Диффузионные модели и их использование для атаки и защиты систем компьютерного зрения

Матюшин Дмитрий

ИСП РАН

22 октября 2025 г.

План занятия

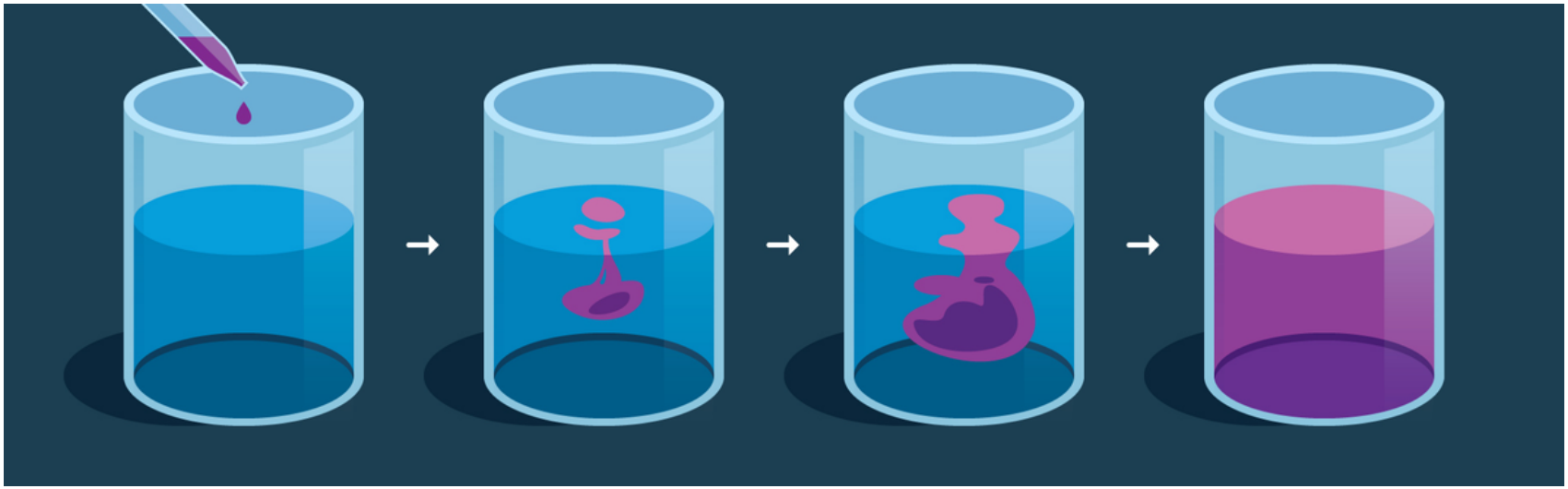
- Общая теория диффузионных моделей;
- Методы генерации изображений: DDPM и DDIM;
- Генерация Text-to-Image;
- Методы защиты при помощи диффузионных моделей. Методы диффузионного редактирования изображений;
- Методы атаки при помощи диффузионных моделей. Проективный градиентный спуск на основе диффузии (Diff-PGD).

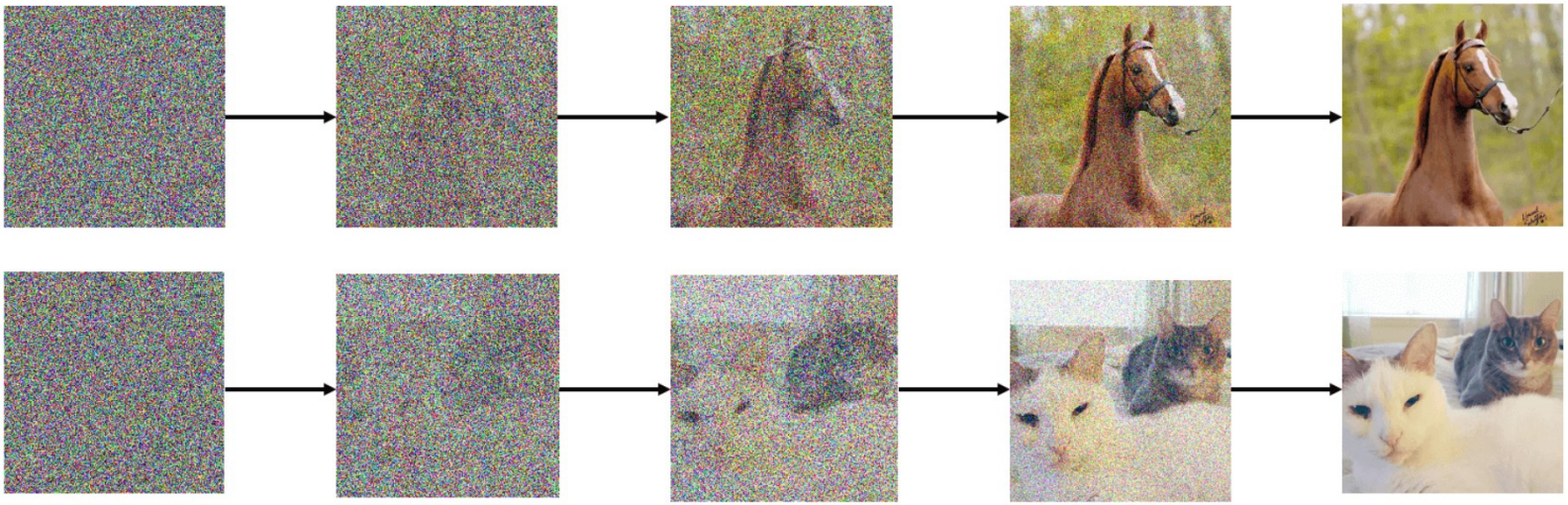
Возможности диффузионных моделей

Качество генерации диффузионных моделей (DDPM)

Вариации изображение при помощи диффузионных моделей

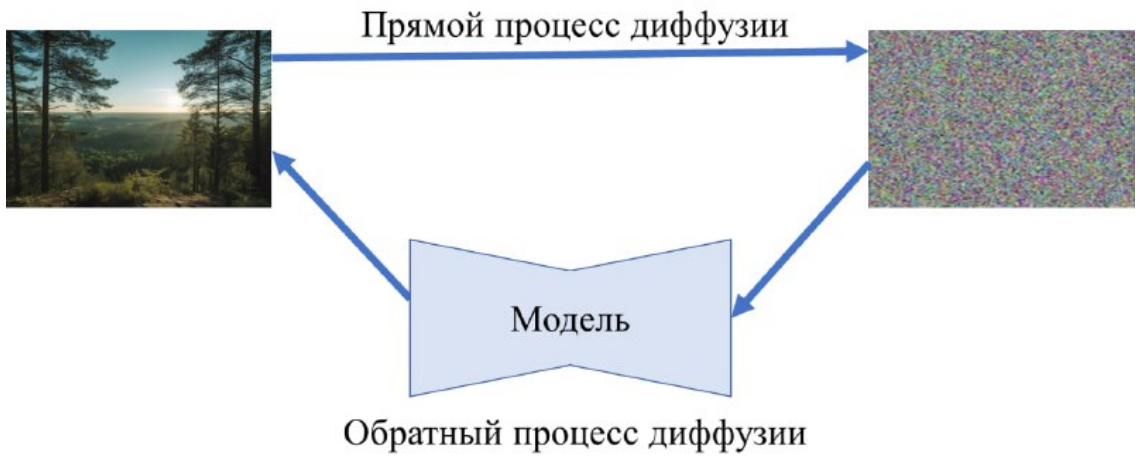
Интуитивное понимание





Основная идея диффузии

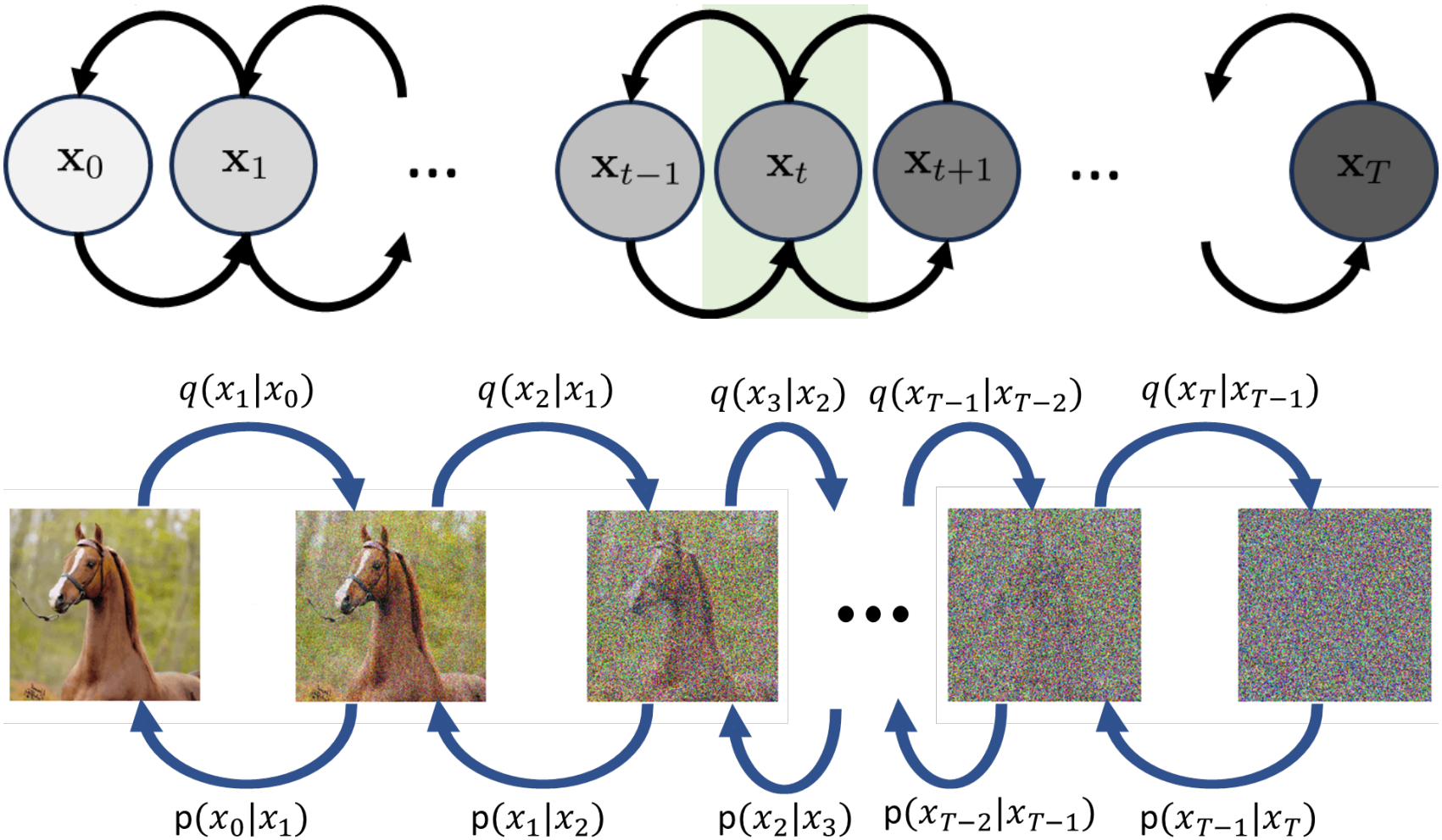
- **Прямой процесс диффузии** - итеративно зашумляет изображение.
- **Обратный диффузионный процесс** - итеративно очищает изображение от шума.



Процесс обучения модели

Процесс генерации изображений

Схема работы диффузионной модели



Аппроксимация гауссианами

Аппроксимируем распределения переходов гауссианами:

$$p(\mathbf{x}_i|\mathbf{x}_{i+1}) \simeq p_\theta(\mathbf{x}_i|\mathbf{x}_{i+1})$$

$$q(\mathbf{x}_{i+1}|\mathbf{x}_i) \simeq q_\varphi(\mathbf{x}_{i+1}|\mathbf{x}_i)$$

Рассмотрим:

$$q_\varphi(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t|a\mathbf{x}_{t-1}, b^2\mathbf{I})$$

Выберем a и b так, чтобы распределение $\mathbf{x}_t$ стало $\mathcal{N}(0, \mathbf{I})$, когда t достаточно велико.
Решение: $a = \sqrt{\alpha}$ и $b = \sqrt{1-\alpha}$, то есть

$$\boxed{q_\varphi(\mathbf{x}_t|\mathbf{x}_{t-1}) \stackrel{\text{def}}{=} \mathcal{N}(\mathbf{x}_t|\sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})}$$

Доказательство $a = \sqrt{\alpha}$ и $b = \sqrt{1-\alpha}$

Эквивалентный шаг выборки:

$$\mathbf{x}_t = a\mathbf{x}_{t-1} + b\boldsymbol{\epsilon}_{t-1}, \quad \boldsymbol{\epsilon}_{t-1} \sim \mathcal{N}(0, \mathbf{I}).$$

Продолжая рекурсию, можно показать, что:
$\mathbf{x}_t = a\mathbf{x}_{t-1} + b\boldsymbol{\epsilon}_{t-1} = a(a\mathbf{x}_{t-2} + b\boldsymbol{\epsilon}_{t-2}) + b\boldsymbol{\epsilon}_{t-1} = \cdots =$
$= a^t\mathbf{x}_0 + b\sum_{i=0}^{t-1} a^i\boldsymbol{\epsilon}_{t-1-i} \stackrel{\text{def}}{=} a^t\mathbf{x}_0 + \mathbf{w}_t$

Это - конечная сумма независимых гауссовских случайных величин.
Среднее значение $\mathbb{E}[\mathbf{w}_t] = 0$, т.к. все средние значения равны нулю.
Ковариация для вектора с нулевым средним будет:

$$\text{Cov}(\mathbf{w}_t) \stackrel{\text{def}}{=} \mathbb{E}[\mathbf{w}_t\mathbf{w}_t^T] = b^2(\text{Cov}(\boldsymbol{\epsilon}_{t-1}) + a^2\text{Cov}(\boldsymbol{\epsilon}_{t-2}) + \cdots + a^{2(t-1)}\text{Cov}(\boldsymbol{\epsilon}_0)).$$

Доказательство $a = \sqrt{\alpha}$ и $b = \sqrt{1-\alpha}$

Поскольку ковариации $\mathrm{Cov}(\boldsymbol{\epsilon}_i) = \mathbf{I}$, это даёт:

$$\mathrm{Cov}(\mathbf{w}_t) = [b^2 \sum_{i=0}^{t-1} a^{2i}]\mathbf{I} = [b^2 \frac{1-a^{2t}}{1-a^2}]\mathbf{I}.$$

Когда $t \to \infty$, $a^t \to 0$ для любого $0 < a < 1$, поэтому при предельном значении $t = \infty$:

$$\lim_{t\to\infty} \mathrm{Cov}(\mathbf{w}_t) = \frac{b^2}{1-a^2}\mathbf{I}.$$

Чтобы $\lim_{t\to\infty} \mathrm{Cov}(\mathbf{w}_t) = \mathbf{I}$, необходимо $b = \sqrt{1-a^2}$.
Если выбрать $a = \sqrt{\alpha}$, то $b = \sqrt{1-\alpha}$. Поэтому

$$\boxed{q_\varphi(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t|\sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})}$$

Итеративное изменение распределения

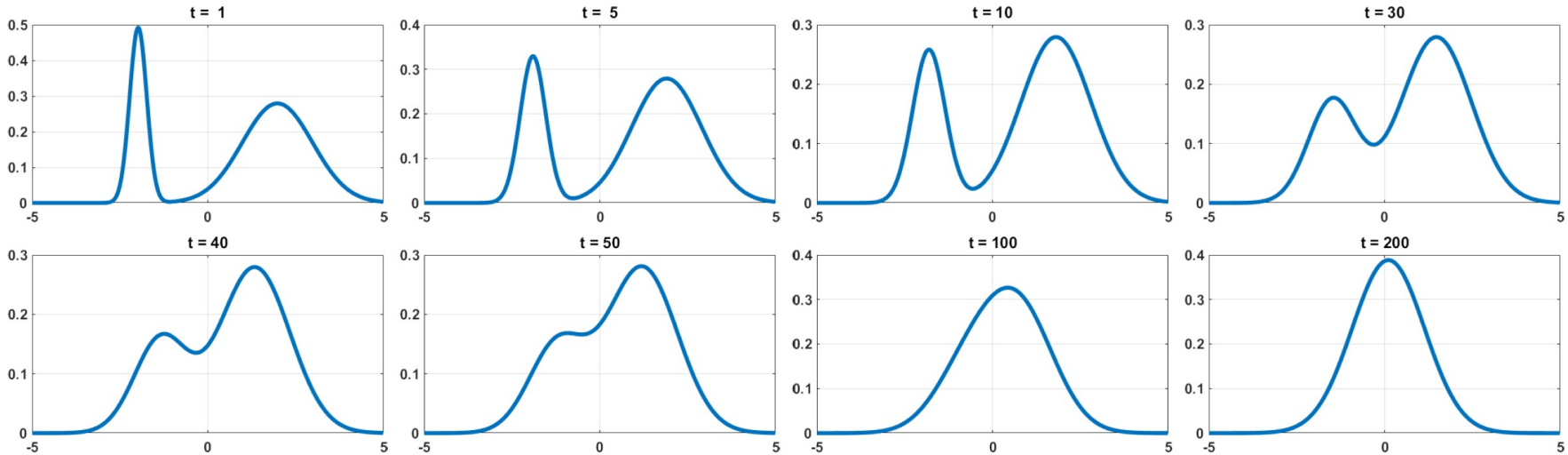
Пример: взвешенная сумма двух гауссиан

$$p_t(x) = \pi_1 \mathcal{N}(x|\mu_1, \sigma_1^2) + \pi_2 \mathcal{N}(x|\mu_2, \sigma_2^2).$$

Запустим для нее прямой процесс диффузии:

$$p_t(x) = \pi_1 \mathcal{N}(x|\sqrt{\alpha_t}\mu_{1,t-1}, \alpha_t\sigma_{1,t-1}^2 + (1-\alpha_t)) + \pi_2 \mathcal{N}(x|\sqrt{\alpha_t}\mu_{2,t-1}, \alpha_t\sigma_{2,t-1}^2 + (1-\alpha_t))$$

Для примера возьмем $\pi_1 = 0.3$, $\pi_2 = 0.7$, $\mu_1 = -2$, $\mu_2 = 2$, $\sigma_1 = 0.2$ и $\sigma_2 = 1$. Шаговая функция: $\alpha_t = 0.97$ для всех t.



