preconditioning (Section 5)					
Architecture of F_θ		DDPM++	NCSN++	DDPM	(any)
Skip scaling	$c_{\text{skip}}(\sigma)$	1	1	1	$\sigma_{\text{data}}^2/\left(\sigma^2+\sigma_{\text{data}}^2\right)$
Output scaling	$c_{\text{out}}(\sigma)$	$-\sigma$	σ	$-\sigma$	$\sigma \cdot \sigma_{\text{data}}/\sqrt{\sigma_{\text{data}}^2+\sigma^2}$
Input scaling	$c_{\text{in}}(\sigma)$	$1/\sqrt{\sigma^2+1}$	1	$1/\sqrt{\sigma^2+1}$	$1/\sqrt{\sigma^2+\sigma_{\text{data}}^2}$
Noise cond.	$c_{\text{noise}}(\sigma)$	$(M-1)\,\sigma^{-1}(\sigma)$	$\ln(\frac{1}{2}\sigma)$	$M-1-\arg\min_j \lvert u_j - \sigma\rvert$	$\frac{1}{4}\ln(\sigma)$
Training (Section 5)					
Noise distribution		$\sigma^{-1}(\sigma) \sim \mathcal{U}(\epsilon_t, 1)$	$\ln(\sigma) \sim \mathcal{U}(\ln(\sigma_{\min}), \ln(\sigma_{\max}))$	$\sigma = u_j,\ \ j \sim \mathcal{U}\{0, M-1\}$	$\ln(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$
Loss weighting	$\lambda(\sigma)$	$1/\sigma^2$	$1/\sigma^2$	$1/\sigma^2$ (note: *)	$\left(\sigma^2+\sigma_{\text{data}}^2\right)/(\sigma\cdot\sigma_{\text{data}})^2$
Parameters		$\beta_d = 19.9, \beta_{\min} = 0.1$	$\sigma_{\min} = 0.02$	$\bar{\alpha}_j = \sin^2\left(\frac{\pi}{2}\frac{j}{M(C_2+1)}\right)$	$\sigma_{\min} = 0.002, \sigma_{\max} = 80$
		$\epsilon_s = 10^{-3}, \epsilon_t = 10^{-5}$	$\sigma_{\max} = 100$	$C_1 = 0.001, C_2 = 0.008$	$\sigma_{\text{data}} = 0.5, \rho = 7$
		$M = 1000$		$M = 1000, j_0 = 8^{\dagger}$	$P_{\text{mean}} = -1.2, P_{\text{std}} = 1.2$

* iDDPM also employs a second loss term L_{vlb} † In our tests, $j_0 = 8$ yielded better FID than $j_0 = 0$ used by iDDPM

Параметризация

Мы знаем, что нейронные сети очень важно хорошо параметризовать. Например, если данные и таргет нормализованные, то модель будет проще учить.

Далее мы считаем, что учим диффузионную модель на предсказание x_0 в следующем виде:

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

Обучаем минимизируя:

$$\mathbb{E}_{\sigma,\boldsymbol{y},\boldsymbol{n}}\left[\lambda(\sigma)\,\|D(\boldsymbol{y}+\boldsymbol{n};\sigma)-\boldsymbol{y}\|_2^2\right]$$

$$\sigma \sim p_{\text{train}},\ \boldsymbol{y} \sim p_{\text{data}},\ \text{and}\ \boldsymbol{n} \sim \mathcal{N}(\mathbf{0},\sigma^2\mathbf{I})$$

Как учить?

Все гипер параметры авторы либо находили из перебора (подробнее на семинаре) или из теории.

