Анализ статьи Denoising Diffusion Probabilistic Models

Королев Кирилл

10 ноября 2022 г.

Аннотация Данная работа посвящена устройству диффузионных моделей, которые показали отличные результаты в синтезе изображений в последнее время, и опирается на статью **Denoising Diffusion Probabilistic Models**. В начале будет мотивация для возникновения диффузионных моделей на примере вариационных автокодировщиков и как идеи оттуда легли в основу DDPM. Далее представлен непосредственно анализ самой статьи, выводы из экспериментов и достоинства и недостатки данной модели.

1 Мотивация и VAE

Целью генеративных моделей является поиск распределения $p(x \mid \theta)$, с помощью которого сэмплируются новые данные. Обычно точно найти распределение невозможно, но можно его аппроксимировать. Такие модели называют **латентными**, так как имеют латентные переменные z - представления наших данных обычно в пространстве более низкой размерности.

То есть наша модель описывается совместным распределением $p(x, z \mid \theta) = p(x \mid z, \theta)\ p(z)$, где $p(z)$ моделирует поведение латентных переменных, а $p(x \mid z, \theta)$ определяет как представления связаны с настоящими данными.

Как обычно будем пытаться решать задачу максимизации правдоподобия

$$\log p(x \mid \theta) \to \max_{\theta}$$

Так как $p(x \mid \theta) = \int p(x, z \mid \theta) dz$, решить аналитически данную задачу не представляется возможным. Поэтому можно использовать итеративные методы, например, градиентный спуск. Переписав градиент логарифма правдоподобия, получим

$$\nabla \log p(x \mid \theta) = \frac{\nabla p(x \mid \theta)}{p(x \mid \theta)} = \frac{\nabla \int p(x, z \mid \theta) dz}{p(x \mid \theta)} = \int \frac{\nabla p(x, z \mid \theta)}{p(x \mid \theta)} \frac{p(x, z \mid \theta)}{p(x, z \mid \theta)} dz =$$

$$= \int p(z \mid x, \theta) \nabla \log p(x, z \mid \theta) dz$$

Таким образом, чтобы посчитать градиент нам нужно знать постериорную вероятность $p(z \mid x, \theta)$, которая моделирует представления по изначальным данным. Будем аппроксимировать ее неким распределением $q(z \mid x, \phi)$. Для его поиска помогает следующая нижняя оценка, называемая также **ELBO**

$$\log p(x \mid \theta) = \log \int p(x, z \mid \theta) dz = \log \int p(x, z \mid \theta) \frac{q(z \mid x, \phi)}{q(z \mid x, \phi)} dz =$$
$$= \log \mathbb{E}_{q(z|x,\phi)} \left[\frac{p(x, z \mid \theta)}{q(z \mid x, \phi)} \right] \geq \mathbb{E}_{q(z|x,\phi)} \left(\log \frac{p(x, z \mid \theta)}{q(z \mid x, \phi)} \right) = L_{\theta,\phi}(x)$$

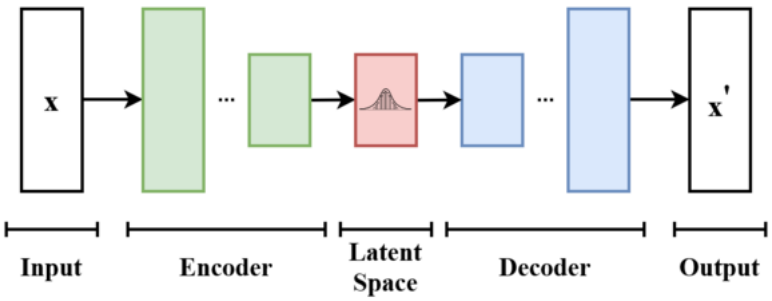
Записав KL-дивергенцию между апостериорными распределениями, можно заметить, что

$$KL(q_\phi(z \mid x) \mid\mid p(z \mid x, \theta)) = \mathbb{E}_{q(z|x,\phi)} \left(\log \frac{q(z \mid x, \phi)}{p(z \mid x, \theta)} \right) =$$
$$\mathbb{E}_{q(z|x,\phi)} \left(\log \frac{q(z \mid x, \phi)}{p(x, z \mid \theta)} p(x \mid \theta) \right) = -L_{\theta,\phi}(x) + \mathbb{E}_{q(z|x,\phi)} (\log p(x \mid \theta)) =$$
$$-L_{\theta,\phi}(x) + \log p(x \mid \theta)$$