Функции распределения вероятностей для различных значений t

Вывод формулы для $q_\varphi(\mathbf{x}_t|\mathbf{x}_0)$

Докажем, что $\boxed{q_\varphi(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})}$, где $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

Шаг выборки:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1}, \quad \boldsymbol{\epsilon}_{t-1} \sim \mathcal{N}(0, \mathbf{I})$$

Рекурсивно подставим значения $\mathbf{x}_{t-1}, \mathbf{x}_{t-2}, ..., \mathbf{x}_0$ в это выражение:

$$\mathbf{x}_t = \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2}\right) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} =$$

$$= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \overset{\text{def}}{=} \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \mathbf{w}_1$$

$\mathbf{w}_1$ соответствует сумме двух гауссиан, которая тоже является гауссианой. Вычислим новую ковариацию (среднее по-прежнему ноль):

$$\mathbb{E}[\mathbf{w}_1\mathbf{w}_1^T] = [\left(\sqrt{\alpha_t}\sqrt{1-\alpha_{t-1}}\right)^2 + \left(\sqrt{1-\alpha_t}\right)^2]\mathbf{I} =$$

$$= \left(\alpha_t(1-\alpha_{t-1}) + 1 - \alpha_t\right)\mathbf{I} = (1-\alpha_t\alpha_{t-1})\mathbf{I}$$

Вывод формулы для $q_\varphi(\mathbf{x}_t|\mathbf{x}_0)$

Подставляя полученную ковариацию в выражение для $\mathbf{x}_t$, получаем следующую гауссиану:

$$\mathbf{x}_t = \sqrt{\alpha_t \alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}}\boldsymbol{\epsilon}_{t-2}$$

Продолжая рекурсию до состояния $\mathbf{x}_0$, получаем:

$$\mathbf{x}_t = \sqrt{\prod_{i=1}^{t} \alpha_i}\mathbf{x}_0 + \sqrt{1 - \prod_{i=1}^{t} \alpha_i}\boldsymbol{\epsilon}_0$$

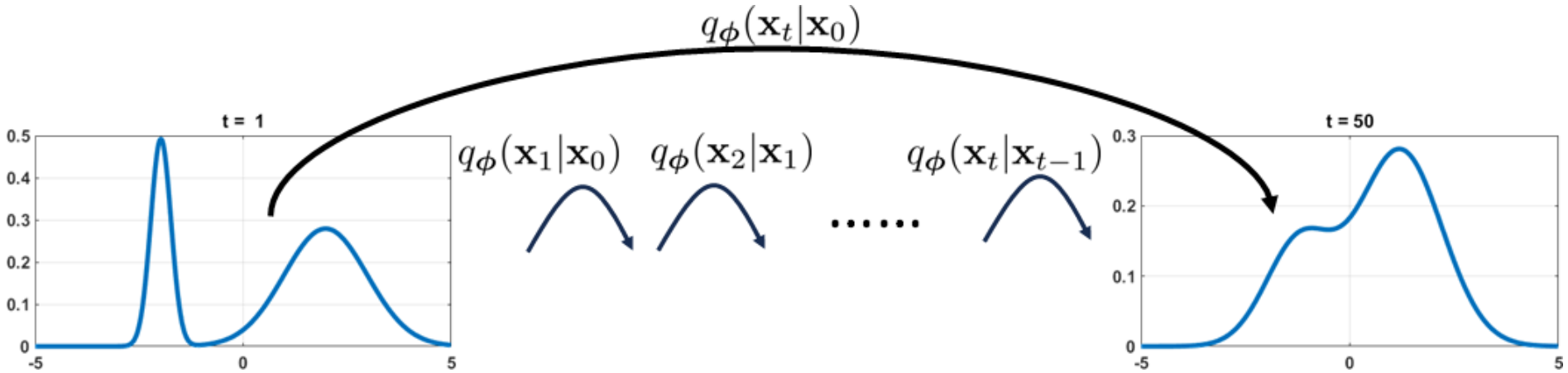
Таким образом, если определить $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$, можно показать, что:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}_0$$

Другими словами, распределение $q_\varphi(\mathbf{x}_t|\mathbf{x}_0)$ можно записать как:

$$\boxed{q_\varphi(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I})}$$

Значение распределения $q_\varphi(\mathbf{x}_t|\mathbf{x}_0)$



Прямой переход между состояниями x_0 и x_t

Вспомнить всё

- $x = \{\mathbf{x}_i\}, \mathbf{x}_i \in \mathbb{R}^p$ - входные данные.
- Хотим выучить $p_\theta(x)$ так, чтобы $p_\theta(x) \approx p_{data}(x)$

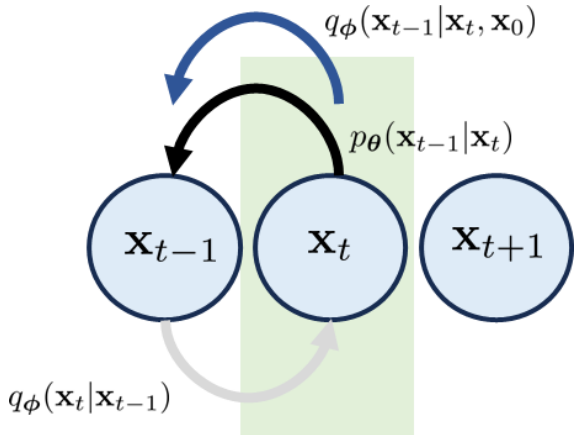
Для любых непрерывных распределений $p(x)$ и $q(x)$:

- Математическое ожидание: $\mathbb{E}_{q(x)}\left[p(x)\right] = \int p(x)q(x)dx$
- KL-дивергенция: $\mathbb{D}_{KL}(p||q) = \int p(x) log \frac{p(x)}{q(x)} dx = \mathbb{E}_{p(x)} log \frac{p(x)}{q(x)}$

Применение теоремы Байеса

Для дальшейших математических выкладок нам удобнее сделать так, чтобы направление $q(x_{t-1}|x_t)$ совпадало с направлением $p_\theta(x_{t-1}|x_t)$. Решение — в использовании простой теоремы Байеса:

$$q(x_t|x_{t-1}) = \frac{q(x_{t-1}|x_t)q(x_t)}{q(x_{t-1})} \overset{\text{марковский процесс}}{\Rightarrow} q(x_t|x_{t-1}, x_0) = \frac{q(x_{t-1}|x_t, x_0)q(x_t|x_0)}{q(x_{t-1}|x_0)}$$



Направление $q(x_{t-1}|x_t)$ теперь совпадает с $p_\theta(x_{t-1}|x_t)$ (см. синия стрелка на рисунке).

Нижняя вариационная граница (ELBO - Evidence Lower BOund, или VLB - Variance Lower Bound)

ELBO обеспечивает оценку снизу (наихудший случай) для логарифмической вероятности распределения данных (в нашем случае $p_{\text{data}}(x)$). Запишем ELBO для вариационной диффузионной модели:

$$\text{ELBO}_{\varphi,\theta}(x) = \boxed{\mathbb{E}_{q_\varphi(x_1|x_0)}[\log p_\theta(x_0|x_1)]} -$$

Восстановление (начальный блок)

$$- \quad \boxed{\mathbb{D}_{KL}(q_\varphi(x_T|x_0)\|p(x_T))} \quad -$$

Соответствие априорному распределению
(финальный блок)

$$-\boxed{\sum_{t=2}^{T} \mathbb{E}_{q_\varphi(x_t|x_0)} \left[\mathbb{D}_{KL}(q_\varphi(x_{t-1}|x_t, x_0)\|p_\theta(x_{t-1}|x_t))\right]}$$

Последовательность (переходные блоки)

Вывод формулы для ELBO

Обозначим через $x_{0:T} = \{x_0, \ldots, x_T\}$ совокупность всех переменных состояния от $t = 0$ до $t = T$. Априорное распределение $p(x)$ — это распределение для изображения x_0, поэтому оно эквивалентно $p(x_0)$. Таким образом, мы можем показать, что:

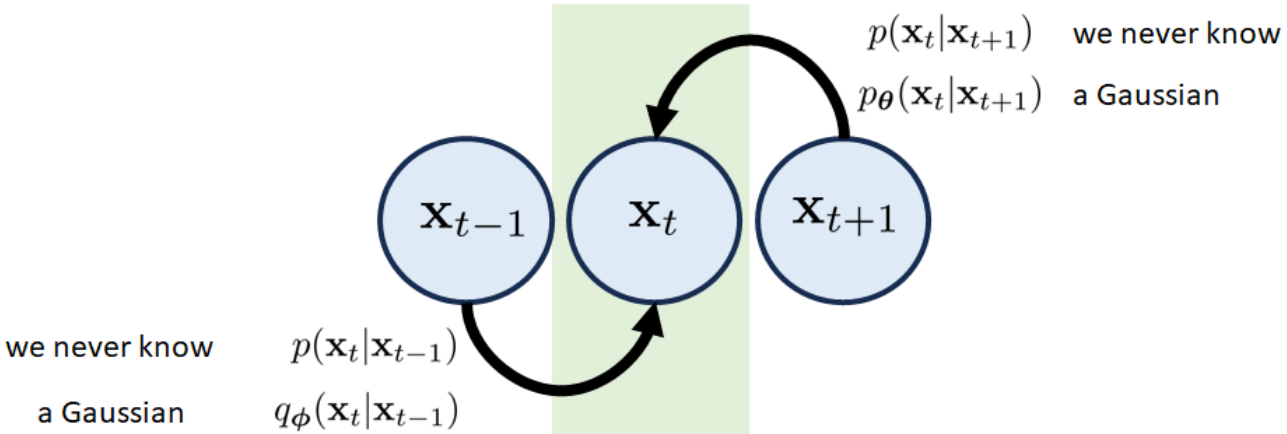
$$\log p(x) = \log p(x_0) = \quad \text{(маргинализация по } x_{1:T})$$
$$= \log \int p(x_{0:T}) dx_{1:T} = \quad \text{(умножение и деление на } q_\varphi(x_{1:T}|x_0$$
$$= \log \int \frac{p(x_{0:T})}{q_\varphi(x_{1:T}|x_0)} q_\varphi(x_{1:T}|x_0) dx_{1:T} = \quad \text{(определение матожидания)}$$
$$= \log \mathbb{E}_{q_\varphi(x_{1:T}|x_0)} \left[\frac{p(x_{0:T})}{q_\varphi(x_{1:T}|x_0)} \right]$$

Неравенство Йенсена: для любой случайной величины X и любой вогнутой функции f выполняется $f(\mathbb{E}[X]) \geq \mathbb{E}[f(X)]$.

Вывод формулы для ELBO

Подставляя, что $f(\cdot) = \log(\cdot)$, мы можем показать, что:

$$\log p(x) = \log \mathbb{E}_{q_\varphi(x_{1:T}|x_0)} \left[\frac{p(x_{0:T})}{q_\varphi(x_{1:T}|x_0)} \right] \geq \mathbb{E}_{q_\varphi(x_{1:T}|x_0)} \left[\log \frac{p(x_{0:T})}{q_\varphi(x_{1:T}|x_0)} \right]$$



Из рисунка видно, что для разбиения $p(x_{0:T})$ надо использовать условие для $x_{t-1}|x_t$:

$$p(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p(x_{t-1}|x_t) = p(x_T)p(x_0|x_1) \prod_{t=2}^{T} p(x_{t-1}|x_t)$$

Вывод формулы для ELBO

Что касается $q_\varphi(x_{1:T}|x_0)$, рисунок подсказывает, что надо использовать условие $x_t|x_{t-1}$. Из условия марковского процесса запишем:

$$q_\varphi(x_{1:T}|x_0) = \prod_{t=1}^{T} q_\varphi(x_t|x_{t-1}) = q_\varphi(x_1|x_0)\prod_{t=2}^{T} q_\varphi(x_t|x_{t-1}) \overset{\text{марковский процесс}}{=}$$

$$\overset{\text{марковский процесс}}{=} q_\varphi(x_1|x_0)\prod_{t=2}^{T} q_\varphi(x_t|x_{t-1}, x_0)$$

Подставим выражения для $p(x_{0:T})$ и $q_\varphi(x_{1:T}|x_0)$ в неравенство Йенсена:

$$\log p(x) \geq \mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \frac{p(x_T)p(x_0|x_1)\prod_{t=2}^{T} p(x_{t-1}|x_t)}{q_\varphi(x_1|x_0)\prod_{t=2}^{T} q_\varphi(x_t|x_{t-1}, x_0)}\right]$$

$$= \underbrace{\mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \frac{p(x_T)p(x_0|x_1)}{q_\varphi(x_1|x_0)}\right]}_{1} + \underbrace{\mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_t|x_{t-1}, x_0)}\right]}_{2}$$

Преобразование произведения из слагаемого 2

Распишем произведение из второго слагаемого 2:

$$\prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_t|x_{t-1}, x_0)} \underset{\text{Байеса}}{\overset{\text{теорема}}{=}} \prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{\frac{q_\varphi(x_{t-1}|x_t, x_0) q_\varphi(x_t|x_0)}{q_\varphi(x_{t-1}|x_0)}}$$

$$= \prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t, x_0)} \times \prod_{t=2}^{T} \frac{q_\varphi(x_{t-1}|x_0)}{q_\varphi(x_t|x_0)} =$$

$$= \underbrace{\prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t, x_0)}}_{\text{без изменений}} \times \left[\frac{q_\varphi(x_1|x_0)}{q_\varphi(x_2|x_0)} \cdot \frac{q_\varphi(x_2|x_0)}{q_\varphi(x_3|x_0)} \cdot \frac{q_\varphi(x_3|x_0)}{q_\varphi(x_4|x_0)} \cdot \ldots \cdot \frac{q_\varphi(x_{T-1}|x_0)}{q_\varphi(x_T|x_0)} \right] =$$

$$= \left[\frac{q_\varphi(x_1|x_0)}{q_\varphi(x_T|x_0)} \right] \times \underbrace{\prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t, x_0)}}_{\text{без изменений}}$$

Снова неравенство Йенсена

Вернемся к неравенству Йенсена, где мы получили слагаемые 1 и 2:

$$\log p(x) \geq \underbrace{\mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \frac{p(x_T)p(x_0|x_1)}{q_\varphi(x_1|x_0)}\right]}_{1} + \underbrace{\mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_t|x_{t-1}, x_0)}\right]}_{2}$$

$$= \underbrace{\mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \frac{p(x_T)p(x_0|x_1)}{q_\varphi(x_1|x_0)}\right]}_{1} + \underbrace{\mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \frac{q_\varphi(x_1|x_0)}{q_\varphi(x_T|x_0)}\right]}_{2a} +$$

$$+ \underbrace{\mathbb{E}_{q_\varphi(x_{1:T}|x_0)}\left[\log \prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t, x_0)}\right]}_{2b} \circledminus$$

Доказательство формулы для ELBO

$$\ominus \underbrace{\mathbb{E}_{q_\varphi(x_1:T|x_0)}\left[\log \frac{p(x_T)p(x_0|x_1)}{q_\varphi(x_T|x_0)}\right]}_{\boxed{*}} + \underbrace{\mathbb{E}_{q_\varphi(x_1:T|x_0)}\left[\log \prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t, x_0)}\right]}_{\boxed{2b}}$$

Перепишем первое слагаемое $\boxed{*}$:

$$\mathbb{E}_{q_\varphi(x_1:T|x_0)}\left[\log \frac{p(x_T)p(x_0|x_1)}{q_\varphi(x_T|x_0)}\right] = \mathbb{E}_{q_\varphi(x_1|x_0)}\left[\log p(x_0|x_1)\right]$$

$$+\mathbb{E}_{q_\varphi(x_T|x_0)}\left[\log \frac{p(x_T)}{q_\varphi(x_T|x_0)}\right] =$$

$$= \underbrace{\boxed{\mathbb{E}_{q_\varphi(x_1|x_0)}\left[\log p_\theta(x_0|x_1)\right]}}_{\textbf{Восстановление}\text{ (начальный блок)}} - \underbrace{\boxed{\mathbb{D}_{KL}(q_\varphi(x_T|x_0)||p(x_T))}}_{\textbf{Соответствие априорному распределению}\text{ (финальный блок)}}$$

Преобразование слагаемого 2b

Временно раскроем формулу математического ожидания:

$$\mathbb{E}_{q_\varphi(x_{t-1},x_t|x_0)}\left[\log \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t,x_0)}\right] = \int q_\varphi(x_{t-1},x_t|x_0)\left[\log \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t,x_0)}\right] dx$$

Применим формулу совместного распределения для марковской цепи: $q_\varphi(x_{t-1},x_t|x_0) = \underbrace{q_\varphi(x_{t-1}|x_t,x_0)}_{\text{заносим в KL-дивергенцию}} \cdot \underbrace{q_\varphi(x_t|x_0)}_{\text{берем по нему матожидание}}$

Теперь перепишем второе слагаемое 2b:

$$\mathbb{E}_{q_\varphi(x_1:T|x_0)}\left[\log \prod_{t=2}^{T} \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t,x_0)}\right] = \sum_{t=2}^{T} \mathbb{E}_{q_\varphi(x_{t-1},x_t|x_0)}\left[\log \frac{p(x_{t-1}|x_t)}{q_\varphi(x_{t-1}|x_t,x_0)}\right] =$$

$$= -\boxed{\sum_{t=2}^{T} \mathbb{E}_{q_\varphi(x_t|x_0)}\left[\mathbb{D}_{KL}(q_\varphi(x_{t-1}|x_t,x_0)||p_\theta(x_{t-1}|x_t))\right]}$$

Последовательность (переходные блоки)

Доказали формулу для ELBO

Итак, мы доказали, что ELBO для вариационной диффузионной модели записывается так:

$$\text{ELBO}_{\varphi,\theta}(x) = \boxed{\mathbb{E}_{q_\varphi(x_1|x_0)}[\log p_\theta(x_0|x_1)]} -$$

Восстановление (начальный блок)

$$- \quad \boxed{\mathbb{D}_{KL}(q_\varphi(x_T|x_0)\|p(x_T))} \quad -$$

Соответствие априорному распределению
(финальный блок)

$$-\boxed{\sum_{t=2}^{T} \mathbb{E}_{q_\varphi(x_t|x_0)} \left[\mathbb{D}_{KL}(q_\varphi(x_{t-1}|x_t, x_0)\|p_\theta(x_{t-1}|x_t))\right]}$$

Последовательность (переходные блоки)

Так чего же мы добились?