$$\mathbb{E}_{\sigma,\boldsymbol{y},\boldsymbol{n}}\left[\lambda(\sigma)\,\|D(\boldsymbol{y}+\boldsymbol{n};\sigma)-\boldsymbol{y}\|_2^2\right]$$

$$\sigma \sim p_{\text{train}},\ \boldsymbol{y} \sim p_{\text{data}},\ \text{and}\ \boldsymbol{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$$

$$\mathbb{E}_{\sigma,\boldsymbol{y},\boldsymbol{n}}\Big[\underbrace{\lambda(\sigma)\,c_{\text{out}}(\sigma)^2}_{\text{effective weight}}\Big\|\underbrace{F_\theta\big(c_{\text{in}}(\sigma)\cdot(\boldsymbol{y}+\boldsymbol{n});c_{\text{noise}}(\sigma)\big)}_{\text{network output}}-\underbrace{\tfrac{1}{c_{\text{out}}(\sigma)}\big(\boldsymbol{y}-c_{\text{skip}}(\sigma)\cdot(\boldsymbol{y}+\boldsymbol{n})\big)}_{\text{effective training target}}\Big\|_2^2\Big]$$

$$\lambda(\sigma) = 1/c_{\text{out}}(\sigma)^2$$

Итоги

Мы научились:

- задавать зашумления в общем виде
- выбирать правильную параметризацию
- эффективно обучать

Мы не умеем:

- Правильно и эффективно решать ОДУ/СДУ

Как инферить?

Мы умеем круто обучать диффузионные модели, но все еще не умеем быстро генерировать качественные изображения.

Есть следующие подходы:
- Дистилляции (https://arxiv.org/abs/2303.01469)
- Кэширования
- Квантизации
- Солверы ОДУ/СДУ
- есть и другие, но все основные направления современной науке описаны выше

Солверы

Одним из современных направлений в диффузионных моделях это изучение, оптимизация и разработка солверов.

Что такое солвер?
Солвер – метод/алгоритм, который решает конкретную численную задачу. Например, метод Эйлера для решения ОДУ

Зачем нужны солверы?
Чем более точное решение ОДУ/СДУ мы получим, тем лучше будут изображения, при этом часто методы 2 порядка при меньшем числе шагов получают решения сильно лучше, чем методы 1 порядка.

Солверы

Уже существует много работ о солверах, но мы поговорим про основные:

- DDIM (метод придуманный для дискретных моделей)
- DPM-Solver (солвер для решения ОДУ, которые возникают в диффузионках) (https://arxiv.org/abs/2206.00927)

Как решать ОДУ?

В статье EDM также предложен метод решения ОДУ 2 порядка Heun, но мы посмотрим чуть с другой стороны.

Algorithm 1 Deterministic sampling using Heun's 2^{nd} order method with arbitrary $\sigma(t)$ and $s(t)$.

procedure HEUNSAMPLER($D_\theta(\boldsymbol{x};\sigma),\ \sigma(t),\ s(t),\ t_{i\in\{0,\dots,N\}}$)

  sample $\boldsymbol{x}_0 \sim \mathcal{N}\big(\mathbf{0},\ \sigma^2(t_0)\, s^2(t_0)\,\mathbf{I}\big)$ ▷ Generate initial sample at t_0

  for $i \in \{0,\dots,N-1\}$ **do** ▷ Solve Eq. 4 over N time steps

    $\boldsymbol{d}_i \leftarrow \left(\frac{\dot\sigma(t_i)}{\sigma(t_i)} + \frac{\dot s(t_i)}{s(t_i)}\right)\boldsymbol{x}_i - \frac{\dot\sigma(t_i)s(t_i)}{\sigma(t_i)} D_\theta\left(\frac{\boldsymbol{x}_i}{s(t_i)};\sigma(t_i)\right)$ ▷ Evaluate $\mathrm{d}\boldsymbol{x}/\mathrm{d}t$ at t_i

    $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + (t_{i+1}-t_i)\boldsymbol{d}_i$ ▷ Take Euler step from t_i to t_{i+1}

    if $\sigma(t_{i+1}) \neq 0$ **then** ▷ Apply 2^{nd} order correction unless σ goes to zero

      $\boldsymbol{d}'_i \leftarrow \left(\frac{\dot\sigma(t_{i+1})}{\sigma(t_{i+1})} + \frac{\dot s(t_{i+1})}{s(t_{i+1})}\right)\boldsymbol{x}_{i+1} - \frac{\dot\sigma(t_{i+1})s(t_{i+1})}{\sigma(t_{i+1})} D_\theta\left(\frac{\boldsymbol{x}_{i+1}}{s(t_{i+1})};\sigma(t_{i+1})\right)$ ▷ Eval. $\mathrm{d}\boldsymbol{x}/\mathrm{d}t$ at t_{i+1}