Следовательно нижнюю оценку можно выразить следующим образом

$$L_{\theta,\phi}(x) = \log p(x \mid \theta) - KL(q_\phi(z \mid x) \mid\mid p(z \mid x, \theta))$$

Максимизируя ELBO, мы увеличиваем логарифм правдоподобия и все лучше приближаем в смысле KL-дивергенции $p(z \mid x, \theta)$ с помощью $q(z \mid x, \phi)$.



Данный подход называется вариационными автокодировщиками - **VAE** и работает следующим образом. Вход кодируется как распределение в латентном пространстве, из него сэмплируется точка, которая впоследствии декодируется, после чего считается reconstruction error - насколько хорошо изначальные данные были реконструированы.

Однако известная проблема заключается в том, что во время обучения мы не можем покрыть все распределение $p(z)$ с помощью $q(z_i \mid x_i, \phi)$. Поэтому пришла следующая идея.

2 Диффузионные модели

2.1 Введение

Мы опять находимся в сеттинге латентных моделей, но важным отличием будет то, что роль апостериорного распределения будет играть марковская цепь.

Пусть к нам на вход пришел $x_0 \sim q(x_0)$. Будем последовательно зашумлять данные по следующей схеме

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon \quad \epsilon \sim N(0, I)$$

Тогда условное распределение равно

$$q(x_t \mid x_{t-1}) = N(x_t;\ \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

Отнормировав x_0 таким образом, что $D(x_0) = 1$, заметим, что для любого момента времени $D(x_t) = 1$ при малых β_t, что также известно как variance preserving diffusion.

Замечательным является тот факт, что мы можем выразить x_t в закрытой форме через x_0. Это можно показать по индукции.

$$x_t = \sqrt{(1-\beta_1)\dots(1-\beta_t)}x_0 + \sqrt{1-(1-\beta_1)\dots(1-\beta_t)}\epsilon$$

Обозначив за $\alpha_t = 1 - \beta_t$ и $\hat{\alpha_t} = \prod_{j=1}^{t} \alpha_j$, получим распределение

$$q(x_t \mid x_0) = N(x_t;\ \sqrt{\hat{\alpha_t}}x_0, (1-\hat{\alpha_t})I) \to N(x_t \mid 0, I), \quad t \to \infty$$

при довольно малых β_t, которые задают, так называемое, вариционное расписание. То есть в итоге мы получаем чистый равномерный шум.

Данная схема называется прямым или **диффузионным процессом**. И задает марковскую цепь, так как распределение x_t зависит только от x_{t-1}. Более формально

$$q(x_1, \dots, x_T \mid x_0) = q(x_1 \mid x_0)\dots q(x_T \mid x_{T-1})$$

Это и есть апостериорное распределение, которое мы моделировали в контексте латентных моделей, где $x_1, \dots, x_T$ - латентные переменные.

Истинное распределение задается как $p_\theta(x_0) = \int p_\theta(x_0, x_1, \dots, x_T)\ dx_1 \dots dx_T$, где совместное распределение $p_\theta(x_0, x_1, \dots, x_T)$ тоже марковская цепь, называемая также **обратным процессом**, где переходы задаются как гауссианы, которые мы впоследствии обучаем, постепенно расшумляя данные.

Таким образом, роль кодировщика играет **прямой процесс**, а декодировщика - **обратный**.

2.2 Функция потерь и верхняя оценка

В качестве функции потерь авторы статьи используют NLLLoss. Это стандартная функция потерь в машинном и глубинном обучении, когда мы имеем дело с вероятностными моделями. Ведь глобально наша цель максимизировать правдоподобие, а лучше логарифм правдоподобия. Так как мы будем применять итеративные методы, например, градиентный спуск, то задачу можно переформулировать как задачу минимизации добавив минус. И будем выбирать слагаемые, которые реально встречаются в данных. Получим сумму логарифмов, взвешенных индикаторами. Соответственно можно записать это как математическое ожидание. Так же можно заметить, что по сути мы записали кросс-энтропию между нашим модельным распределением и вырожденным идеальным с индикаторами.