- Хотим, чтобы $\mathbf{x}_t$ стало $\mathcal{N}(0, \mathbf{I})$, когда t достаточно велико $\Rightarrow$ $\boxed{q_\varphi(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})}$ - аппроксимировали переходы прямой диффузии гауссианами
- «Развернули» направление действия q_φ при помощи теоремы Байеса: $\boxed{q(x_t|x_{t-1}) = \frac{q(x_{t-1}|x_t)q(x_t)}{q(x_{t-1})}}$
- Оценили логарифмическое правдоподобие снизу: $\log p(x) \geq ELBO_{\varphi,\theta}(x)$

Формула для $q_\varphi(x_{t-1}|x_t, x_0)$

Распределение $q_\varphi(x_{t-1}|x_t, x_0)$ имеет вид:

$$\boxed{q_\varphi(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1} \mid \mu_q(x_t, x_0), \Sigma_q(t))}$$

$$\mu_q(x_t, x_0) = \frac{(1-\overline{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\overline{\alpha}_t} x_t + \frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t} x_0$$

$$\Sigma_q(t) = \frac{(1-\alpha_t)(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{I} = \sigma_q^2(t)\mathbf{I}$$

Вывод формулы для $q_\varphi(x_{t-1}|x_t, x_0)$

Применим теорему Байеса для $q_\varphi(x_{t-1}|x_t, x_0)$:

$$q_\varphi(x_{t-1}|x_t, x_0) = \frac{q_\varphi(x_t|x_{t-1})q_\varphi(x_{t-1}|x_0)}{q_\varphi(x_t|x_0)}$$

$$= \frac{\mathcal{N}(x_t|\sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\mathbf{I})\mathcal{N}(x_{t-1}|\sqrt{\overline{\alpha}_{t-1}}x_0, (1-\overline{\alpha}_{t-1})\mathbf{I})}{\mathcal{N}(x_t|\sqrt{\overline{\alpha}_t}x_0, (1-\overline{\alpha}_t)\mathbf{I})}$$

Для упрощения предположим, что все векторы являются скалярами. Тогда это произведение гауссиан становится:

$$q_\varphi(x_{t-1}|x_t, x_0) \propto \exp\left\{-\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{2(1-\alpha_t)} - \frac{(x_{t-1} - \sqrt{\overline{\alpha}_{t-1}}x_0)^2}{2(1-\overline{\alpha}_{t-1})} + \frac{(x_t - \sqrt{\overline{\alpha}_t}x_0)^2}{2(1-\overline{\alpha}_t)}\right\}$$

Здесь применена формула: $\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(x-\mu)^2}{2\sigma^2})$

Замена переменных

В целях упрощения выражения рассмотрим замену:

$$x = x_t, \quad y = x_{t-1}, \quad z = x_0$$

$$a = \alpha_t, \quad b = \overline{\alpha}_{t-1}, \quad c = \overline{\alpha}_t$$

Рассмотрим показатель экспоненты как квадратичную функцию от y:

$$f(y) = -\frac{(x - \sqrt{a}y)^2}{2(1-a)} - \frac{(y - \sqrt{b}z)^2}{2(1-b)} + \frac{(x - \sqrt{c}z)^2}{2(1-c)}.$$

Независимо от порядка членов, результирующая функция остаётся параболой. Максимум $f(y)$ — это среднее значение результирующего гауссового распределения. Найдем этот максимум при помощи производной $f'(y)$:

$$f'(y) = \frac{ab-1}{(1-a)(1-b)}y + \left(\frac{\sqrt{a}}{1-a}x + \frac{\sqrt{b}}{1-b}z\right).$$

Нахождение максимума через производную

Приравняв $f'(y) = 0$, мы получаем:

$$y = \frac{(1-b)\sqrt{a}}{1-ab}x + \frac{(1-a)\sqrt{b}}{1-ab}z.$$

Отсюда мы знаем, что $ab = \alpha_t\overline{\alpha}_{t-1} = \overline{\alpha}_t$, следовательно:

$$\mu_q(x_t, x_0) = \frac{(1-\overline{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\overline{\alpha}_t}x_t + \frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t}x_0.$$

Аналогично для дисперсии можем вычислить кривизну $f''(y)$. Легко показать, что:

$$f''(y) = -\frac{1-ab}{(1-a)(1-b)} = -\frac{1-\overline{\alpha}_t}{(1-\alpha_t)(1-\overline{\alpha}_{t-1})}.$$

Обратная величина даёт нам ковариационную матрицу:

$$\Sigma_q(t) = \frac{(1-\alpha_t)(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{I}.$$

Доказали формулу для $q_\varphi(x_{t-1}|x_t, x_0)$

Итак, мы показали, что распределение $q_\varphi(x_{t-1}|x_t, x_0)$ имеет вид:

$$\boxed{q_\varphi(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1} \mid \mu_q(x_t, x_0), \Sigma_q(t))}$$

$$\mu_q(x_t, x_0) = \frac{(1-\overline{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\overline{\alpha}_t}x_t + \frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t}x_0$$

$$\Sigma_q(t) = \frac{(1-\alpha_t)(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{I} = \sigma_q^2(t)\mathbf{I}$$

Анализ формулы $q_\varphi(x_{t-1}|x_t, x_0)$

$q_\varphi(x_{t-1}|x_t, x_0)$ полностью определяется через x_t и x_0. Нейронная сеть здесь не требуется для оценки среднего значения и дисперсии! Распределение $q_\varphi(x_{t-1}|x_t, x_0)$ автоматически задаётся, если мы знаем x_t и x_0.

Слагаемое Последовательность - это сумма нескольких членов с KL-дивергенцией ($\mathbb{D}_{KL}$). Проанализируем t-ю KL-дивергенцию:

$$\mathbb{D}_{KL}(\underbrace{q_\varphi(x_{t-1}|x_t, x_0)}_{\text{знаем}} \| \quad \underbrace{p_\theta(x_{t-1}|x_t)}_{\text{Хьюстон, у нас проблема}} \quad).$$

Распределение $p_\theta(x_{t-1}|x_t)$

$q_\varphi(x_{t-1}|x_t, x_0)$ мы уже определили.

Смело предположим, что $p_\theta(x_{t-1}|x_t)$ - гауссиана. Имеем право выбирать p_θ, поэтому выбираем самое простое:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1} \mid \mu_\theta(x_t), \sigma_q^2(t)\mathbf{I})$$

Да кто такая эта ваша дисперсия?

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1} \mid \underbrace{\mu_\theta(x_t)}_{\text{нейронка}}, \underbrace{\sigma_q^2(t)\mathbf{I}}_{\text{фиксируем, как нам удобно}})$$

Мы предполагаем, что среднее значение $\mu_\theta(x_t)$ гауссианы $p_\theta(x_{t-1}|x_t)$ может быть определено с помощью нейронной сети. Выберем дисперсию $\sigma_q^2(t)$ такой же, как у $q_\varphi(x_{t-1}|x_t, x_0)$! А именно:

$$\Sigma_q(t) = \frac{(1-\alpha_t)(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{I} = \sigma_q^2(t)\mathbf{I}$$

Сопоставим $q_\varphi(x_{t-1}|x_t, x_0)$ и $p_\theta(x_{t-1}|x_t)$:

$$q_\varphi(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1} \mid \underbrace{\mu_q(x_t, x_0)}_{\text{знаем}}, \underbrace{\sigma_q^2(t)\mathbf{I}}_{\text{знаем}}),$$

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1} \mid \underbrace{\mu_\theta(x_t)}_{\text{нейронка}}, \underbrace{\sigma_q^2(t)\mathbf{I}}_{\text{знаем}}).$$

Упрощение KL-дивергенции

KL-дивергенция между двумя гауссианами с одинаковыми дисперсиями — это просто квадрат евклидова расстояния между двумя средними значениями:

$$D_{KL}(q_\varphi(x_{t-1}|x_t, x_0)\|p_\theta(x_{t-1}|x_t)) = \frac{1}{2\sigma_q^2(t)}\|\mu_q(x_t, x_0) - \mu_\theta(x_t)\|^2$$

Пользуясь формулой для KL-двергенции, перепишем выражение для $ELBO$:

$$ELBO_\theta(x) = \underbrace{\boxed{\mathbb{E}_{q(x_1|x_0)}[\log p_\theta(x_0|x_1)]}}_{\text{Восстановление}} - \underbrace{\boxed{D_{KL}(q(x_T|x_0)\|p(x_T))}}_{\text{Соответствие априорному распределению}} - \overbrace{\boxed{\sum_{t=2}^{T}\mathbb{E}_{q(x_t|x_0)}\left[\frac{1}{2\sigma_q^2(t)}\|\mu_q(x_t, x_0) - \mu_\theta(x_t)\|^2\right]}}^{\text{Последовательность}}$$

нейронная сеть не нужна

Выводы из выражения для $ELBO$

Выражение для $ELBO$ предполагает, что нам нужно найти сеть μ_θ, которая каким-то образом минимизирует вот такую норму разницы:

$$\frac{1}{2\sigma_q^2(t)}\|\mu_q(x_t, x_0) - \mu_\theta(x_t)\|^2$$

Вспомним уравнение для $\mu_q(x_t, x_0)$:

$$\mu_q(x_t, x_0) = \frac{(1-\overline{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\overline{\alpha}_t}x_t + \frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t}x_0$$

DDPM - Denoising Diffusion Probabilistic Models, диффузионные вероятностные модели денойзинга [5].

Реализация DDPM 1. Предсказание изображения

Поскольку μ_θ — это наша конструкция (выучиваемая сетью), мы можем определить её более удобным образом. Вот первый вариант:

$$\mu_\theta(x_t) = \frac{(1-\overline{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\overline{\alpha}_t}x_t + \frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t}\hat{x}_\theta(x_t).$$

Такая форма записи лежит в основе метода "Предсказание изображения".
Подставим выражение для $\mu_\theta(x_t)$ в выражение для $D_{KL}(q_\varphi(x_{t-1}|x_t, x_0)\|p_\theta(x_{t-1}|x_t))$:

$$\frac{1}{2\sigma_q^2(t)}\|\mu_q(x_t, x_0) - \mu_\theta(x_t)\|^2 = \frac{1}{2\sigma_q^2(t)}\left\|\frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t}(\hat{x}_\theta(x_t) - x_0)\right\|^2 =$$

$$= \frac{1}{2\sigma_q^2(t)}\frac{(1-\alpha_t)^2\overline{\alpha}_{t-1}}{(1-\overline{\alpha}_t)^2}\|\hat{x}_\theta(x_t) - x_0\|^2$$

Реализация DDPM 1. Упрощение формулы для $ELBO$

Таким образом, ELBO можно упростить до следующего выражения (учитываем только слагаемые, где требуется обучение):

$$ELBO_\theta = \mathbb{E}_{q(x_1|x_0)}[\log p_\theta(x_0|x_1)] - \sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)} \left[\frac{(1-\alpha_t)^2 \overline{\alpha}_{t-1}}{2\sigma_q^2(t)(1-\overline{\alpha}_t)^2} \|\hat{x}_\theta(x_t) - x_0\|^2 \right]$$

Распишем первое слагаемое, логарифмируя формулу плотности вероятности нормального распределения и отбрасывая константные слагаемые:

$$\log p_\theta(x_0|x_1) = \log \mathcal{N}(x_0|\mu_\theta(x_1), \sigma_q^2(1)\mathbf{I}) \overset{def}{\propto} -\frac{1}{2\sigma_q^2(1)} \|\mu_\theta(x_1) - x_0\|^2 =$$

$$= -\frac{1}{2\sigma_q^2(1)} \left\| \frac{(1-\overline{\alpha}_0)\sqrt{\alpha_1}}{1-\overline{\alpha}_1} x_1 + \frac{(1-\alpha_1)\sqrt{\overline{\alpha}_0}}{1-\overline{\alpha}_1} \hat{x}_\theta(x_1) - x_0 \right\|^2 \overset{\boxed{\alpha_0 = 1}}{=}$$

$$\overset{\boxed{\alpha_0 = 1}}{=} -\frac{1}{2\sigma_q^2(1)} \left\| \frac{(1-\alpha_1)}{1-\overline{\alpha}_1} \hat{x}_\theta(x_1) - x_0 \right\|^2 \overset{\boxed{\overline{\alpha}_1 = \alpha_1}}{=} -\frac{1}{2\sigma_q^2(1)} \|\hat{x}_\theta(x_1) - x_0\|^2$$

Реализация DDPM 1. Окончательное выражение для $ELBO$

Подставляя выражение для первого слагаемого в формулу $ELBO$ получаем окончательную $ELBO$:

$$ELBO_\theta = -\sum_{t=1}^{T} \mathbb{E}_{q(x_t|x_0)} \left[\frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2 \overline{\alpha}_{t-1}}{(1-\overline{\alpha}_t)^2} \|\hat{x}_\theta(x_t) - x_0\|^2 \right]$$

Функция потерь для модели диффузионного денойзинга (DDPM) - Предсказание изображения:

$$\theta^* = \arg\min_\theta \sum_{t=1}^{T} \frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2 \overline{\alpha}_{t-1}}{(1-\overline{\alpha}_t)^2} \mathbb{E}_{q(x_t|x_0)} \left[\|\hat{x}_\theta(x_t) - x_0\|^2 \right]$$

Реализация DDPM 1. Обучение

Процесс обучения для реализации DDPM

Предсказание изображения:

Для каждого изображения x_0 из тренировочного набора данных выполните следующие шаги до сходимости:

1. Выберите случайную временную метку (timestamp) $t \sim \text{Uniform}[1, T]$.
2. Сгенерируйте образец (sample) $x_t \sim \mathcal{N}(x_t | \sqrt{\overline{\alpha}_t} x_0, (1 - \overline{\alpha}_t) I)$, т.е.

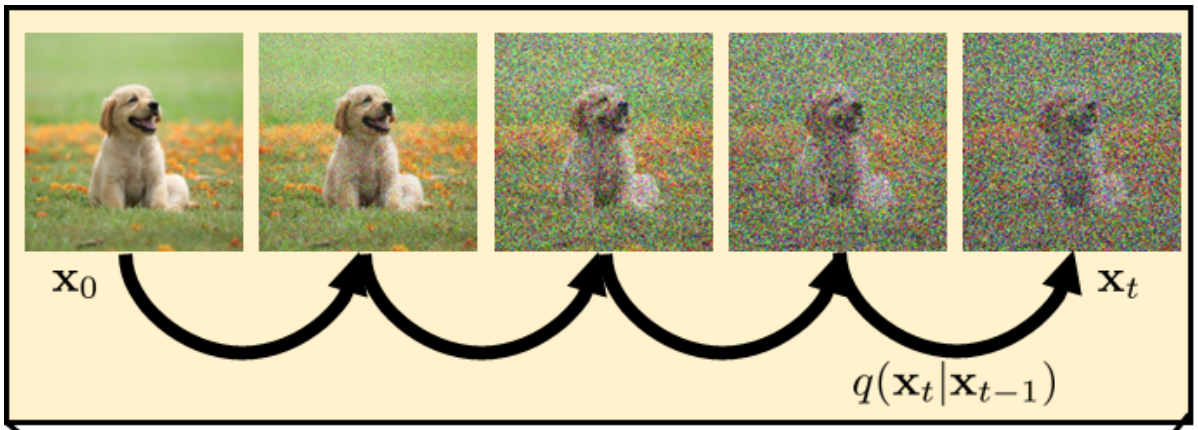
$$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t} z, \quad z \sim \mathcal{N}(0, I).$$

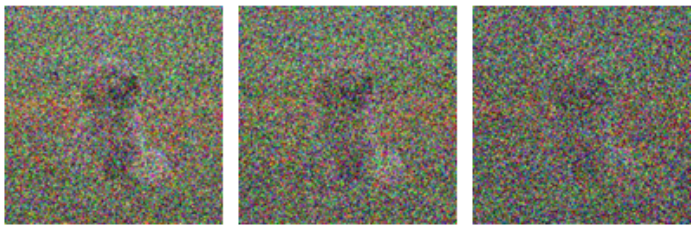
3. Выполните шаг градиентного спуска по параметру:

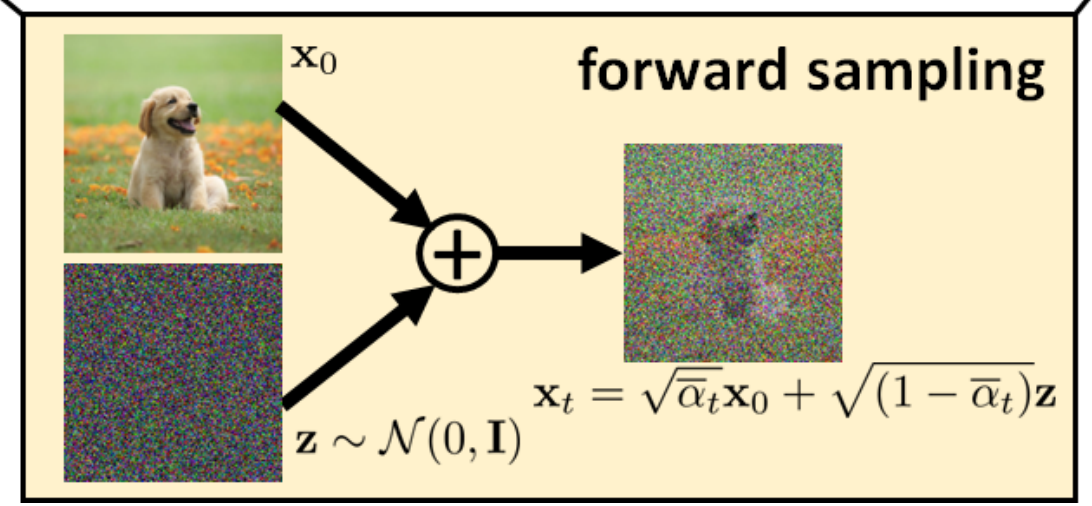
$$\nabla_\theta \| \hat{x}_\theta(x_t) - x_0 \|^2$$