      $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + (t_{i+1}-t_i)\left(\frac{1}{2}\boldsymbol{d}_i + \frac{1}{2}\boldsymbol{d}'_i\right)$ ▷ Explicit trapezoidal rule at t_{i+1}

  return $\boldsymbol{x}_N$ ▷ Return noise-free sample at t_N

Солверы или как решать ОДУ

Forward process

$$q(x_t \mid x_0) = \mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2 I)$$

$$dx_t = f(t) x_t dt + g(t) dW_t$$

$$f(t) = \frac{\dot{\alpha}_t}{\alpha_t}, \quad g^2(t) = 2\sigma_t \dot{\sigma}_t - \frac{2\sigma_t^2 \dot{\alpha}_t}{\alpha_t}$$

Backward process

$$dx_t = (f(t) x_t - g^2(t) \nabla_x \log p(x_t, t)) dt + g(t) dW_t$$

$$\frac{dx_t}{dt} = f(t) x_t - \frac{1}{2} g^2(t) \nabla_x \log p(x_t, t)$$

Качество генерации

Мы знаем: чем точнее решаем ОДУ/СДУ, тем выше будет качество изображения (следует из теории). Вопрос ли в том, что решать?

Качество генерации

Мы знаем: чем точнее решаем ОДУ/СДУ, тем выше будет качество изображения (следует из теории). Вопрос ли в том, что решать?

Ответ: Будем решать вариант с ОДУ, а не СДУ, так как при больших шагах дисперсия при dW_t слишком высокая, что будет мешать сходимости. Запустим на ОДУ численный метод и все готово, например Рунге–Кутта. Правда же???

Напоминание с курса ЧМов

Пусть задан ОДУ

x’(t) - f(t) x(t) = F(x(t), t, \theta)

Обычное такие дифференциальные уравнения называют линейными с переменными коэффициентами. Для них существует аналитическое решение, например, с помощью метода вариации постоянной

$$\boldsymbol{x}_t = e^{\int_s^t f(\tau)\mathrm{d}\tau}\boldsymbol{x}_s + \int_s^t \left(e^{\int_\tau^t f(r)\mathrm{d}r}\frac{g^2(\tau)}{2\sigma_\tau}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau) \right) \mathrm{d}\tau$$

Что мы получили?

Если посмотрим внимательней, то некоторые коэффициенты мы можем рассчитать аналитически. То есть использовать знания о задаче, а не кидать все в готовый солвер! (или же…)
Попробуем упростить задачу и уменьшить число сложных букв.

$$\boldsymbol{x}_t = e^{\int_s^t f(\tau)\mathrm{d}\tau}\boldsymbol{x}_s + \int_s^t \left(e^{\int_\tau^t f(r)\mathrm{d}r} \frac{g^2(\tau)}{2\sigma_\tau} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau) \right) \mathrm{d}\tau$$

Полезные формулы

$$logSNR = \log \frac{\alpha_t^2}{\sigma_t^2}$$

$$\lambda_t := \log \frac{\alpha_t}{\sigma_t}, \quad \frac{d\lambda_t}{dt} = \frac{\dot{\alpha}_t}{\alpha_t} - \frac{\dot{\sigma}_t}{\sigma_t}$$

$$\log p_\theta(x_t, t) = -\frac{\epsilon_\theta(x_t, t)}{\sigma_t}$$

Чуть упростим

$$\boldsymbol{x}_t = e^{\int_s^t f(\tau)\mathrm{d}\tau}\boldsymbol{x}_s + \int_s^t \left(e^{\int_\tau^t f(r)\mathrm{d}r} \frac{g^2(\tau)}{2\sigma_\tau} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau) \right) \mathrm{d}\tau$$

$$\int f(t)dt = \int \frac{d \log \alpha_t}{dt} dt = \log \alpha_t + C$$

$$g^2(t) = 2\sigma_t \dot{\sigma}_t - \frac{2\sigma_t^2 \dot{\alpha}_t}{\alpha_t} = 2\sigma_t^2 (\frac{\dot{\sigma}_t}{\sigma_t} - \frac{\dot{\alpha}_t}{\alpha_t}) = -2\sigma_t^2 \frac{d\lambda_t}{dt}$$