Вспомнив нижнюю оценку, которая применялась в VAE, и подставив вместо z латентные переменные $x_1, \dots, x_T$, получим

$$\mathbb{E}\left(-\log p_\theta(x_0)\right) \leq \mathbb{E}_q\left(-\log \frac{p_\theta(x_0, x_1, \dots, x_T)}{q(x_1, \dots, x_T \mid x_0)}\right)$$

Вспомним, что p и q моделируются марковскими цепями, распишем выражения для них и разобьем произведение в сумму логарифмов таким образом, чтобы были пары $p_\theta(x_{t-1} \mid x_t)$ и $q(x_t \mid x_{t-1})$. В одиночестве останется лишь $p(x_T)$.

$$\mathbb{E}_q \left(-\log \frac{p_\theta(x_0, x_1, \ldots, x_T)}{q(x_1, \ldots, x_T \mid x_0)} \right) = \mathbb{E}_q \left(-\log p(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})} \right) := \mathbb{L}$$

Оптимизируя случайные слагаемые из $\mathbb{L}$ стохастическим градиентным спуском, можно добиться эффективного обучения. Но давайте запишем функцию потерь чуть по-другому.

$$\mathbb{E}_q \left(-\log p(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})} \right) =$$

$$\mathbb{E}_q \left(-\log p(x_T) - \sum_{t > 1} \log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})} - \log \frac{p_\theta(x_0 \mid x_1)}{q(x_1 \mid x_0)} \right) = (*)$$

Распишем по формуле Байеса вероятность

$$q(x_t \mid x_{t-1}) = q(x_t \mid x_{t-1}, x_0) = \frac{q(x_t, x_{t-1}, x_0)}{q(x_{t-1}, x_0)} =$$

$$\frac{q(x_{t-1} \mid x_t, x_0)\ q(x_t, x_0)}{q(x_{t-1} \mid x_0)\ q(x_0)} = \frac{q(x_{t-1} \mid x_t, x_0)\ q(x_t \mid x_0)}{q(x_{t-1}|x_0)} \tag{1}$$

Тогда

$$(*) = \mathbb{E}_q \left(-\log p(x_T) - \sum_{t > 1} \log \frac{p_\theta(x_{t-1} \mid x_t)\ q(x_{t-1}|x_0)}{q(x_{t-1} \mid x_t, x_0)\ q(x_t \mid x_0)} - \log \frac{p_\theta(x_0 \mid x_1)}{q(x_1 \mid x_0)} \right) =$$

Разбив логарифм произведения в сумму и сократив лишнее

$$= \mathbb{E}_q \left(-\log \frac{p(x_T)}{q(x_T)} - \sum_{t > 1} \log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)} - \log p_\theta(x_0 \mid x_1) \right) =$$

Теперь отщипляя одну переменную интегрирования, можно записать выражение внутри ожидания как KL-дивергенцию, а потом вернуть ожидание по всем латентным переменным, потому что она не будет зависить от той переменной.

$$= \mathbb{E}_q \left(D_{KL}(q(x_T \mid x_0) \mid\mid p(x_T)) + \sum_{t > 1} D_{KL}(q(x_{t-1} \mid x_t, x_0) \mid\mid p_\theta(x_{t-1} \mid x_t)) - \log p_\theta(x_0 \mid x_1) \right)$$

Обозначим отдельные ожидания KL-дивергенций за L_j по номеру момента времени, где $j \in \{0, \ldots, T\}$. Заметим, что KL-дивергенции записаны между

нормальными распределениями. Действительно, выражение (1) это произведение плотностей нормальных векторов. Не трудно показать, что в результате тоже будет плотность нормального вектора. В частности, новая дисперсия очевидно просто произведение дисперсий в одномерном случае. Ожидание ищется приведением подобных слагаемых в экспоненте.

$$q(x_{t-1} \mid x_t) = N(x_{t-1};\ \hat{\mu_t}(x_t, x_0), \hat{\beta}_t I)$$

где

$$\hat{\mu_t}(x_t, x_0) := \frac{\sqrt{\hat{\alpha}_{t-1}}\beta_t}{1-\hat{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1-\hat{\alpha}_{t-1})}{1-\hat{\alpha}_t} x_t$$

$$\hat{\beta}_t := \frac{1-\hat{\alpha}_{t-1}}{1-\hat{\alpha}_t}\beta_t$$

2.3 Параметризация

Авторы статьи предлагают избавиться от обучаемых параметров в прямом процессе и задать вариационное расписание в виде констант, которые бы линейно изменялись. Таким образом, первое слагаемое в ожидании не влияет на минимизацию и можно его игнорировать.