Реализация DDPM 1. Сэмплирование

Процесс сэмплирования (генерации шумовых образцов) для реализации DDPM **Предсказание изображения**:



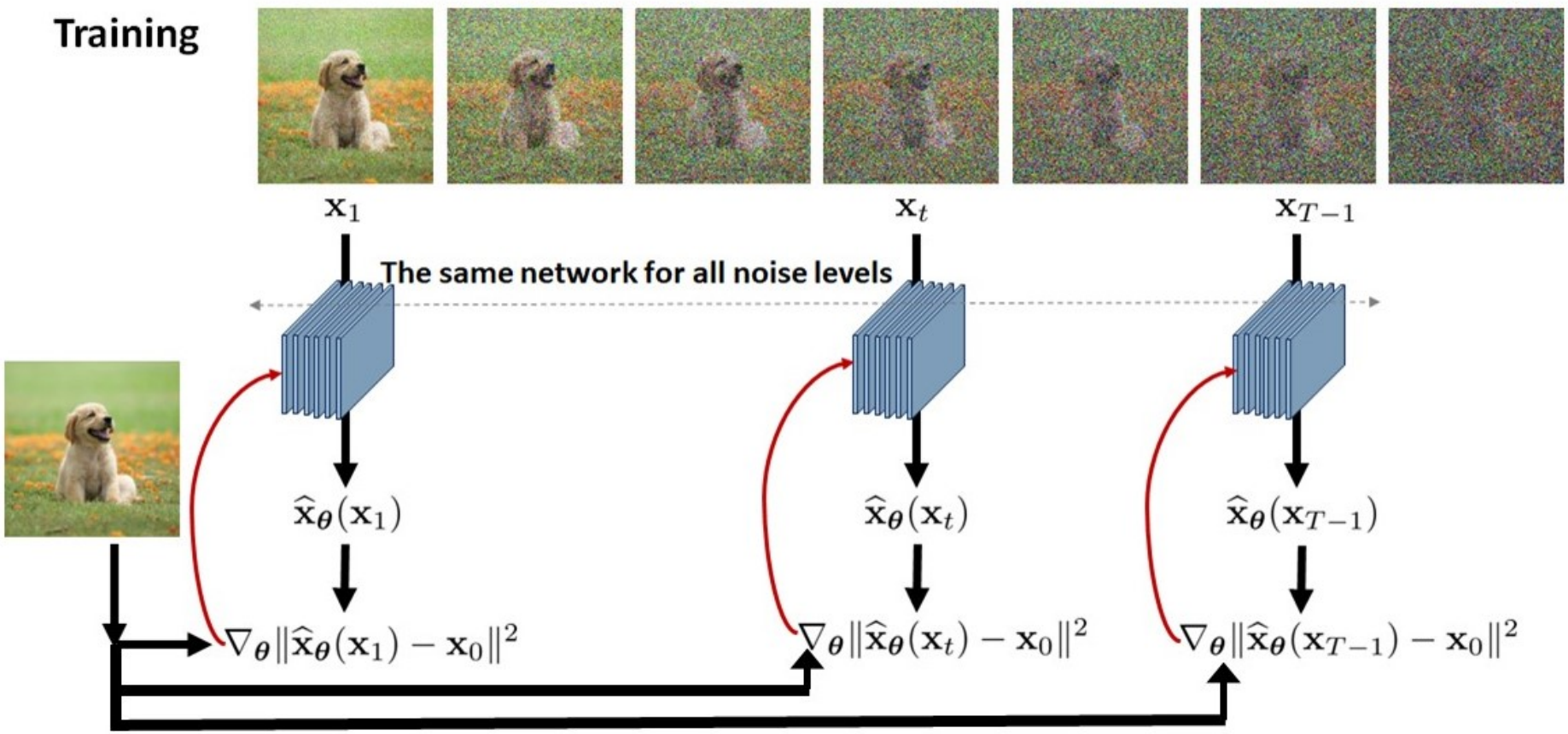




Реализация DDPM 1. Обучение

Процесс обучения для реализации DDPM

Предсказание изображения:



Реализация DDPM 1. Генерация (Inference)

Задача генерации заключается в том, чтобы сэмплировать изображения из распределений $p_\theta(x_{t-1}|x_t)$ на последовательности состояний $x_T, x_{T-1}, \ldots, x_1$. Поскольку это обратный диффузионный процесс, его нужно выполнять рекурсивно:

$$x_{t-1} \sim p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}|\mu_\theta(x_t), \sigma_q^2(t)\mathbf{I})$$

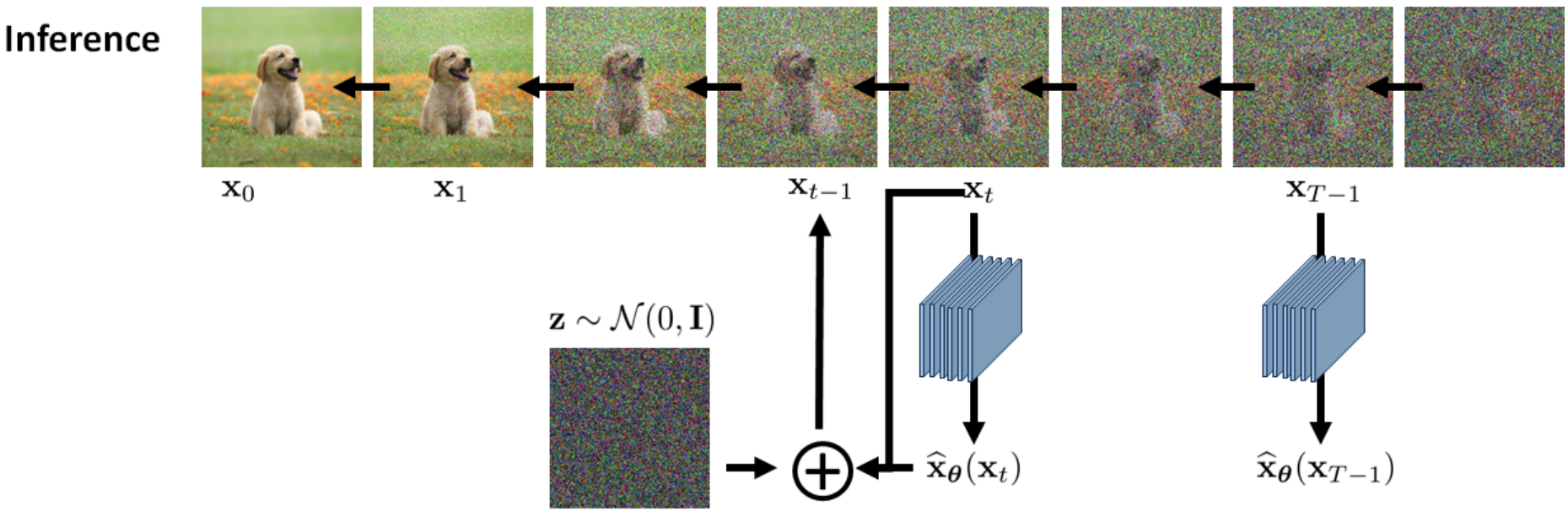
$$= \mu_\theta(x_t) + \sigma_q^2(t)z\textcircled{=}, \quad z \sim \mathcal{N}(0, I).$$

$$\textcircled{=}\frac{(1-\overline{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\overline{\alpha}_t}x_t + \frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t}\hat{x}_\theta(x_t) + \sigma_q(t)z$$

Реализация DDPM 1. Генерация (Inference)

1. Начинаем с вектора белого шума $x_T \sim \mathcal{N}(0, I)$.
2. Повторяем следующие шаги для $t = T, T-1, \ldots, 1$.
   - Вычисляем $\hat{x}_\theta(x_t)$ с использованием обученной сети $\hat{x}_\theta$.
   - Обновляем состояние по следующему правилу:

$$x_{t-1} = \frac{(1-\overline{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\overline{\alpha}_t} x_t + \frac{(1-\alpha_t)\sqrt{\overline{\alpha}_{t-1}}}{1-\overline{\alpha}_t} \hat{x}_\theta(x_t) + \sigma_q(t) z, \quad z \sim (0, I)$$



Реализация DDPM 2. Предсказание шума

Другой подход: будем предсказывать шум, а не само изображение. Давайте рассмотрим выражение x_t через x_0 и шумовое слагаемое. Выразим из него x_0:

$$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon_0 \quad \Rightarrow \quad x_0 = \frac{x_t - \sqrt{1 - \overline{\alpha}_t}\epsilon_0}{\sqrt{\overline{\alpha}_t}}.$$

Подставим это в выражение для $\mu_q(x_t, x_0)$:

$$\mu_q(x_t, x_0) = \frac{\sqrt{\alpha_t}(1 - \overline{\alpha}_{t-1})x_t + \sqrt{\overline{\alpha}_{t-1}}(1 - \alpha_t)x_0}{1 - \overline{\alpha}_t}$$

$$= \frac{\sqrt{\alpha_t}(1 - \overline{\alpha}_{t-1})x_t + \sqrt{\overline{\alpha}_{t-1}}(1 - \alpha_t)\left(x_t - \frac{\sqrt{1-\overline{\alpha}_t}\epsilon_0}{\sqrt{\overline{\alpha}_t}}\right)}{1 - \overline{\alpha}_t} =$$

$$= \boxed{\text{...алгебра...}} = \frac{x_t}{\sqrt{\alpha_t}} - \frac{1 - \alpha_t}{\sqrt{1 - \overline{\alpha}_t}\sqrt{\alpha_t}}\epsilon_0$$

Реализация DDPM 2. Вывод $ELBO$

Таким образом, можем выбрать оценку для μ_θ так, чтобы она соответствовала форме:

$$\mu_\theta(x_t) = \frac{x_t}{\sqrt{\alpha_t}} - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}\sqrt{\alpha_t}}\hat{\epsilon}_\theta(x_t).$$

Подставив выражения для $\mu_q(x_t, x_0)$ и $\mu_\theta(x_t)$ в уравнение для $ELBO$ через $\|\mu_q(x_t, x_0) - \mu_\theta(x_t)\|^2$, мы получаем новое выражение для $ELBO$:

$$ELBO_\theta = -\sum_{t=1}^{T} \mathbb{E}_{q(x_t|x_0)} \left[\frac{1}{2\sigma_q^2(t)} \frac{(1-\alpha_t)^2 \overline{\alpha}_{t-1}}{(1-\overline{\alpha}_t)^2} \|\hat{\epsilon}_\theta(x_t) - \epsilon_0\|^2 \right].$$

Реализация DDPM 2. Предсказание шума. Обучение

Повторяем следующие шаги до сходимости:

1. Выбираем случайную временную метку (timestamp) $t \sim \text{Uniform}[1, T]$.
2. Генерируем образец (sample) $x_t \sim \mathcal{N}(x_t | \sqrt{\overline{\alpha}_t} x_0, (1 - \overline{\alpha}_t) I)$, т.е.

$$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t} z, \quad z \sim \mathcal{N}(0, I).$$

3. Выполняем шаг градиентного спуска по следующей функции:

$$\nabla_\theta \| \hat{\epsilon}_\theta(x_t) - \epsilon_0 \|^2.$$

Реализация DDPM 2. Генерация (Inference)

Соответственно, шаг генерации можно вывести так:

$$x_{t-1} \sim p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}|\mu_\theta(x_t), \sigma_q^2(t)\mathbf{I}) = \mu_\theta(x_t) + \sigma_q^2(t)z =$$

$$= \frac{x_t}{\sqrt{\alpha_t}} - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}\sqrt{\alpha_t}}\hat{\epsilon}_\theta(x_t) + \sigma_q(t)z =$$

$$= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\hat{\epsilon}_\theta(x_t)\right) + \sigma_q(t)z, \quad z \sim (0, I)$$

Реализация DDPM 2. Алгоритм генерации (Inference)

1. Начинаем с вектора белого шума $x_T \sim \mathcal{N}(0, I)$.
2. Повторяем следующие шаги для $t = T, T-1, \ldots, 1$:
   - Вычисляем $\hat{\epsilon}_\theta(x_t)$ с использованием обученной сети $\hat{\epsilon}_\theta$.
   - Обновляем состояние по следующему правилу:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}} \hat{\epsilon}_\theta(x_t) \right) + \sigma_q(t) z, \quad z \sim (0, I).$$

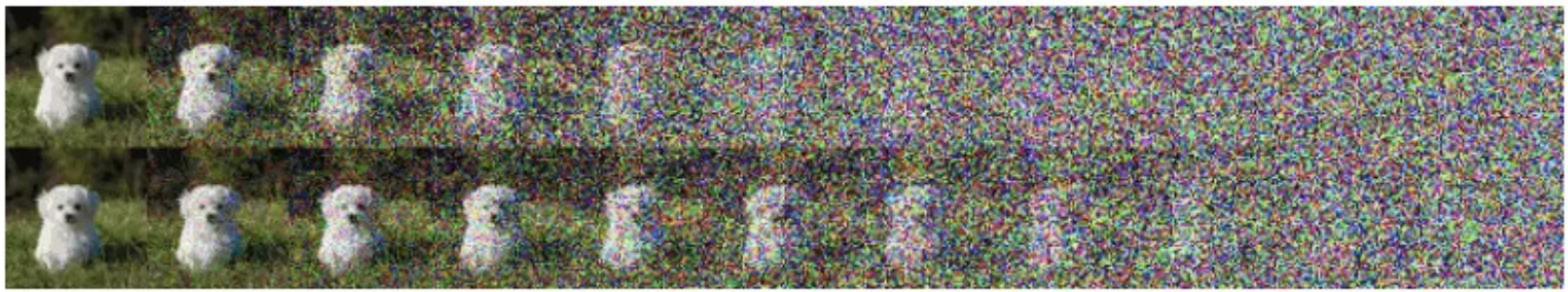
Выбор гиперпараметров расписания α_t

Введем $\beta_t = 1 - \alpha_t$

Основное требование - неубывание $\beta_0 \leq ... \leq \beta_T$ и чтобы прямой процесс сходился к $\mathcal{N}(0, \mathbf{I})$ в пределе по T.

В оригинальной статье DDPM [5] предложено брать $T = 1000$ и линейное «расписание» с $\beta_1 = 10^{-4}, ..., \beta_T = 0.02$.

В следующих работах было предложено «косинусное расписание»:
$\beta_t = clip(1 - \frac{\overline{\alpha}_t}{\overline{\alpha}_{t-1}}, 0.999), \quad \overline{\alpha}_t = \frac{f(t)}{f(0)}, \quad$ где $f(t) = \cos(\frac{\frac{t}{T}+s}{1+s} \cdot \frac{\pi}{2})$



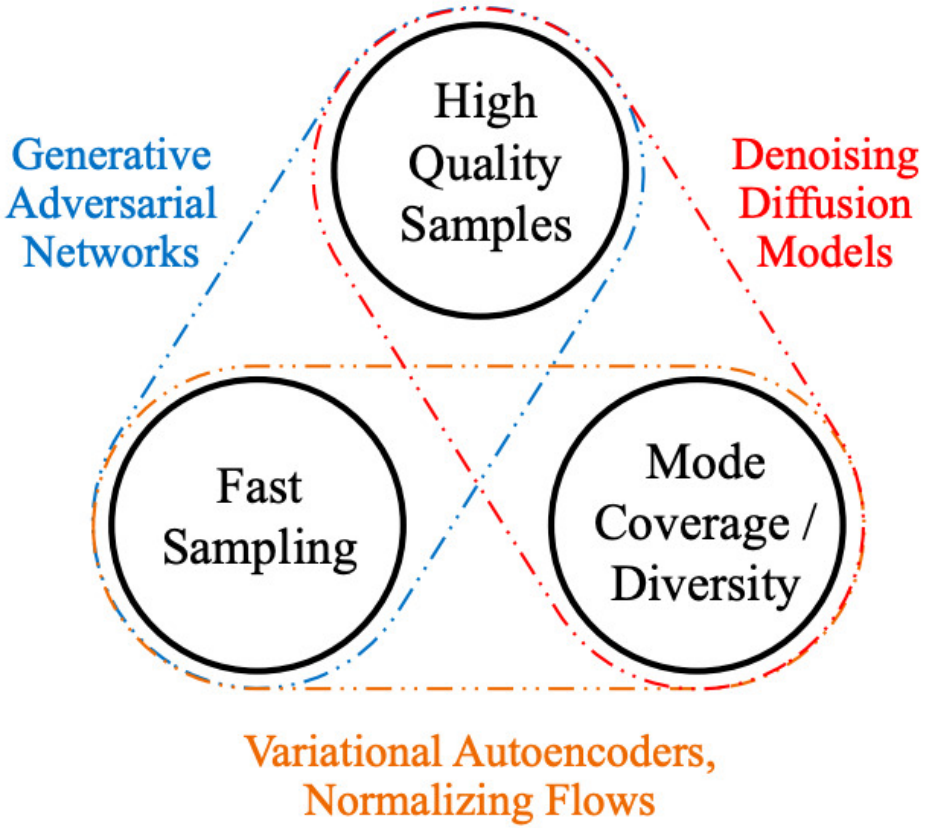
Сравнение зашумления при линейном (верняя строка) и косинуном (нижняя строка) расписаниях

Ускоряем DDPM

Генерация DDPM - много итераций (т.к. марковская цепь). Из-за этого процесс генерации DDPM требует серьёзных временных и вычислительных ресурсов, в чем сильно уступает тем же GAN-ам.