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \alpha_t \int_s^t \left(\frac{\mathrm{d}\lambda_\tau}{\mathrm{d}\tau} \right) \frac{\sigma_\tau}{\alpha_\tau} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau)\mathrm{d}\tau$$

Еще проще

Сделаем замену переменных

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \alpha_t \int_s^t \left(\frac{\mathrm{d}\lambda_\tau}{\mathrm{d}\tau}\right) \frac{\sigma_\tau}{\alpha_\tau} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau)\mathrm{d}\tau$$

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \alpha_t \int_{\lambda_s}^{\lambda_t} e^{-\lambda} \hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)\mathrm{d}\lambda$$

Что получили?

Мы аналитически можем подсчитать коэффициент перед линейной частью, что раньше мы кидали в солвер не думая. Причем перед этим коэффициентом у нас e^{\int …}, так что ошибка накапливалась очень весомая.

Теперь мы свели задачу к оценке интеграла, который известен в литературе как exponentially weighted integral и сводиться к exponential integrators

$$\int_{\lambda_s}^{\lambda_t} e^{-\lambda} \hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda) \mathrm{d}\lambda$$

Что дает Тейлор

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i}\int_{\lambda_{t_{i-1}}}^{\lambda_{t_i}} e^{-\lambda}\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda,\lambda)\mathrm{d}\lambda$$

$$\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda,\lambda) = \sum_{n=0}^{k-1}\frac{(\lambda-\lambda_{t_{i-1}})^n}{n!}\hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}},\lambda_{t_{i-1}}) + \mathcal{O}((\lambda-\lambda_{t_{i-1}})^k)$$

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i}\sum_{n=0}^{k-1}\hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}},\lambda_{t_{i-1}})\int_{\lambda_{t_{i-1}}}^{\lambda_{t_i}} e^{-\lambda}\frac{(\lambda-\lambda_{t_{i-1}})^n}{n!}\mathrm{d}\lambda + \mathcal{O}(h_i^{k+1})$$

Производные...

Пока забудем о производных и возьмем k = 1 тогда, получим:

$$\tilde{\boldsymbol{x}}_{t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}(e^{h_i} - 1)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}), \quad \text{where } h_i = \lambda_{t_i} - \lambda_{t_{i-1}}$$

Назовем метод DPM-Solver-1, порядок сходимости будет O(max_i h_i), то есть чем меньше h_i, тем лучше сходимость.
Идея: тогда разумнее взять h_i = const, получим расписание logSNR

Где-то это уже было... DDIM и DPM-Solver-1

Denoising Diffusion Implicit Models (DDIM) [19] design a deterministic method for fast sampling from DPMs. For two adjacent time steps t_{i-1} and t_i, assume that we have a solution $\tilde{\boldsymbol{x}}_{t_{i-1}}$ at time t_{i-1}, then a single step of DDIM from time t_{i-1} to time t_i is

$$\tilde{\boldsymbol{x}}_{t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i}\left(\frac{\sigma_{t_{i-1}}}{\alpha_{t_{i-1}}} - \frac{\sigma_{t_i}}{\alpha_{t_i}}\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}). \tag{4.1}$$

Although motivated by entirely different perspectives, we show that the updates of DPM-Solver-1 and Denoising Diffusion Implicit Models (DDIM) [19] are identical. By the definition of λ, we have $\frac{\sigma_{t_{i-1}}}{\alpha_{t_{i-1}}} = e^{-\lambda_{t_{i-1}}}$ and $\frac{\sigma_{t_i}}{\alpha_{t_i}} = e^{-\lambda_{t_i}}$. Plugging these and $h_i = \lambda_{t_i} - \lambda_{t_{i-1}}$ to Eq. (4.1) results in exactly a step of DPM-Solver-1 in Eq. (3.7). However, the semi-linear ODE formulation of DPM-Solver allows for principled generalization to higher-order solvers and convergence order analysis.

Мы не умеем считать производные...