Теперь нужно выбрать параметризацию для переходов $p_\theta(x_{t-1} \mid x_t) = N(x_{t-1};\ \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$. В статье берут $\Sigma_\theta(x_t, t) = \sigma_t^2 I$ и в качестве $\sigma_t^2 = \beta_t$. Можно использовать и другое выражение для σ_t^2, но эксперименты показали идентичные результаты.

Чтобы представить $\mu_\theta(x_t, t)$, вспомним, что KL-дивергенция в ожидании для $t > 1$ берется между нормальными векторами $N(x_{t-1};\ \hat{\mu_t}(x_t, x_0), \hat{\beta}_t I)$ и $N(x_{t-1};\ \mu_\theta(x_t, t), \sigma_t^2 I)$. Обозначим его за L_{t-1}.

$$D_{KL}(q(x_{t-1} \mid x_t, x_0) \mid\mid p_\theta(x_{t-1} \mid x_t)) = \int q(x) \log q(x) dx - \int q(x) \log p(x) dx$$

Посчитаем первый интеграл.

$$\int q(x) \log \left[\frac{1}{(2\pi\hat{\beta}_t)^{n/2}} \exp\left(-\frac{||x - \hat{\mu}_t(x_t, x_0)||^2}{2\hat{\beta}_t} \right) \right] dx =$$

$$-\frac{n}{2} \log(2\pi\hat{\beta}_t) - \int q(x) \frac{||x - \hat{\mu}_t(x_t, x_0)||^2}{2\hat{\beta}_t} dx =$$

$$-\frac{n}{2} \log(2\pi\hat{\beta}_t) - \frac{1}{2\hat{\beta}_t} \mathbb{E}\left(\sum_{j=1}^{n} x_j^2 - 2x_j \hat{\mu}_t(x_t, x_0)_j + {\hat{\mu}_t}^2(x_t, x_0)_j \right) =$$

$$-\frac{n}{2} \log(2\pi\hat{\beta}_t) - \frac{1}{2\hat{\beta}_t} \sum_{j=1}^{n} \hat{\beta}_t + {\hat{\mu}_t}^2(x_t, x_0)_j - 2{\hat{\mu}_t}^2(x_t, x_0)_j + {\hat{\mu}_t}^2(x_t, x_0)_j =$$

$$-\frac{n}{2} \log(2\pi\hat{\beta}_t) - \frac{n}{2}$$

А теперь второй.

$$-\int q(x) \log \left[\frac{1}{(2\pi\sigma_t^2)^{n/2}} \exp\left(-\frac{||x - \mu_\theta(x_t, t)||^2}{2\sigma_t^2} \right) \right] dx =$$

$$\frac{n}{2} \log(2\pi\sigma_t^2) + \int q(x) \frac{||x - \mu_\theta(x_t, t)||^2}{2\sigma_t^2} dx =$$

$$\frac{n}{2} \log(2\pi\sigma_t^2) + \frac{1}{2\sigma_t^2} \mathbb{E}\left(\sum_{j=1}^{n} x_j^2 - 2x_j \mu_\theta(x_t, t)_j + \mu_\theta^2(x_t, t)_j \right) =$$

$$\frac{n}{2} \log(2\pi\sigma_t^2) + \frac{1}{2\sigma_t^2} \sum_{j=1}^{n} {\hat{\beta}_t}^2 + {\hat{\mu}_t}^2(x_t, x_0)_j - 2\hat{\mu}_t(x_t, x_0)_j \mu_\theta(x_t, t)_j + \mu_\theta^2(x_t, t)_j =$$

$$\frac{n}{2} \log(2\pi\sigma_t^2) + \frac{n{\hat{\beta}_t}^2 + ||\hat{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)||^2}{2\sigma_t^2}$$