Рассмотрим улучшение алгоритма DDPM, которое работает в разы эффективнее при сравнимом качестве результата.

DDIM - Denoising Diffusion Implicit Models, диффузионные неявные модели денойзинга [9].



Трилемма генеративного обучения

Идея DDIM

Обобщим **прямой** процесс диффузии, используемый DDPM [5], который является марковским, на **немарковские**, для которых мы все еще можем разработать подходящие обратные генеративные марковские цепи.

Пересмотрим процесс **генерации** (inference), чтобы уменьшить количество итераций для создания изображения.

Ключевое наблюдение: целевая функция DDPM зависит только от $q(x_t|x_0)$, но не напрямую от совместного распределения $q(x_{1:T}|x_0)$. Поскольку существует множество совместных распределений с одинаковыми маргинальными распределениями $q(x_t|x_0)$, мы исследуем альтернативные (немарковские) процессы генерации.

Вывод DDIM

Рассмотрим семейство распределений вывода Q, индексированное вещественным вектором $\sigma \in \mathbb{R}^T_{\geq 0}$:

$$q_\sigma(x_{1:T}|x_0) := q_\sigma(x_T|x_0) \prod_{t=2}^{T} q_\sigma(x_{t-1}|x_t, x_0),$$

Рассмотрим:

$$q_\sigma(x_{t-1}|x_t, x_0) = \mathcal{N}\left(\underbrace{\sqrt{\overline{\alpha}_{t-1}}x_0 + \sqrt{1 - \overline{\alpha}_{t-1} - \sigma_t^2} \cdot \frac{x_t - \sqrt{\overline{\alpha}_t}x_0}{\sqrt{1-\overline{\alpha}_t}}}_{\mu_\sigma(x_t, x_0)}, \sigma_t^2 I \right)$$

где ϵ — над дробью.

Функция для среднего значения $\mu_\sigma(x_t, x_0)$ выбрана таким образом, чтобы гарантировать, что $q_\sigma(x_t|x_0) = \mathcal{N}(\sqrt{\overline{\alpha}_t}x_0, (1 - \overline{\alpha}_t)\mathbf{I})$ для всех t.

DDIM. Применение теоремы Байеса

Прямой процесс можно «развернуть» с помощью теоремы Байеса:

$$q_\sigma(x_t|x_{t-1}, x_0) = \frac{q_\sigma(x_{t-1}|x_t, x_0)q_\sigma(x_t|x_0)}{q_\sigma(x_{t-1}|x_0)},$$

В отличие от DDPM [5], прямой процесс здесь больше не является марковским, поскольку каждое x_t может зависеть как от x_{t-1}, так и от x_0. Величина σ управляет степенью стохастичности прямого процесса: когда $\sigma \to 0$, мы приходим к крайнему случаю, когда, зная x_0 и x_t для некоторого t, x_{t-1} становится известным и фиксированным.

DDIM. Генеративный процесс

Определим обучаемый генеративный процесс $p_\theta(x_{0:T})$, где каждое $p_\theta^{(t)}(x_{t-1}|x_t)$ использует знание $q_\sigma(x_{t-1}|x_t, x_0)$.

Имеем x_t $\Rightarrow$ Предсказываем x_0 $\Rightarrow$ Получаем из него x_{t-1} через обратное условное распределение $q_\sigma(x_{t-1}|x_t, x_0)$ (см. предыдущий слайд).

Вспомним, как мы получали образец (sample) x_t в DDPM:

$$x_t = \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

Выразим отсюда x_0 и обозначим его параметризованной функцией f:

$$x_0(x_t) = f_\theta^{(t)}(x_t) := \frac{x_t - \sqrt{1-\overline{\alpha}_t} \cdot \epsilon_\theta^{(t)}(x_t)}{\sqrt{\overline{\alpha}_t}}$$

DDIM. Генеративный процесс

Используя $p_\theta^{(t)}(x_{t-1}|x_t) \sim q_\sigma(x_{t-1}|x_t, x_0)$, можем определить генеративный процесс с фиксированным априорным распределением $p_\theta(x_T) = \mathcal{N}(0, I)$, причем:

$$p_\theta^{(t)}(x_{t-1}|x_t) \begin{cases} \mathcal{N}(f_\theta^{(1)}(x_1), \sigma_1^2 I), & \text{если } t = 1 \\ q_\sigma(x_{t-1}|x_t, f_\theta^{(t)}(x_t)), & \text{при других t} \end{cases}$$

Напомним, как определяется $q_\sigma(x_{t-1}|x_t, f_\theta^{(t)}(x_t))$:

$$q_\sigma(x_{t-1}|x_t, f_\theta^{(t)}(x_t)) = \mathcal{N}\left(\sqrt{\overline{\alpha}_{t-1}} f_\theta^{(t)}(x_t) + \sqrt{1 - \overline{\alpha}_{t-1} - \sigma_t^2} \cdot \frac{\overbrace{x_t - \sqrt{\overline{\alpha}_t} f_\theta^{(t)}(x_t)}^{\epsilon}}{\sqrt{1 - \overline{\alpha}_t}}, \sigma_t^2 I\right)$$

В этом выражении мы x_0 заменили на $f_\theta^{(t)}(x_t)$.
Мы добавляем гауссовский шум (с ковариацией $\sigma_1^2 I$) для случая $t = 1$, чтобы гарантировать, что генеративный процесс поддерживается повсюду.

DDIM. Шаг денойзинга

Из $p_\theta(x_{1:T})$ (предыдущий слайд) можно сгенерировать выборку x_{t-1} по выборке x_t через:

$$x_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(x_t - \frac{\sqrt{1-\alpha_t} \cdot \epsilon_\theta^{(t)}(x_t)}{\sqrt{\alpha_t}} \right)}_{\text{Предсказанный } x_0} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(x_t)}_{\text{шаг в направлении } x_t} + \underbrace{\sigma_t \epsilon_t}_{\substack{\text{случайный} \\ \text{шум}}}$$

где $\epsilon_t \sim \mathcal{N}(0, I)$ — это стандартный гауссовский шум, независимый от x_t.

Определим $\alpha_0 := 1$.

Изменили только генеративный процесс, используем ранее обученную по методу DDPM модель.

Разные генеративные процессы

Рассмотрим случаи для σ_t:

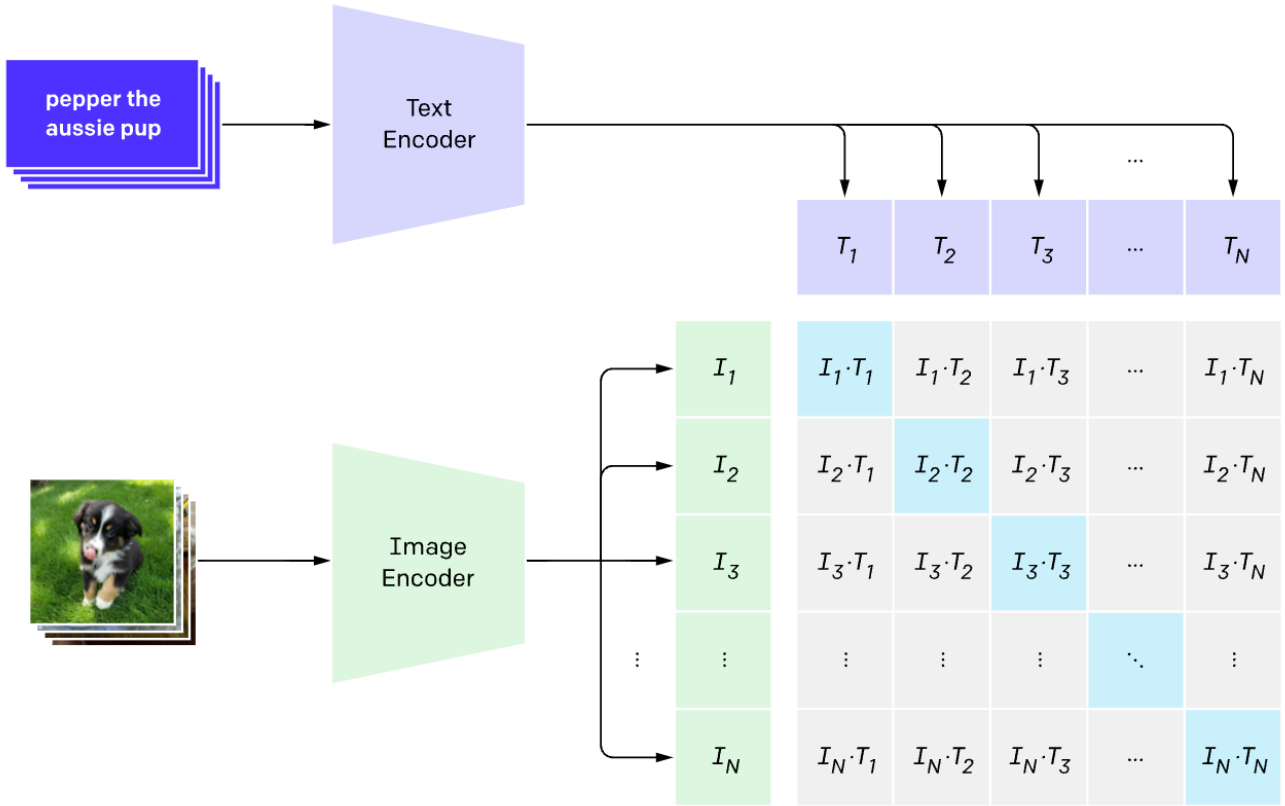
- $\sigma_t = \sqrt{\frac{(1-\overline{\alpha}_{t-1})}{(1-\overline{\alpha}_t)}}\sqrt{1-\alpha_t}$ - марковский процесс, **DDPM**.
- $\sigma_t = 0$ для всех t: генеративный процесс - **детерминированный** за исключением случая $t = 1$. Коэффициент перед шумом ϵ_t равен нулю.
  Полученная модель - диффузионная неявная модель денойзинга, **DDIM**. Диффузионная - потому что обучена с использованием процесса DDPM, несмотря на то, что обратный процесс (генерации) является детерминированным.

Text-to-Image диффузионные модели

Внутри Text-to-Image диффузионных моделей в связке работают несколько инструментов. Рассмотрим 2 основных:

- CLIP - отображает картинки и тексты в единое векторное пространство. Возьмем из него Text Encoder.
- U-Net - предсказывает шум $\varepsilon_\theta^{(t)}$ на картинке.

CLIP - Contrastive Language-Image Pre-training

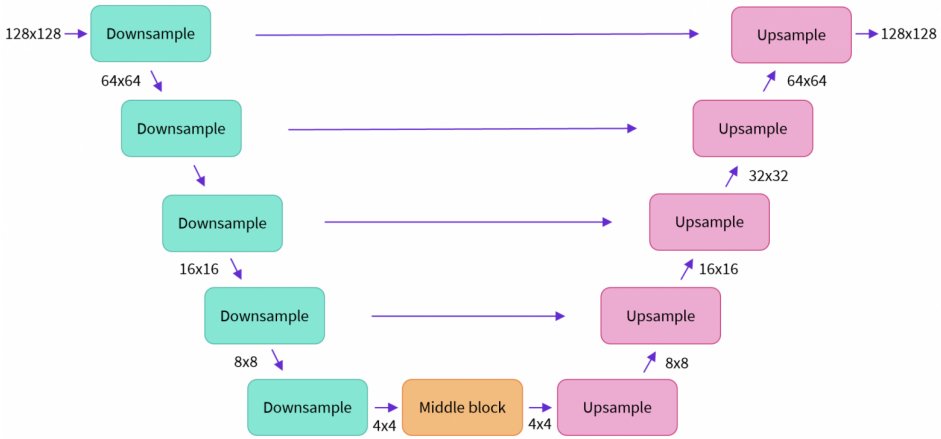


CLIP - модель от OpenAI, отображает картинки и тексты в единое векторное пространство. Из него берем *Text Encoder* - будет формировать внутреннее представление текстовой метки.

U-Net - сверточная сетка для сегментации картинок

U-Net = Энкодер + Декодер. На каждом уровне мы выделяем карту признаков различных объектов (формы, размеры, цвета и т.д.). При Text-to-Image генерации принимает на вход:

- Зашумленное изображение x_t
- *Text Embedding* - внутреннее представление текстового условия (промпта)
- *Time Embedding* - внутреннее представление номера шага t, модель обуславливается в том числе на него.

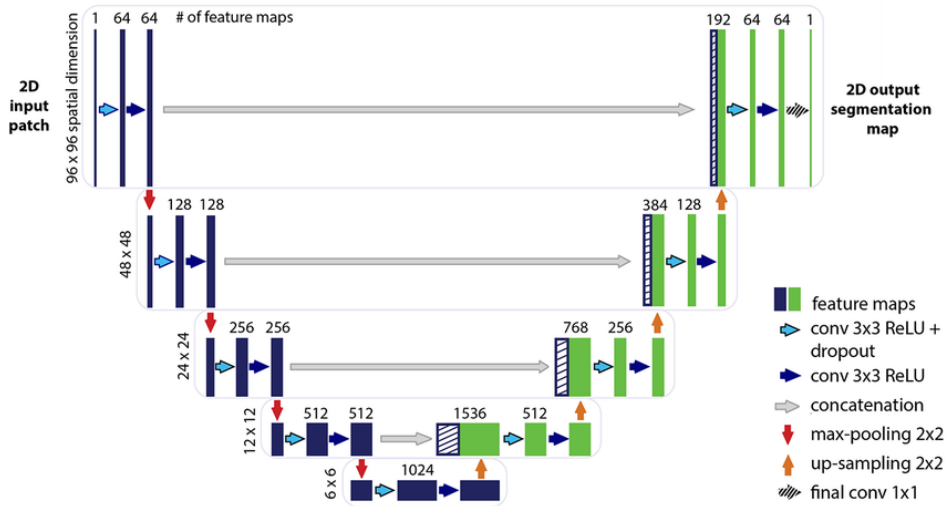


Упрощенная схема работы Unet

Особенности устройства U-Net

Основные блоки U-Net:

- *Residual Block* - два сверточных слоя с групповой нормализацией
- *Multi-Head Attention Block* - делает модель «внимательной к деталям» и к глобальным зависимостям на изображении
- *Down Block* = *ResBlock* + *AttnBlock*
- *Up Block* = *ResBlock* + *AttnBlock*
- *Middle Block* = *ResBlock* + *AttnBlock* + *ResBlock*



Подробное устройство U-Net

Латентные диффузионные модели

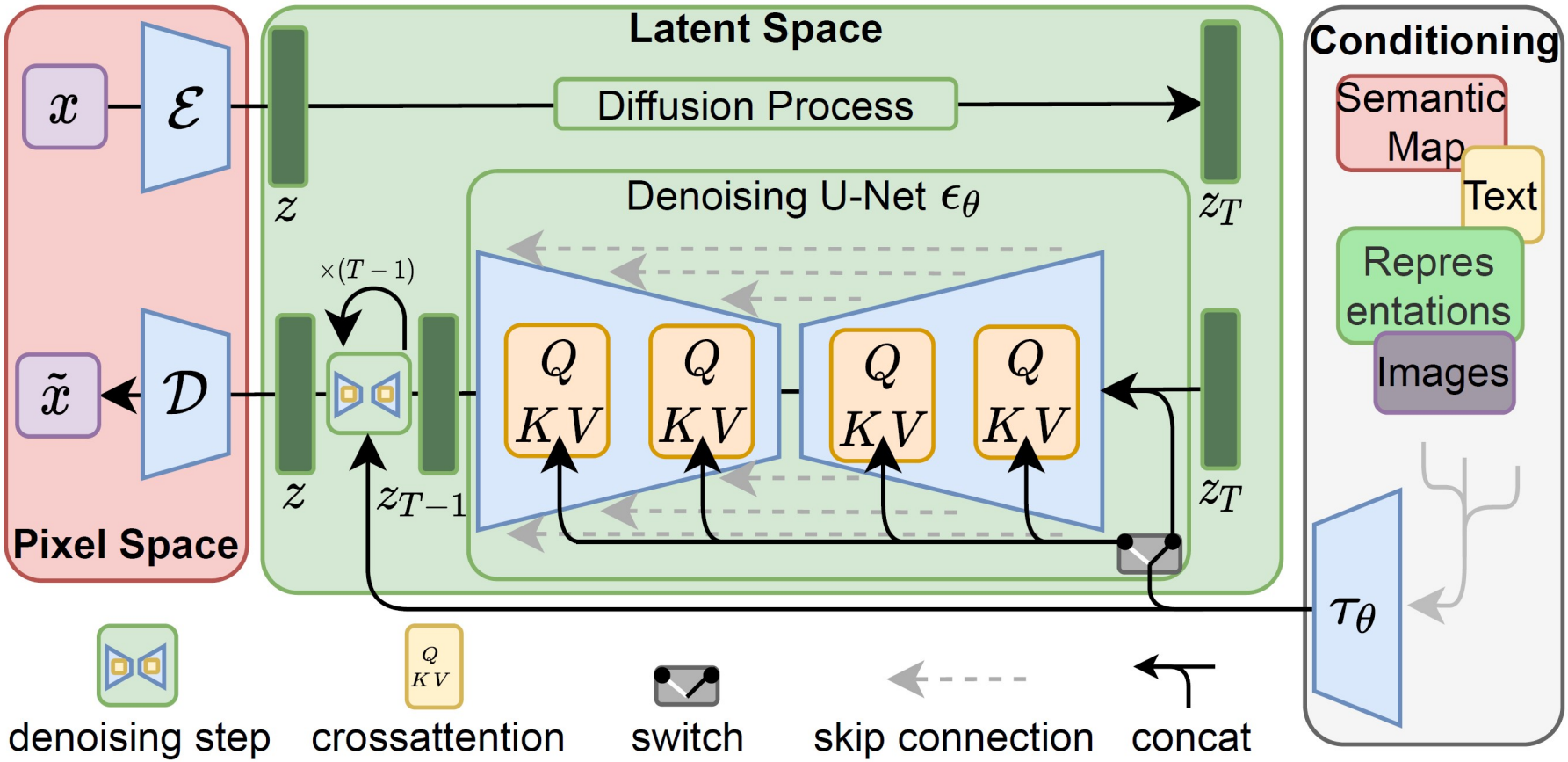


Схема реализации LDM

Промежуточные итоги

Изученный материал:

- Вывели формулы для переходных распределений в диффузионном процессе;
- Оценили снизу логарифмическое правдоподобие через $ELBO$;
- Рассмотрели алгоритмы обучения и генерации DDPM;
- Изучили DDIM как метод ускорения генерации при помощи диффузионных моделей;
- Обсудили схему реализации диффузионных моделей для генерации изображений.