Когда речь идет о численных методах и возникают производные высокого порядка, то значит пора переходить к аппроксимациям)))

Причем чем больше производных мы приблизим, тем выше порядок метода

Зоопарк

DPM-Solver-1
DPM-Solver-2
DPM-Solver-2 (general version)
DPM-Solver-3
DPM-Solver-12
DPM-Solver-23
DPM-Solver-fast

Algorithm 1 DPM-Solver-2.

Require: initial value $\boldsymbol{x}_T$, time steps $\{t_i\}_{i=0}^{M}$, model $\boldsymbol{\epsilon}_\theta$

$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$
for $i \leftarrow 1$ to M **do**
  $s_i \leftarrow t_\lambda\left(\frac{\lambda_{t_{i-1}}+\lambda_{t_i}}{2}\right)$
  $\boldsymbol{u}_i \leftarrow \frac{\alpha_{s_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{s_i}\left(e^{\frac{h_i}{2}}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
  $\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}\left(e^{h_i}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{u}_i, s_i)$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

Algorithm 2 DPM-Solver-3.

Require: initial value $\boldsymbol{x}_T$, time steps $\{t_i\}_{i=0}^{M}$, model $\boldsymbol{\epsilon}_\theta$

$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T, r_1 \leftarrow \frac{1}{3}, r_2 \leftarrow \frac{2}{3}$
for $i \leftarrow 1$ to M **do**
  $s_{2i-1} \leftarrow t_\lambda\left(\lambda_{t_{i-1}} + r_1 h_i\right), \quad s_{2i} \leftarrow t_\lambda\left(\lambda_{t_{i-1}} + r_2 h_i\right)$
  $\boldsymbol{u}_{2i-1} \leftarrow \frac{\alpha_{s_{2i-1}}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{s_{2i-1}}\left(e^{r_1 h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
  $\boldsymbol{D}_{2i-1} \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{u}_{2i-1}, s_{2i-1}) - \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
  $\boldsymbol{u}_{2i} \leftarrow \frac{\alpha_{s_{2i}}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{s_{2i}}\left(e^{r_2 h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \frac{\sigma_{s_{2i}} r_2}{r_1}\left(\frac{e^{r_2 h_i}-1}{r_2 h_i}-1\right)\boldsymbol{D}_{2i-1}$
  $\boldsymbol{D}_{2i} \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{u}_{2i}, s_{2i}) - \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
  $\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}\left(e^{h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \frac{\sigma_{t_i}}{r_2}\left(\frac{e^{h_i}-1}{h}-1\right)\boldsymbol{D}_{2i}$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

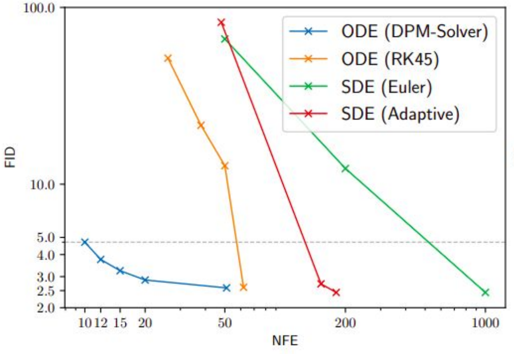
Explicit Exponential Runge Kutta

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \alpha \boldsymbol{x}_t + \boldsymbol{N}(\boldsymbol{x}_t, t)$$

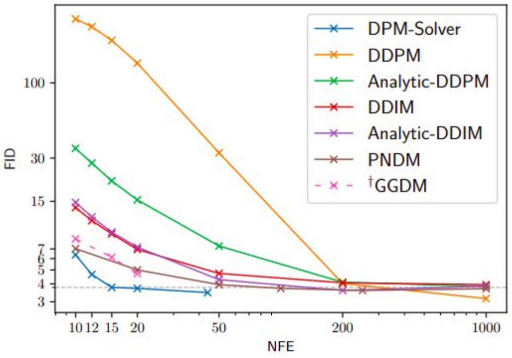
$$\boldsymbol{x}_{t+h} = e^{\alpha h}\boldsymbol{x}_t + e^{\alpha h}\int_0^h e^{-\alpha\tau}\boldsymbol{N}(\boldsymbol{x}_{t+\tau}, t+\tau)\mathrm{d}\tau$$

for approximating the same integral with $\alpha = 1$ and $\boldsymbol{N} = \tilde{\boldsymbol{\epsilon}}_\theta$. However, DPM-Solver is different from the expRK methods, because their linear term $e^{\alpha h}\boldsymbol{x}_t$ is different from our linear term $\frac{\alpha_{t+h}}{\alpha_t}\boldsymbol{x}_t$. In summary, DPM-Solver is inspired by the same technique of expRK for deriving high-order approximations of the exponentially weighted integral, but the formulation of DPM-Solver is different from expRK, and DPM-Solver is customized for the specific formulation of diffusion ODEs.