Таким образом, все что от θ не зависит это какая-то константа C. И L_{t-1} можно переписать как

$$L_{t-1} = \mathbb{E}_q \left(\frac{||\hat{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)||^2}{2\sigma_t^2} \right) + C$$

Следовательно, можно придумать модель, которая бы предсказывала $\hat{\mu}_t(x_t, x_0)$, которая была получена при прямом процессе. Но если вспомнить закрытую форму для x_t, выразить оттуда x_0 через x_t и подставить в выражение для $\hat{\mu}_t(x_t, x_0)$, то можно получить эквивалентную запись.

$$L_{t-1} = \mathbb{E}_{x_0, \epsilon} \left(\frac{1}{2\sigma_t^2} || \frac{1}{\sqrt{\alpha_t}} (x_t(x_0, \epsilon) - \frac{\beta_t}{\sqrt{1 - \hat{\alpha_t}}} \epsilon) - \mu_\theta(x_t, t) ||^2 \right) + C$$

Таким образом, μ_θ должна предсказывать $\frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\hat{\alpha_t}}}\epsilon)$. Тогда параметризуем μ_θ как

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} (x_t - \frac{\beta_t}{\sqrt{1 - \hat{\alpha}_t}} \epsilon_\theta)$$

где ϵ_θ предсказывает ϵ по x_t. То есть мы перешли от задачи приближения данных к задаче приближения шума. Подставив в L_{t-1} выражение для ϵ_θ, x_t сокращается и норма в ожидании превращается в $||\epsilon - \epsilon_\theta(x_t, t)||^2$. Ее мы можем минизировать градиентным спуском. Авторы игнорируют скаляры перед нормой в функции потерь и называют ее

$$L_{simple}(\theta) := \mathbb{E}_{x_0,\epsilon,t}\left[||\epsilon - \epsilon_\theta(\sqrt{\hat{\alpha_t}}x_0 + \sqrt{1-\hat{\alpha_t}}\epsilon, t)||^2\right]$$

Теперь когда мы параметризовали ожидание и ковариационную матрицу для обратного процесса, чтобы сэмплировать x_{t-1} из $p_\theta(x_{t-1} \mid x_t)$ и вспомнив, что это нормальное распределение

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\hat{\alpha}_t}}\epsilon_\theta)x_t + \sigma_t^2 z \quad z \sim N(0, I)$$

Теперь все готово, чтобы записать итоговые алгоритмы для обучения и сэмплинга.

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

3 Эксперименты

Авторы статьи берут $T = 1000$ и $\beta_1 = 10^{-4}$ линейно возрастающий до $\beta_T = 0.02$. Значение были подобраны таким образом, чтобы сделать L_T поменьше. Сами данные в свою очередь скалируются в отрезок $[-1, 1]$. Для обратного процесса используется backbone архитектуры U-Net. Параметры шарятся по моменту времени с помощью трансформеров - Transformer sinusoidal position embedding.

В таблице ниже приведены значения IS, FID и Negative Log Likelihood для некоторых известных генеративных моделей. DDPM оказалась лучше в смысле FID метрики всех моделей, кроме conditional версии StyleGAN2+ADA.

Table 1: CIFAR10 results. NLL measured in bits/dim.

Model	IS	FID	NLL Test (Train)
Conditional			
EBM [11]	8.30	37.9	
JEM [17]	8.76	38.4	
BigGAN [3]	9.22	14.73	
StyleGAN2 + ADA (v1) [29]	**10.06**	**2.67**	
Unconditional			
Diffusion (original) [53]			≤ 5.40
Gated PixelCNN [59]	4.60	65.93	3.03 (2.90)
Sparse Transformer [7]			**2.80**
PixelIQN [43]	5.29	49.46	
EBM [11]	6.78	38.2	
NCSNv2 [56]		31.75	
NCSN [55]	8.87 ± 0.12	25.32	
SNGAN [39]	8.22 ± 0.05	21.7	
SNGAN-DDLS [4]	9.09 ± 0.10	15.42	
StyleGAN2 + ADA (v1) [29]	$\mathbf{9.74} \pm 0.05$	3.26	
Ours (L, fixed isotropic $\mathbf{\Sigma}$)	7.67 ± 0.13	13.51	≤ 3.70 (3.69)
Ours (L_{simple})	9.46 ± 0.11	**3.17**	≤ 3.75 (3.72)