Полезные ссылки

- **Методичка по диффузионным моделям от ШАД**: https://education.yandex.ru/handbook/ml/article/diffuzionnye-modeli
- **Объяснение статьи DDPM (на английском)**: https://www.youtube.com/watch?v=HoKDTa5jHvg
- **Под капотом Stable Diffusion**: https://habr.com/ru/articles/688204/

Применение диффузионных моделей для **защиты** систем компьютерного зрения

- **Схема обучения робастного классификатора** с использованием сгенерированных данных [10];
- **Text-guided** методы диффузионного редактирования:
  1. SDEdit [7];
  2. Prompt-to-Prompt [4];
  3. IntructPix2Pix [1];
  4. Null-text inversion [8].
- Методы диффузионного редактирования **в семантическом латентном пространстве**:
  1. Discovering interpretable directions... [3];
  2. Concept Sliders [2].

Робастное обучение с использованием сгенерированных данных

Робастность - устойчивость к выбросам и помехам.

Рассмотрим метод повышения робастности нейронных сетей к возмущениям, ограниченным по l_p-норме (l_p norm-bounded perturbations), основанный на использовании **сгенерированных данных** [10].

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}},$$

где $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, а $p \geq 1$.

Цель: обучить модель на исходных (чистых) данных, чтобы она могла создавать синтетические образцы.

Эти сгенерированные данные при объединении с реальными образцами должны сокращать разрыв в робастности с моделями, обученными на дополнительных реальных данных.

Формулировка задачи

Целевая задача заключается в минимизации *adversarial risk*, определяемого следующим образом:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim D} \left(\max_{\delta \in S} [f(x+\delta;\theta) \neq y] \right),$$

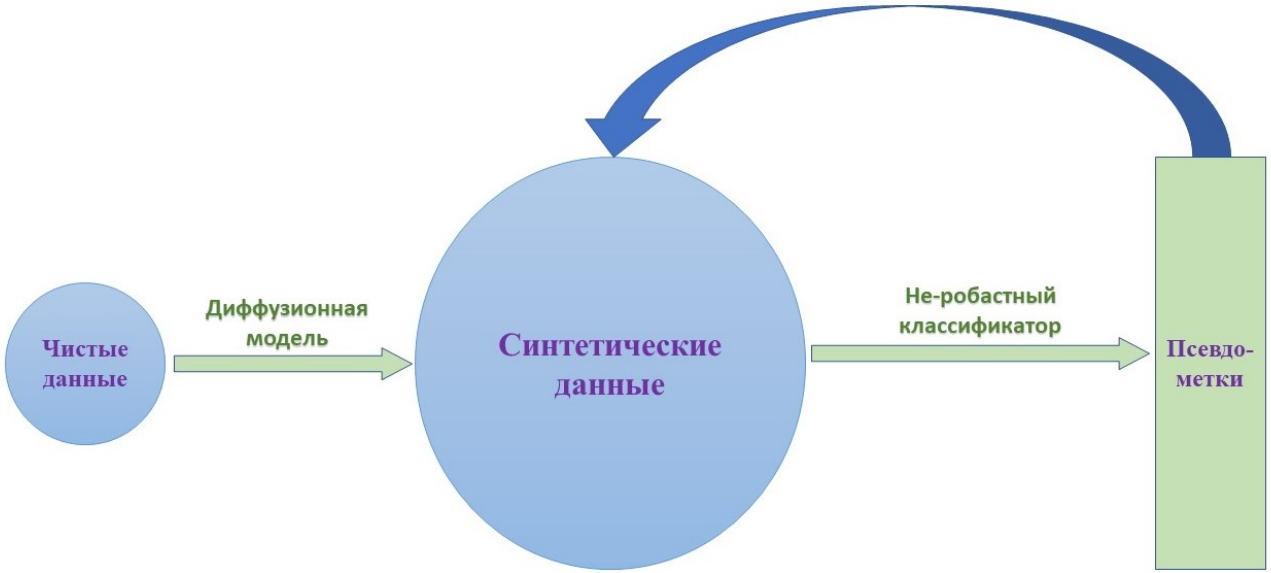
где

- D — распределение данных x,
- y - метки данных,
- $[\cdot]$ - скобочная нотация Иверсона, $[P] = \begin{cases} 1, & \text{если } P \text{ истинно;} \\ 0, & \text{иначе.} \end{cases}$
- $f(\cdot;\theta)$ — модель с параметрами θ,
- S — множество допустимых возмущений.
- δ — допустимое значение возмущения.

Для возмущений, ограниченных по l_p-норме, $S_p = \{\delta : \|\delta\|_p \leq \epsilon\}$.

Описание метода [10]

- **Обучение генеративной модели** для создания синтетических данных, приближенных к исходному распределению.
- **Применение не-робастного классификатора** для присвоения псевдометок (pseudo-labeling) этим данным.
- **Совместное обучение** на исходных и сгенерированных данных с целью улучшения робастности к возмущениям.



Формулировка метода

Оптимизационное уравнение для обучения робастного классификатора $f(\cdot;\theta)$:

$$\theta^* = \arg\min_{\theta} \Bigg(\alpha \cdot \underbrace{\mathbb{E}_{(x,y)\in D_{\text{train}}} \left(\max_{\delta\in S} \mathrm{L}_{\text{ce}}(f(x+\delta;\theta), y) \right)}_{\text{исходные данные}} + (1-\alpha) \cdot \underbrace{\mathbb{E}_{x\sim\hat{D}} \left(\max_{\delta\in S} \mathrm{L}_{\text{ce}}(f(x+\delta;\theta), f_{\text{NR}}(x)) \right)}_{\text{сгенерированные данные}} \Bigg)$$

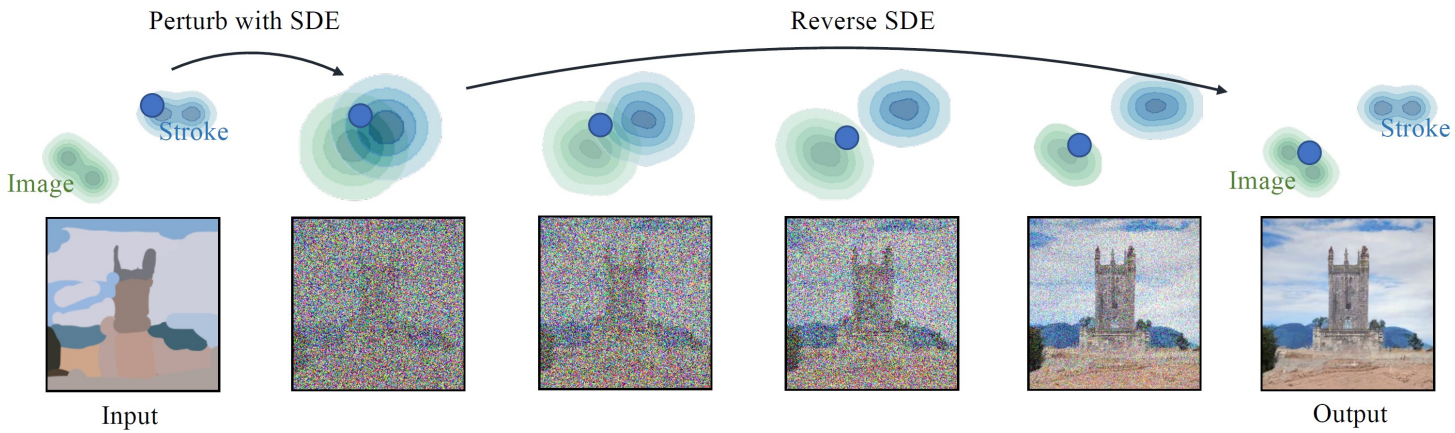
- α — фактор смешивания, определяющий долю данных из исходного набора,
- D_{train} — исходный тренировочный набор,
- $\hat{D}$ — сгенерированный набор данных,
- f_{NR} — не-робастный классификатор,
- L_{ce} — функция кросс-энтропии.

SDEdit - Stochastic Differential Editing

Метод стохастического дифференциального редактирования (SDEdit) [7] использует диффузионную модель для редактирования изображения путем добавления шума и последующего обратного денойзинга:

$$\text{SDEdit}(x, K) = R_\varphi(\dots R_\varphi(R_\varphi(x_K, K), K-1)\dots, 0),$$

где сначала выполняется K шагов диффузии, а затем K шагов обратного денойзинга. Такой процесс можно использовать, например, для очистки скрытых образцов от шумов.



Prompt-to-Prompt: задачи метода

Цель: редактирование изображений **только через изменение текстового промпта**, без масок или дополнительных входов.

Проблема: даже небольшое изменение промпта в диффузионных моделях приводит к полной потере исходной структуры изображения.

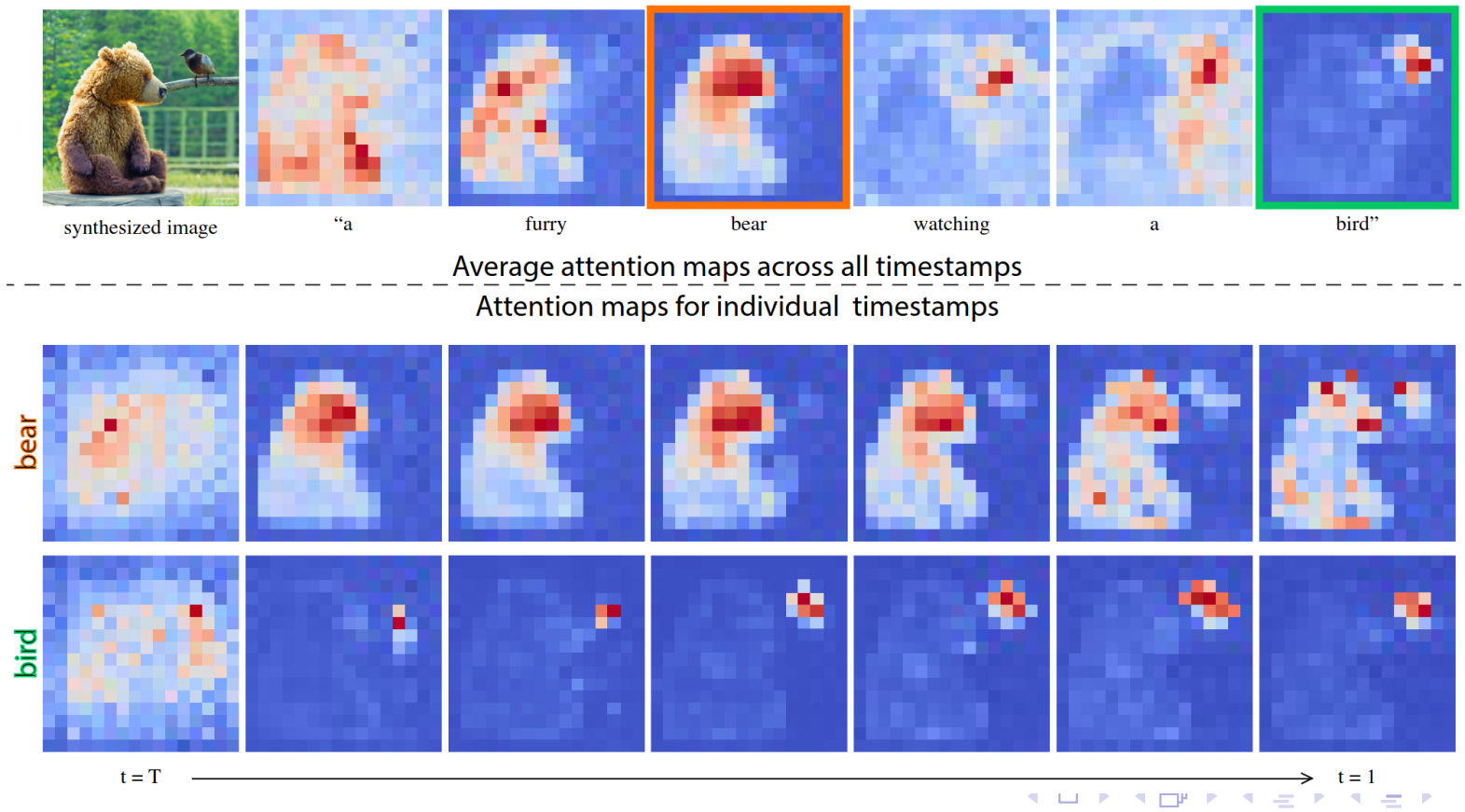
A photo of a butterfly on grass

A photo of a butterfly on a violin

Prompt-to-Prompt: основная идея

Ключевое наблюдение: **кросс-внимание** (cross-attention) связывает токены промпта с пространственными областями изображения.

Решение: контролировать процесс генерации, **инжектируя** или модифицируя карты кросс-внимания из исходного изображения.



Кросс-внимание в диффузионных моделях

На каждом шаге диффузии модель предсказывает шум, используя U-Net с **кросс-вниманием**.

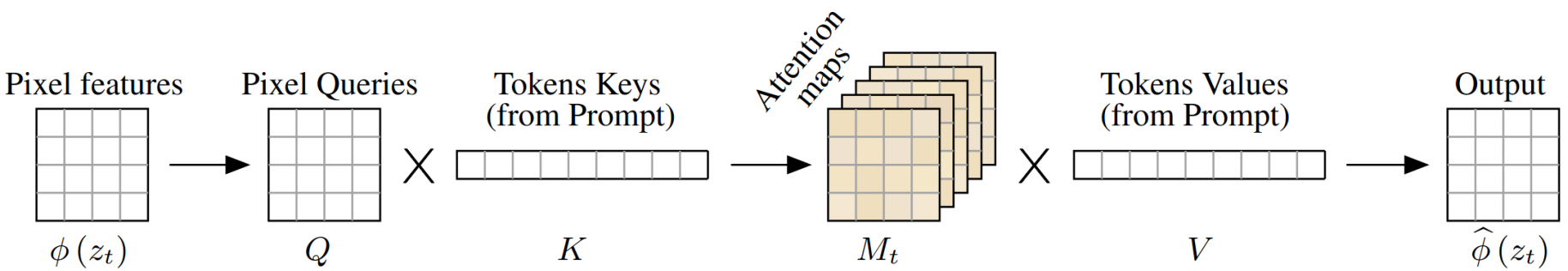
Кросс-внимание вычисляется как:

$$M = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d}}\right), \quad \hat{\varphi}(z_t) = MV$$

где Q — признаки пикселей, K, V — эмбеддинги токенов промпта.

Карта M_{ij} показывает, насколько пиксель i «внимает» токену j.

Эти карты определяют **геометрию и композицию** изображения уже на ранних шагах диффузии.



Text to Image Cross Attention

Общий алгоритм Prompt-to-Prompt

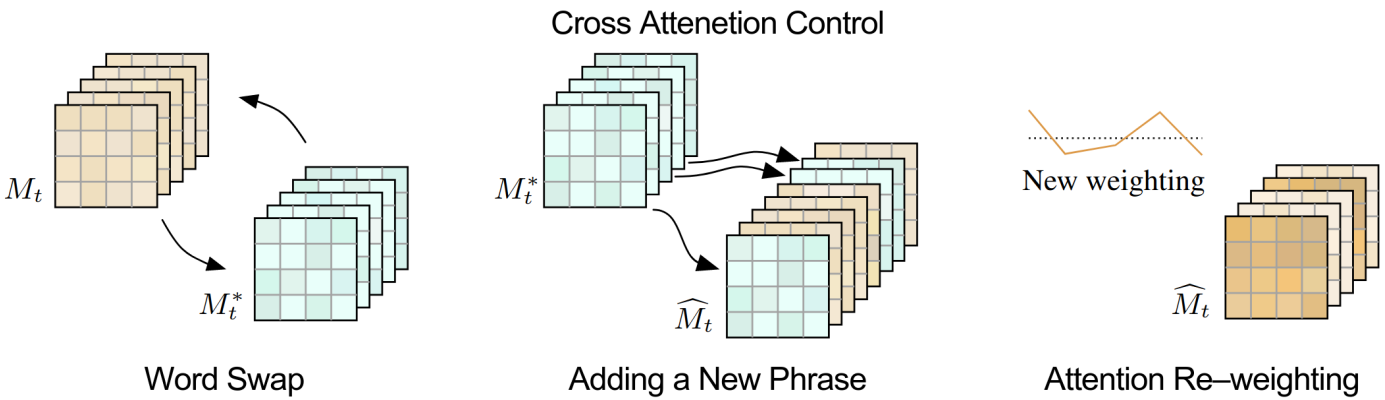
Фиксируется случайное начальное состояние z_T (seed).

Одновременно запускаются два процесса генерации:

- Исходный промпт $P \Rightarrow \mathrm{I}$
- Изменённый промпт $P^* \Rightarrow \mathrm{I}^*$
- На каждом шаге t вычисляются карты внимания M_t и M_t^*.
- Применяется функция редактирования: $\hat{M}_t = \mathrm{Edit}(M_t, M_t^*, t)$.
- В генерацию I^* «вставляется» $\hat{M}_t$, сохраняя значения V от P^*.

