Stable-Diffusion models

Голов В. А.

Конспекты по пройденному материалу

18 января 2023 г.

Содержание

I. Диффузионная модель

I.1. Что такое диффузионная модель?

Положим существует $q(x_0)$ – распределение исходных данных.То есть распределение в котором выборка $x_0 \sim q(x_0)$. **Прямой диффузионный процесс** $q(x_t|x_{t-1})$ зашумляет данные Гауссовым шумом на каждом шаге t.

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

где $\forall t,\ 0 < \beta_t < 1$ и $\beta_t > \beta_{t-1}$. В классической нотации нормальное распределение $\mathcal{N}(\mu, \sigma^2)$ или в общем виде $\mathcal{N}(\vec{\mu}, \Sigma)$ зависит от параметров смещения μ и разброса σ (среднее и стандартное отклонение). В данном случае $\mu_t = \sqrt{1-\beta_t}x_{t-1}$ и $\sigma_t^2 = \beta_t$. Преобразование зашумления можно определить при помощи добавления аддитивного шума $\varepsilon \sim \mathcal{N}(0, I)$ как

$$x_t = \mu_t + \sigma_t \cdot \varepsilon = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\varepsilon. \tag{2}$$

Это следует из того факта, что если $\xi \sim \mathcal{N}(0,1)$, то $\eta = \sigma\xi + \mu \sim \mathcal{N}(\mu, \sigma^2)$. Заметим, что каждое β_t не является постоянной от времени и называется **запланированным разбросом** и может задаваться по-разному (линейно, квадратически, синусом и тд.).

Таким образом, если бы мы знали условное распределение $p(x_{t-1}|x_t)$, мы бы могли запустить процесс в обратном порядке и получить x_0 выборку из зашумленной x_T, где $t = 0, \cdots, T$.

Так как $p(x_{t-1}|x_t)$ мы не знаем, приблизим его при помощи параметризованной функции распределения $p_\theta(x_{t-1}|x_t)$, где θ – веса, обновляемые в процессе обучения. Так как нормальное распределение зависит от двух параметров, введем параметризованные среднее и разброс (μ_θ и Σ_θ). Тогда наше параметризованное распределение имеет вид

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)). \tag{3}$$

Заметим, что авторы статьи [1] обучают модель только на среднем, а разброс фиксируют как $\Sigma_\theta = \sigma^2 I = \beta_t I$, что было улучшено в статье [2].

I.2. Определение процесса обучения

Если рассматривать q и p_θ как VAE, то можно воспользоваться *variational lower bound* (ELBO) для максимизации правдоподобия. В данном случае ELBO преобразуется в сумму $L = L_0 + L_1 + \cdots + L_T$, где все L_t кроме L_0 имеют вид MSE (L_2 нормы).

Заметим, что для получения x_t из x_0 не нужно проделывать все шаги между ними. При известных β_t достаточно выполнить преобразование

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\overline{\alpha}_t}x_0, (1-\overline{\alpha}_t)I), \tag{4}$$

где $\alpha_t = 1 - \beta_t$ и $\overline{\alpha}_t = \prod_{s=1}^t \alpha_s$. Это хорошо, так как позволяет нам оптимизировать случайные члены функции потерь L (случайным образом семплировать выборку по t). Помимо этого, данное свойство позволяет нам использовать **аппроксимацию аддитивного шума** вместо аппроксимации среднего. То есть наше среднее принимает вид

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\varepsilon_\theta(x_t, t)\right), \tag{5}$$

что позволяет ввести функцию потерь L_t вида

$$\begin{aligned} L_t &= \|\varepsilon - \varepsilon_\theta(x_t, t)\|_2^2 = \\ &= \left\|\varepsilon - \varepsilon_\theta(\sqrt{\overline{\alpha}_t}x_0 + \varepsilon\sqrt{1-\overline{\alpha}_t}, t)\right\|_2^2. \end{aligned} \tag{6}$$

Таким образом алгоритм обучения можно свести к виду Алгоритма 1.

Другими словами, речь идет о следующем:

Алгоритм 1: Обучение модели

повторять

$x_0 \sim q(x_0)$;
$t \sim \mathcal{U}[1, T]$;
$\varepsilon \sim \mathcal{N}(0, I)$;
$\theta \leftarrow \theta -$
$\tau \nabla_\theta \left\| \varepsilon - \varepsilon_\theta(\sqrt{\overline{\alpha}_t} x_0 + \varepsilon \sqrt{1 - \overline{\alpha}_t}, t) \right\|_2^2$
`/* Градиентный спуск */`

до тех пор, пока *не покрыто*;

1. Сэмплируем выборку x_0 из реального распределения $q(x_0)$;

2. Сэмплируем уровень шума из дискретного равномерного распределения $\mathcal{U}[1, T]$;

3. Генерируем шум из нормального распределения и зашумляем данные (как показано выше);

4. На основе зашумленных изображений обучаем сеть определять уровень аддитивного шума.

Далее рассмотрим другие алгоритмы, необходимые при обучении. Алгоритм 2. использется

Алгоритм 2: Сэмплирование

$x_T \sim \mathcal{N}(0, I)$;
цикл $t = T, \cdots, 1$ **выполнять**
  если $t > 1$ **тогда**
    $z \sim \mathcal{N}(0, I)$;
  иначе
    $z = \vec{0}$;
  $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}} \varepsilon_\theta(x_t, t) \right) + z\sigma_t$;
вернуть x_0

авторами статьи [1] для отслеживания прогресса. По-факту речь о том, чтобы сгенерировать шум x_T самостоятельно, а затем с использованием модели привести его к x_0. То есть в идеале должно получиться изображение из исходного распределения $q(x_0)$.

I.3. Наблюдения и некоторые выводы

На выходе имеются смешанные ощущения от модели. С одной стороны прямой процесс зашумления данных кажется очень простым за счет сэмплирования по временному шагу и возможности зашумления $x_0 \to x_t$ без промежуточных шагов.

Обратный проход, который и представляет из себя результирующую нейронную сеть требует использовать пошаговое удаление шума, что приводит к циклу при генерации изображения (см алгоритм 2).

Напрашивается вопрос: "можно ли использовать другой инструментарий для получения такого же результата?".

Рассмотрим $q(x_t|x_{t-1})$ как некоторый дифференциальный закон. Тогда можно сформулировать задачу

$$\begin{cases} \dfrac{dx(t)}{dt} = f_\theta(x(t), t, \theta), \\ x(0) = x_0 \sim q(x_0), \\ x(T) = x_T \sim \mathcal{N}(0, I), \end{cases} \tag{7}$$

где f_θ - некоторая параметризованная функция, которую необходимо обучить зашумлять исходные данные в рамках ОДУ. Тогда обратных проход дает нам x_0 из x_T

$$x_0 = \int_T^0 f_\theta(x(t), t, \theta) dt. \tag{8}$$

Данный подход лишает нас возможности обучать сеть на случайных членах L_t. Однако можно обучать на выборочных отрезках равной длины с фиксированным шагом h. Таким образом мы получаем контролируемую непрерывную производную динамики диффузионного процесса.

Этот процесс больше похож на решение задачи **нормализации потока** и может решаться как задача максимизации правдоподобия. Использование ОДУ гарантирует непрерывность и дифференцируемость результирующей функции, что делает трансформацию более устойчивой.

Список литературы

[1] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. June 2020. arXiv:2006.11239.

[2] A. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. *CoRR*, abs/2102.09672, 2021. URL: https://arxiv.org/abs/2102.09672, arXiv:2102.09672.