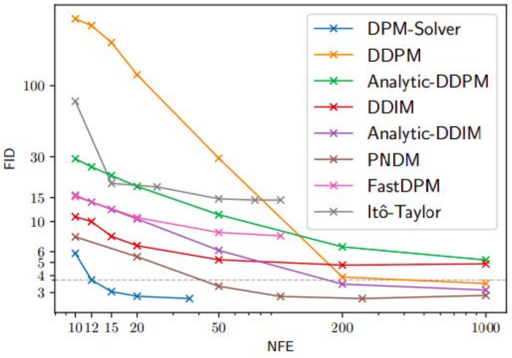
Результаты



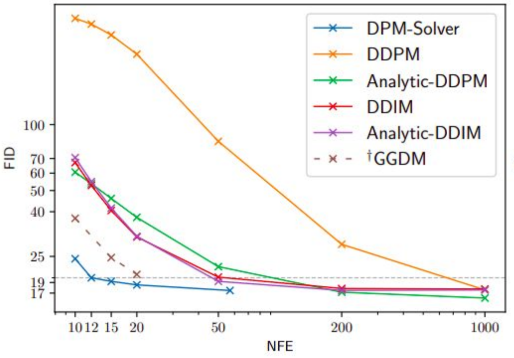
(a) CIFAR-10 (continuous)



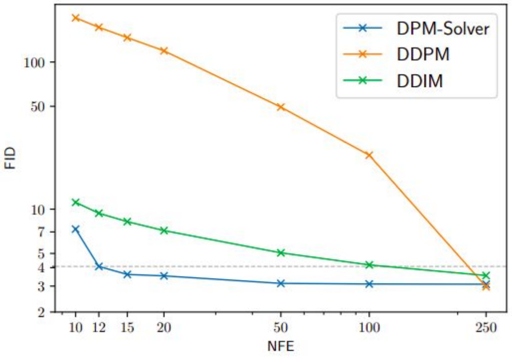
(b) CIFAR-10 (discrete)



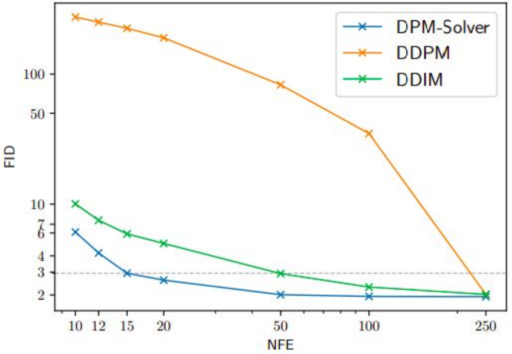
(c) CelebA 64x64 (discrete)



(d) ImageNet 64x64 (discrete)



(e) ImageNet 128x128 (discrete)



(f) LSUN bedroom 256x256 (discrete)

Итоги

Мы умеем:

- Эффективно учить и параметризовать диффузионные модели
- Использовать более умные солверы, которые учитывают условия задачи
- Получили качественную генерацию в 10-20 шагов
- Умеем использовать одну диффузионную модель для разных параметризаций

Мы не умеем:

...

Заключения теории

На самом деле, мы многое еще не умеем и не знаем.
Есть подходы основанные на оптимальном транспорте и flow matching лоссами.
Есть подходы со спрямленнием траектории, но это отдельные.
Много разной математики и эвристик, много с генерации дискретных объектов.
Есть еще много чего! Но нам не хватит и 10 лекций, поэтому пора чуть остановиться.

Все желающие могут пройти отдельный курс про диффузионные модели "Генеративные модели на основе ODE и SDE" от Ракитина Дениса Романовича