Типы текстового редактирования

- **Замена слова** (Word Swap):
  Пример: $P = \text{«велосипед»} \Rightarrow P^* = \text{«машина»}$.
- **Добавление фразы** (Adding a New Phrase):
  Пример: $P = \text{«замок у реки»} \Rightarrow P^* = \text{«детский рисунок замка у реки»}$.
- **Регулировка влияния слова** (Attention Re-weighting):
  Масштабирование карты конкретного токена: $M_{:,j^*} \leftarrow c \cdot M_{:,j^*}$, $c \in [-2, 2]$.
  Позволяет плавно усиливать/ослаблять признак (например, «пушистость»).



Преимущества и ограничения

Преимущества:

- Не требует обучения, масок или оптимизации.
- Поддерживает локальные и глобальные правки через текст.
- Сохраняет структуру и композицию исходного изображения.
- Применим и к синтетическим, и (через инверсию) к реальным изображениям.

Ограничения:

- Инверсия реальных изображений может быть неточной.
- Низкое разрешение карт внимания (из-за bottleneck в U-Net).
- Невозможно перемещать объекты в другую часть изображения.

Метод InstructPix2Pix: описание

- Использует инструкции, а не полные описания результата.
- Stable Diffusion v1.5 дообучена на сгенерированных парах изображений и текстов.


“Make it Paris”


“Make it Hong Kong”


“Make it Manhattan”


“Make it Prague”


“Make it evening”


“Put them on roller skates”


“Turn this into 1900s”


“Make it underwater”

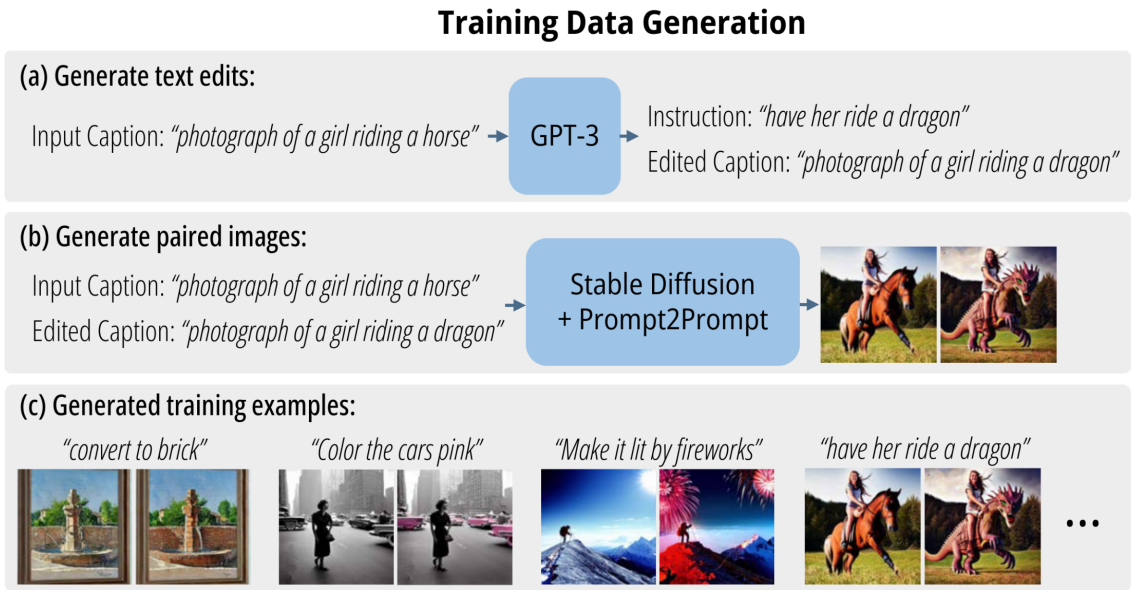
InstructPix2Pix: формирование обучающего набора

- Ручная разметка (700 примеров): изображения + исходные подписи + инструкции редактирования + новые подписи.
- GPT-3 для генерации 450 тыс.+ триплетов:

$$\boxed{\text{исходная подпись}} \Rightarrow \boxed{\text{инструкция редактирования}} + \boxed{\text{новая подпись}}$$

- Stable Diffusion + Prompt-to-Prompt:

$$\boxed{\text{подпись до}} + \boxed{\text{подпись после}} \Rightarrow \boxed{\text{изображение до}} + \boxed{\text{изображение после}}$$



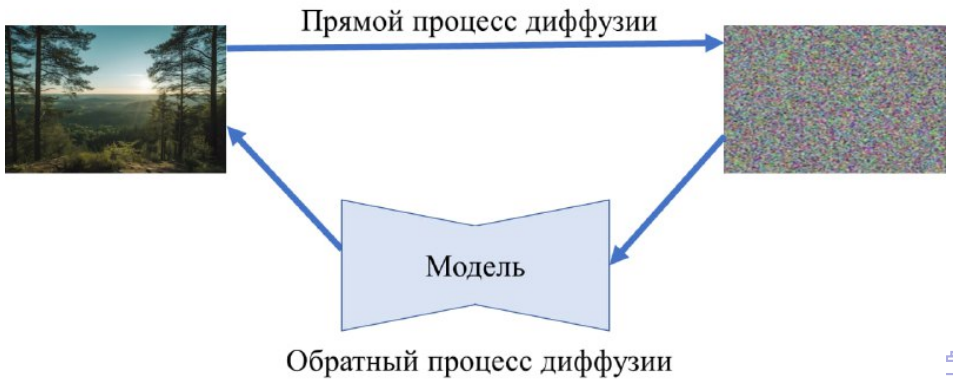
Постановка задачи Null-text inversion [8]

Глобальная цель – отредактировать исходное изображение, имея исходную и новую текстовые метки.

Займемся **инверсией** изображения $\mathcal{I}$, то есть получением как можно более точного представления реальной картинки $\mathcal{I}$ в латентном пространстве (в данном случае – получение соответствующей шумовой картинки).

Качество инверсии проверяется с помощью **реконструкции** исходного изображения $\mathcal{I}$ по исходной текстовой метке P.

Задача - сохранить «редактируемость» получаемого изображения.



Возможности Null-text inversion [8]

Input caption: “A baby wearing a blue shirt lying on the sofa.”

Редактирование изображения при помощи изменения текстовой метки

Возможности Null-text inversion

Input caption: “A man in glasses eating a doughnut in the park.”

Редактирование изображения при помощи изменения текстовой метки

Диффузионные модели с текстовым управлением

Диффузионные модели, управляемые текстом, учатся находить исходное изображение z_0 по случайному шуму z_T и текстовой метке $\mathcal{P}$.

На каждом шаге цепи модель учится предсказывать шум ε_θ.

Таким образом, на этапе обучения стоит задача минимизации функции:

$$\min_\theta ||\varepsilon - \varepsilon_\theta(z_t, t, \mathcal{C})||_2^2$$

где $\mathcal{C} = \psi(\mathcal{P})$ - эмбеддинг (некое внутреннее представление) текстовой метки.

Детали реализации

Детерминированный DDIM-sampling:

$$z_{t-1} = \sqrt{\frac{\alpha_{t-1}}{\alpha_t}} z_t + \left[\sqrt{\frac{1}{\alpha_{t-1}} - 1} - \sqrt{\frac{1}{\alpha_t} - 1} \right] \cdot \varepsilon_\theta(z_t, t, \mathcal{C})$$

Особенности Stable Diffusion:

- Энкодера при прямом процессе диффузии, который на входе «перегоняет» картинку в латентное пространство: $z_0 = E(x_0)$
- Декодера при обратном процессе диффузии, который на выходе «перегоняет» картинку из латентного пространства в нормальное изображение: $x_0 = D(z_0)$

Входную текстовую метку $\mathcal{P}$ можно генерировать при помощи модели генерации субтитров.

Два слова о Classifier-free guidance

Classifier-free guidance (CFG) [6] сочетает условную и безусловную генерации, что соответствует вероятностям $p(x|y)$ и $p(x)$.
Введем обозначения:

- ω - параметр (весовой коэффициент), отвечающий за «соотношение» условной и безусловной генераций в CFG
- $\varnothing = \psi(\text{""})$ - эмбеддинг нулевого текста (null-text embedding)

Тогда предсказание шума в Classifier-free guidance выглядит так:

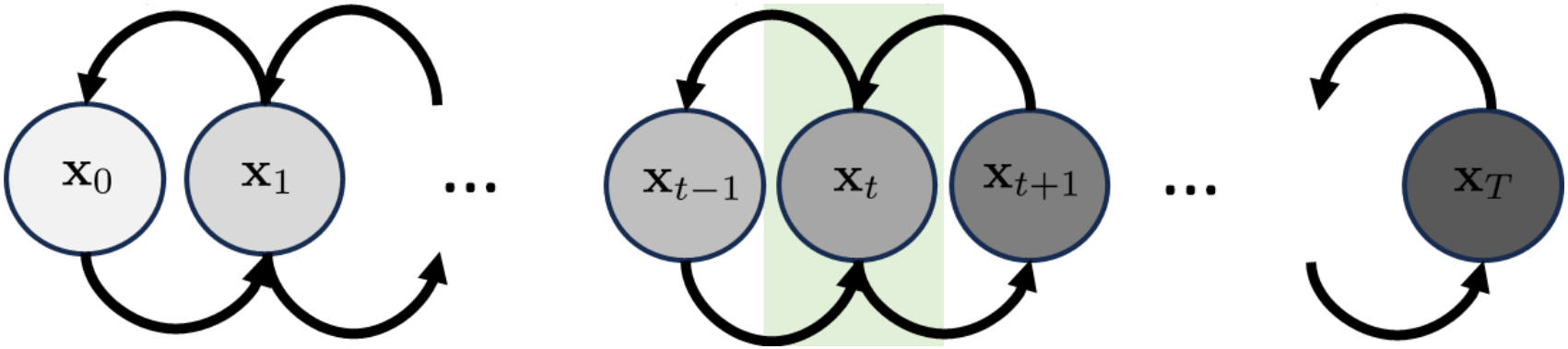
$$\widetilde{\varepsilon}_\theta(z_t, t, \mathcal{C}, \varnothing) = \omega \cdot \underbrace{\varepsilon_\theta(z_t, t, \mathcal{C})}_{\text{условная генерация}} + (1-\omega) \cdot \underbrace{\varepsilon_\theta(z_t, t, \varnothing)}_{\text{безусловная генерация}}$$

Для Stable Diffusion дефолтное значение $\omega = 7.5$

«Разворот» инверсии DDIM

В условиях малых шагов цепи процесс генерации DDIM может быть «развернут» в обратном направлении (от z_0 к z_T):

$$z_{t+1} = \sqrt{\frac{\alpha_{t-1}}{\alpha_t}} z_t + \left[\sqrt{\frac{1}{\alpha_{t-1}} + 1} - \sqrt{\frac{1}{\alpha_t} - 1} \right] \cdot \varepsilon_\theta(z_t, t, \mathcal{C})$$



«Ключевая» (pivotal) инверсия

Инверсия DDIM неплохо работает для безусловного случая ($\omega = 0$).

$$\boxed{\text{Накладываем текстовое условие } P} \Rightarrow \boxed{\text{ставим } \omega > 1}$$

(сильно опираемся на условную составляющую в CFG)

Экспериментально выяснено:

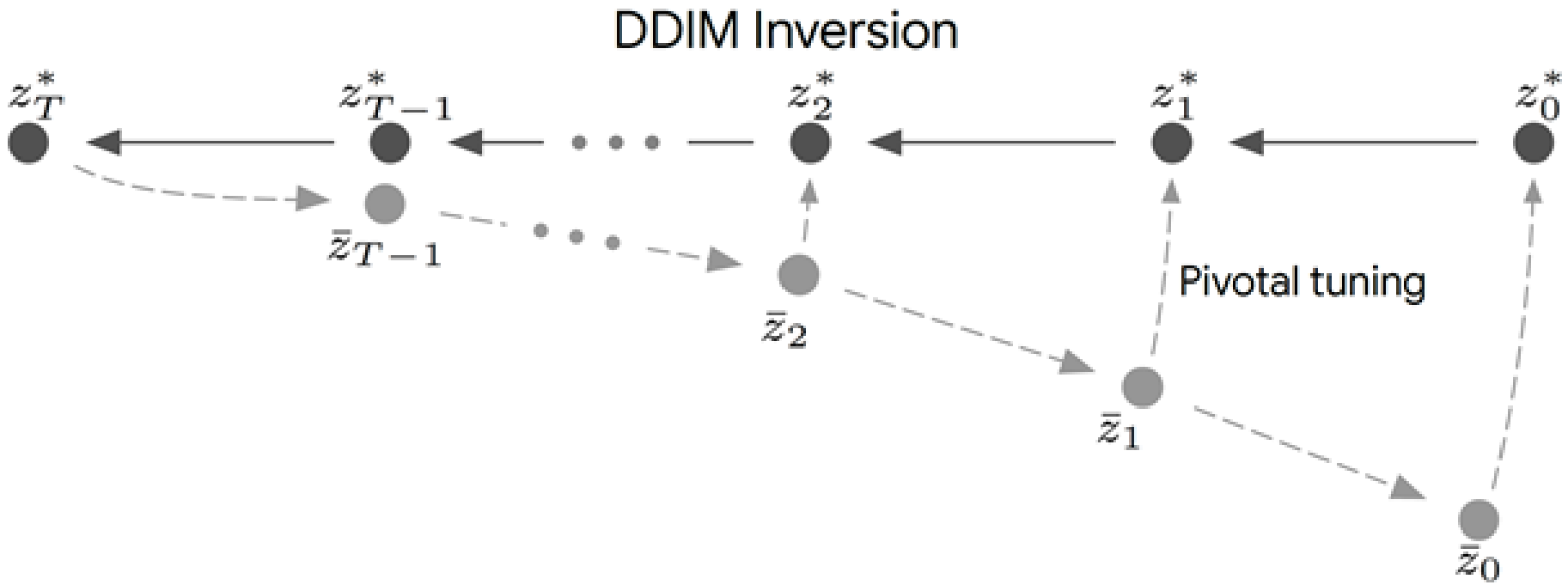
- $\omega > 1$ - накопление ошибки и ухудшение качества инверсии
- $\omega = 1$ - приемлемое качество реконструкции и хорошую «редактируемость»

Назовем траекторию инверсии при $\omega = 1$ **«ключевой траекторией»**, и будем считать ее опорой для дальнейшей оптимизации при $\omega = 7.5$ ($\omega > 1$).

Приближение к «ключевой траектории»

Хотим приблизить траекторию с $\omega = 7.5$ к «ключевой» ($\omega = 1$), то есть:

$$\min ||z^*_{t-1} - z_{t-1}||_2^2$$



Оптимизация траектории

Оптимизация нулевого текста

Оптимизируем эмбеддинг нулевого текста $\varnothing$.

Варианты:

- «Глобальная оптимизация нулевого текста» - когда мы оптимизируем один эмбеддинг для всех шагов (timestamps).
- «Оптимизация нулевого текста» = оптимизируем свой вариант нулевого эмбеддинга для каждого шага t.

  На выходе получаем оптимизированную последовательность $\{\varnothing_t\}_{t=1}^{T}$.

Алгоритм Null-text inversion [8]

Вход: Исходный текстовый эмбеддинг $\mathcal{C} = \psi(\mathcal{P})$, исходная картинка $\mathcal{I}$

- Установим $\omega = 1$
- Найдем «ключевую траекторию» $z_T^*, \ldots, z_0^*$ при помощи инверсии DDIM картинки $\mathcal{I}$

Алгоритм Null-text inversion [8]

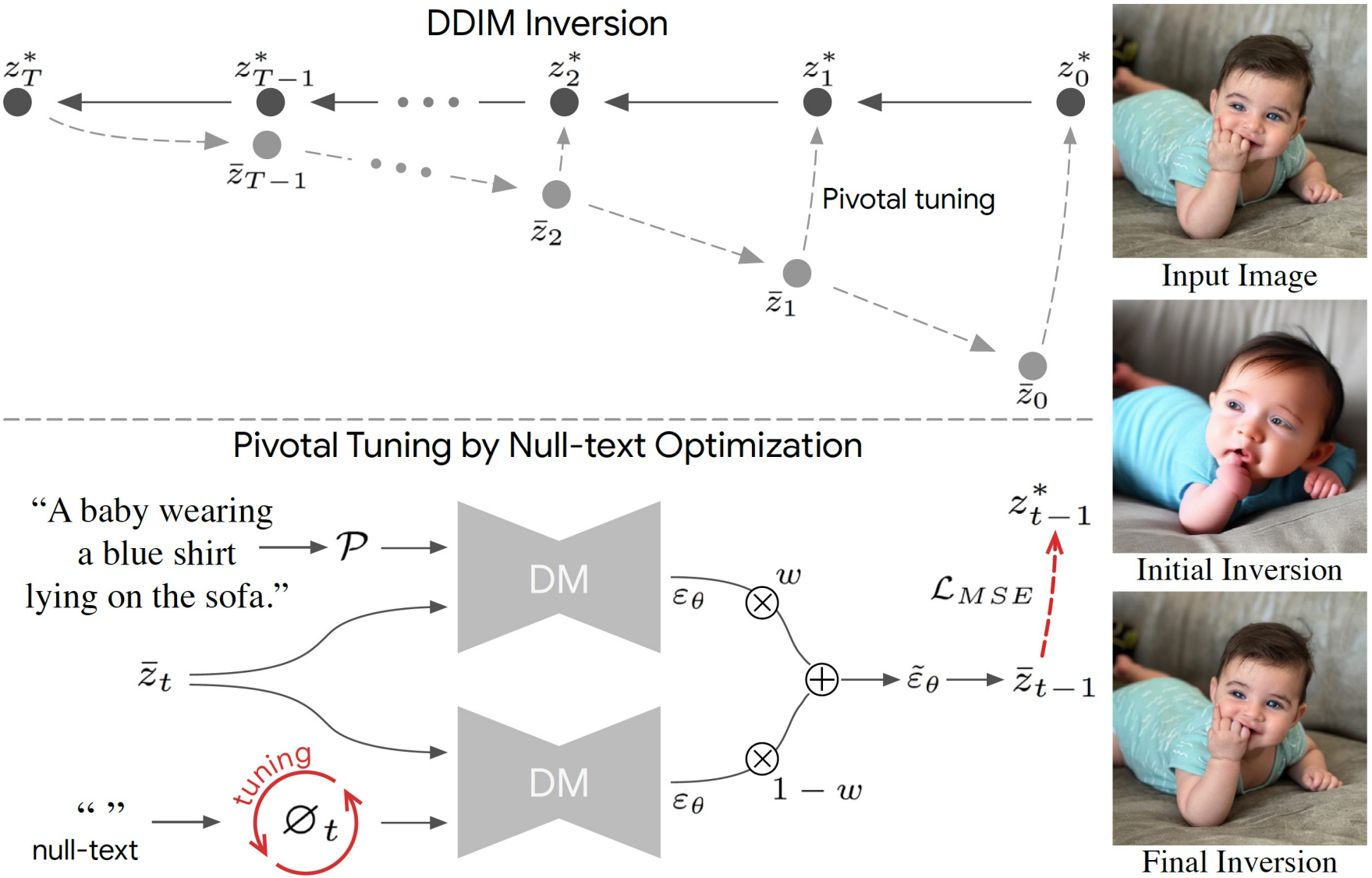
- Установим $\omega = 7.5$
- Начальная инициализация: $\overline{z}_T = z_T^*$, $\varnothing_T = \psi()$
- for $t = T, T-1, \ldots, 1$
  {
  - for $j = 0, \ldots, N-1$
    {
    - Градиентная оптимизация $\varnothing_t$:
      $\varnothing_t = \varnothing_t - \eta \nabla_{\varnothing} ||z_{t-1}^* - z_{t-1}(\overline{z}_t, \varnothing_t, \mathcal{C})||_2^2$

    }
  - $\overline{z}_{t-1} = z_{t-1}(\overline{z}_t, \varnothing_t, \mathcal{C})$,
    $\varnothing_{t-1} = \varnothing_t$

  }

Выход: шумовой вектор z_T и оптимизированная последовательность $\{\varnothing_t\}_{t=1}^T$.