Авторы эксперементируют с CIFAR10 и приводят метрики Inception score и FID score, которые оценивают качество сгенерированных изображений генеративными моделями. Отличие FID score в том, что он сравнивает распределение сгенерированных изображений с ground truth распределением, например, с распределением из обучающей выборки. В результате экспериментов авторы добились FID score на обучающей выборке равным 3.17, что лучше многих известных моделей, на тестовой - 5.24, что тоже неплохо. Также было показано, что качество сэмплирования получается лучше, если оптимизировать упрощенную оценку, нежели изначальную. На датасете 256x256 LSUN, было получено качество сэмплирования сравнимое с ProgressiveGAN.

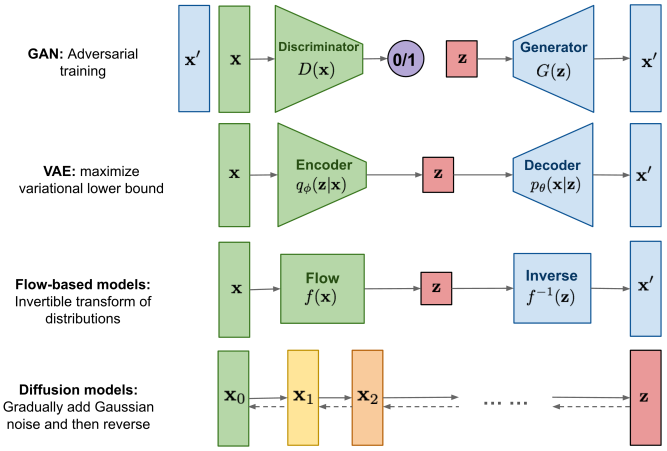
Table 2: Unconditional CIFAR10 reverse process parameterization and training objective ablation. Blank entries were unstable to train and generated poor samples with out-of-range scores.

Objective	IS	FID
$\tilde{\boldsymbol{\mu}}$ prediction (baseline)		
L, learned diagonal $\mathbf{\Sigma}$	7.28 ± 0.10	23.69
L, fixed isotropic $\mathbf{\Sigma}$	8.06 ± 0.09	13.22
$\|\tilde{\boldsymbol{\mu}} - \tilde{\boldsymbol{\mu}}_\theta\|^2$	–	–
$\boldsymbol{\epsilon}$ prediction (ours)		
L, learned diagonal $\mathbf{\Sigma}$	–	–
L, fixed isotropic $\mathbf{\Sigma}$	7.67 ± 0.13	13.51
$\|\tilde{\boldsymbol{\epsilon}} - \boldsymbol{\epsilon}_\theta\|^2$ (L_{simple})	$\mathbf{9.46 \pm 0.11}$	**3.17**

В данной таблице описывается влияние параметризации на качество сэмплирования. Было показано, что baseline, когда мы предсказываем $\hat{\mu}$, работает хорошо только, если использовать полную верхнюю оценку. Также внедрение обучаемых параметров в матрицу ковариации хуже влияет на сэмплирование, чем фиксированные значения. В итоге, оптимизация шума в смысле минимизации второй нормы показала лучшие результаты.

4 Достоинства и недостатки

Сравним диффузионные модели с другими генеративными моделями. Ганы показывают прекрасные результаты, однако обучение нестабильно и пространство сэмплирования ограниченно из-за природы состязательных атак. Потоки требуют поиска обратимого преобразования, что тоже не всегда возможно. Проблемы VAE, как уже замечал ранее, состоят в том, что не всегда получается покрыть $p(z)$ апостериорным распределением.



В статьях, выходивших последние 2 года, было показано, что диффузионные модели показывают лучшее качество сэмплирования, чем state-of-the-art модели до этого, как например ганы.

Одна из проблем диффузионных моделей состоит в том, что несмотря на дешевое обучение, так как нам не нужно проходить всю цепочку от 1 до T, а можно брать случайные моменты времени и делать градиентный шаг, то для сэмплирования приходится это делать, что может занимать достаточно много времени.

5 Источники

1. Оригинальная статья **Denoising Diffusion Probabilistic Models**
2. Выступление Ветрова Дмитрия Петровича **Introduction to diffusion models** на конференции Fall into ML
3. Статья о **Variational Autoencoder**