Иллюстрация алгоритма



Интерпретируемые направления в латентном пространстве DM [3]

Цель: найти **интерпретируемые и дезентанглированные направления** в латентном пространстве *безусловных* диффузионных моделей (DMs).

Проблема: некоторые признаки изображений проблематично описать текстом.

Пример: описать текстом изменения на медицинском изображении.

Решение: использовать **h-space** — пространство активаций bottleneck-слоя U-Net на всех шагах диффузии.

Преимущества:

- Не требует CLIP, промптов, fine-tuning или изменения архитектуры.
- Поддерживает как глобальные, так и локальные правки.
- Работает с предобученными моделями «из коробки».

h-space: семантическое латентное пространство DM

h-space = $\{h_T, h_{T-1}, \ldots, h_1\}$ — набор bottleneck-активаций U-Net на каждом шаге диффузии.

Каждый $h_t \in \mathbb{R}^{C \times H \times W}$ (например, $512 \times 8 \times 8$).

Ключевые свойства h-space (**векторная арифметика**) [3]:

- Направление $\Delta h_{T:1}$ вызывает **один и тот же семантический эффект** на разных изображениях.
- Масштаб γ контролирует силу правки: $h^{\text{edit}} = h + \gamma \cdot v$.
- Аддитивность: комбинация направлений даёт комбинированные правки.
- Редактирование: вносим сдвиг Δh_t в процесс генерации (в обе компоненты P_t и D_t).

Схема формирования h-space

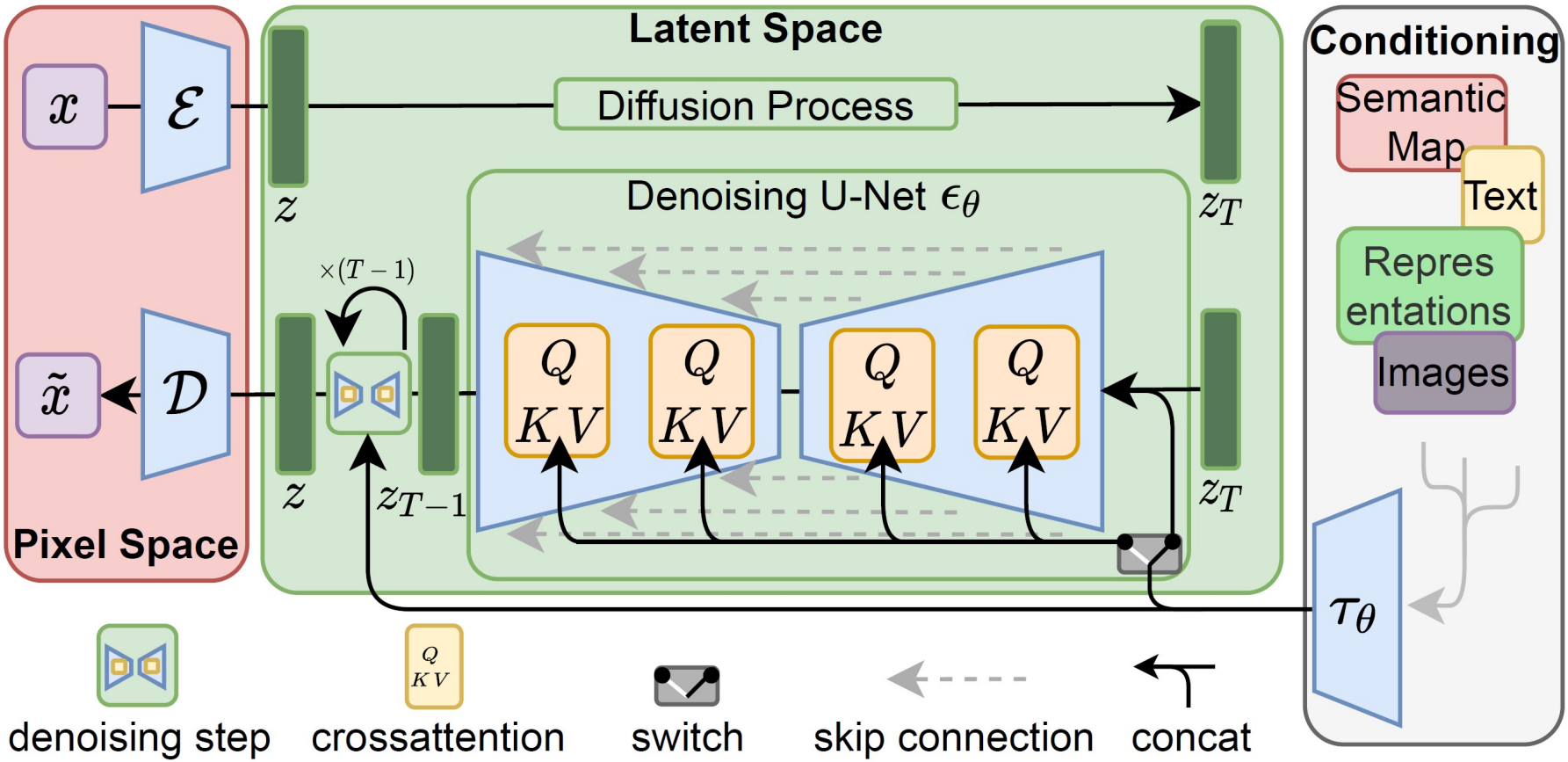
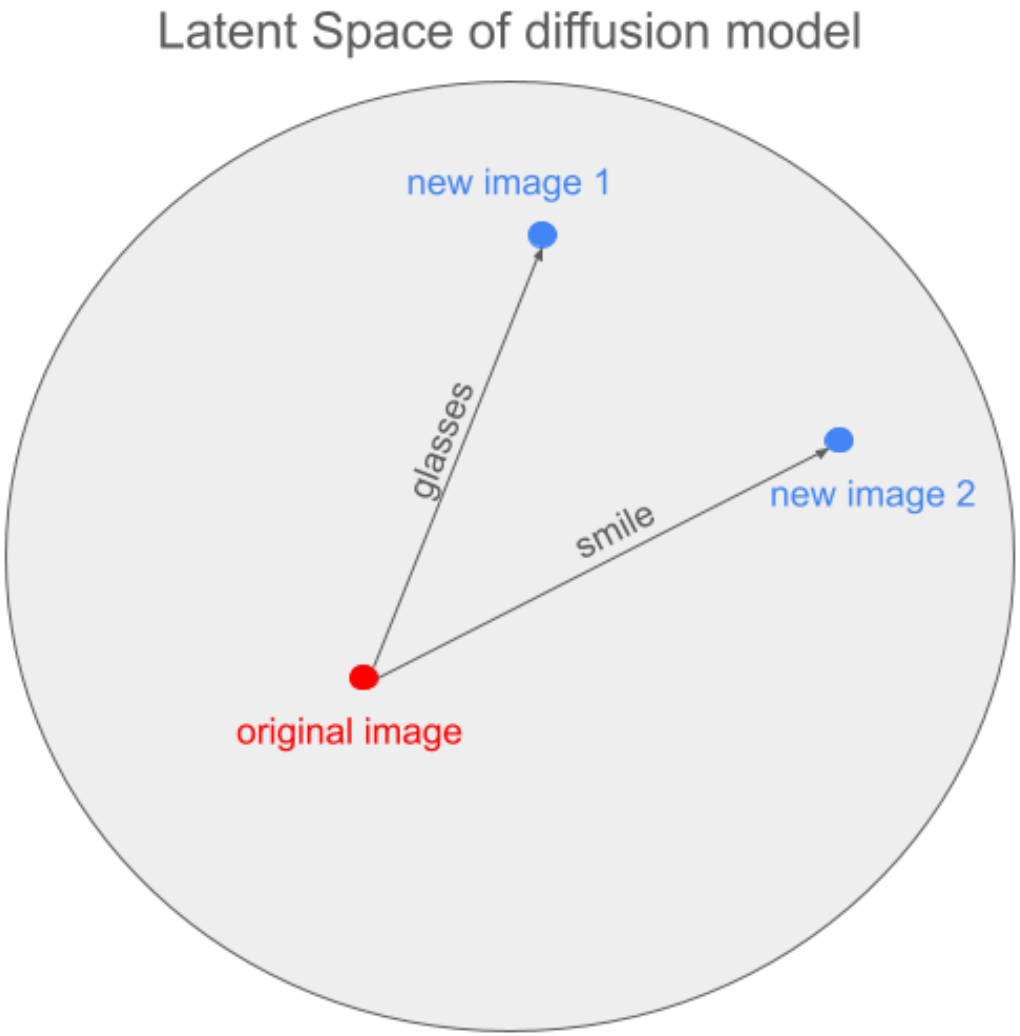


Схема реализации DM

Методы редактирования изображений в латентном пространстве: векторная арифметика



Unsupervised-исследование направлений

1. **PCA в h-space**:
   - Вычисляем главные компоненты по множеству случайных $h_{T:1}^{(i)}$.
   - Получаем глобальные семантические направления: возраст, пол, поза, улыбка.
   - Интерпретируемость появляется автоматически без аннотаций.
2. **Анализ Якобиана для одного изображения**:
   - Через сингулярное разложение Якобиана $J_t = \partial \varepsilon_\theta / \partial h_t$ находим направления, максимально изменяющие предсказание шума ε_θ.
   - Результат: локальные правки — открытие глаз, движение бровей и т.п.

Supervised исследование и «распутывание» направлений

Supervised метод:

- Генерируем пары изображений (x^-, x^+) с/без атрибута (например, «очки»).
- Вычисляем направление как среднюю разность:

$$v = \frac{1}{n}\sum_{i=1}^{n}(h_i^+ - h_i^-)$$

- Атрибуты можно получать от предобученного классификатора (например, CelebA).

«Распутывание» направлений:

- Если направление v_1 влияет и на нежелательный атрибут v_2, проецируем:

$$v_{1\perp 2} = v_1 - \frac{\langle v_1, v_2\rangle}{\|v_2\|^2}v_2$$

Преимущества и ограничения

Преимущества:

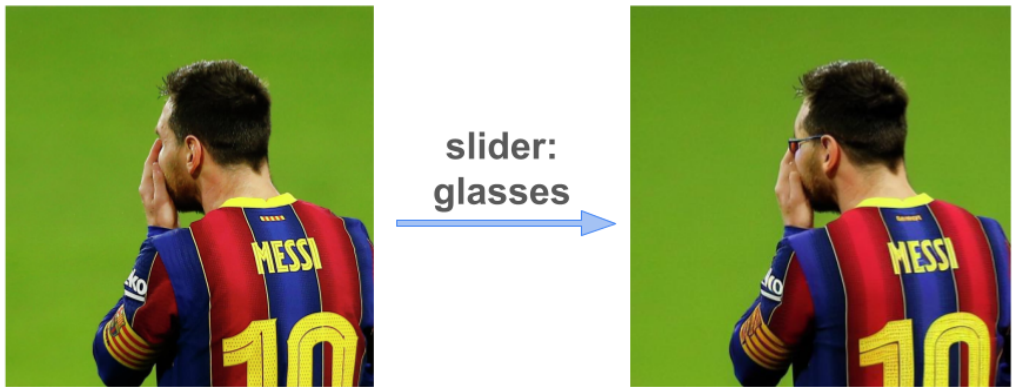
- Работает с *безусловными* DM без текста или CLIP.
- Поддерживает глобальные и локальные правки.
- Направления переносятся между изображениями.
- Простая линейная алгебра — нет оптимизации и обучения.
- Эффективно даже при малом числе примеров (в supervised случае).

Ограничения:

- Лучше всего работает на структурированных доменах (например, лица).
- На неструктурированных данных (церкви, комнаты) PCA даёт менее интерпретируемые направления.
- Требует доступа к внутренним активациям U-Net (но не модификации модели).

Метод Concept Sliders [2]

- позволяет создавать интерпретируемые слайдеры **для точного управления атрибутами изображений**;
- находит низкоранговые направления в латентном пространстве ДМ, соответствующие конкретным концептам;
- слайдеры обучаются **на небольшом наборе текстовых подсказок или пар изображений**;
- слайдеры реализованы как LoRA-адаптеры, которые можно включать во время инференса без изменения основной модели.



Методы атаки систем компьютерного зрения

Рассмотрим следующие типы атак:

- **Нормированные атаки** - Fast Gradient Sign Method (FGSM) и Projected Gradient Descent (PGD), ограничивают норму l_p.

  Проблема: вредоносные картинки расходятся с естественным распределением данных $\Rightarrow$ изображения могут быть очищены от атаки.

- **Семантические атаки** - например: генерация вредоносных картинок в HSV пространстве, поворот 2D изображения, изменение его яркости и т.д.

  Проблема: неэффективны.

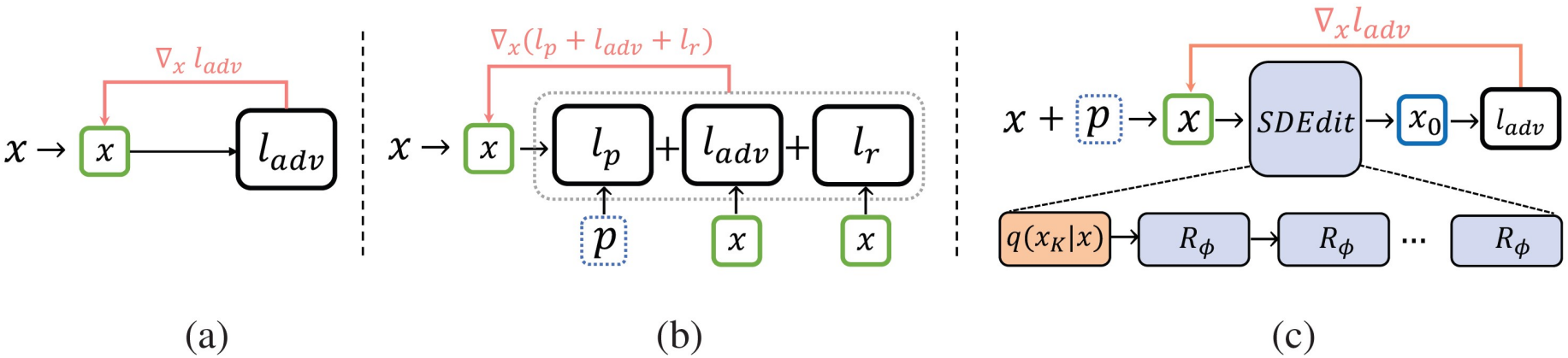
- **Кастомизированные атаки с использованием естественного стиля** - AdvCAM и AdvArt, стараются соханить стилевое соответствие образцов (используют состязательные методы генерации).

  Проблема: нужна тонкая подстройка и результат может оказаться нереалистичным.

- **Атаки в физическом мире** - осложняются факторами поворота, освещения, расстояния обзора и т.д.

  Проблема: Неестественность изображений.

Градиентные методы генерации атакующих изображений



(a) (b) (c)

- a) Традиционная градиентная генерация
- b) Кастомизированная градиентная генерация: добавлен l_p - loss стиля (отличие от текствого промпта p) и l_r - loss реалистичности
- c) Diff-PGD framework, о котором пойдет речь

Подход PGD для глобальной атаки

Введем обозначения:

- x - чистое изображение;
- y - текстовая метка;
- $f_\theta(x)$ - выход целевой модели классификации, параметризованной θ.

Задача: создать атакующий образец x_{adv}, способный обмануть модель классификации.

Традиционные методы (PGD) используют градиент функции потерь:

$$g = \nabla_x l(f_\theta(x), y)$$

где l — функция потерь (loss).

Шаг обновления в PGD с параметром η и количеством итераций n записывается так:

$$x^{t+1} = P_{B_\infty(x,\epsilon)}\left[x^t + \eta \cdot \text{sign}(\nabla_{x^t} l(f_\theta(x^t), y))\right],$$

где $P_{B_\infty(x,\epsilon)}(\cdot)$ — оператор проекции на шар ℓ_∞ с радиусом ϵ.

Да кто такая эта ваша проекция на шар?

- x - оригинальное изображение
- x_{adv} - атакующее изображение

Шар l_∞ определяется условием:

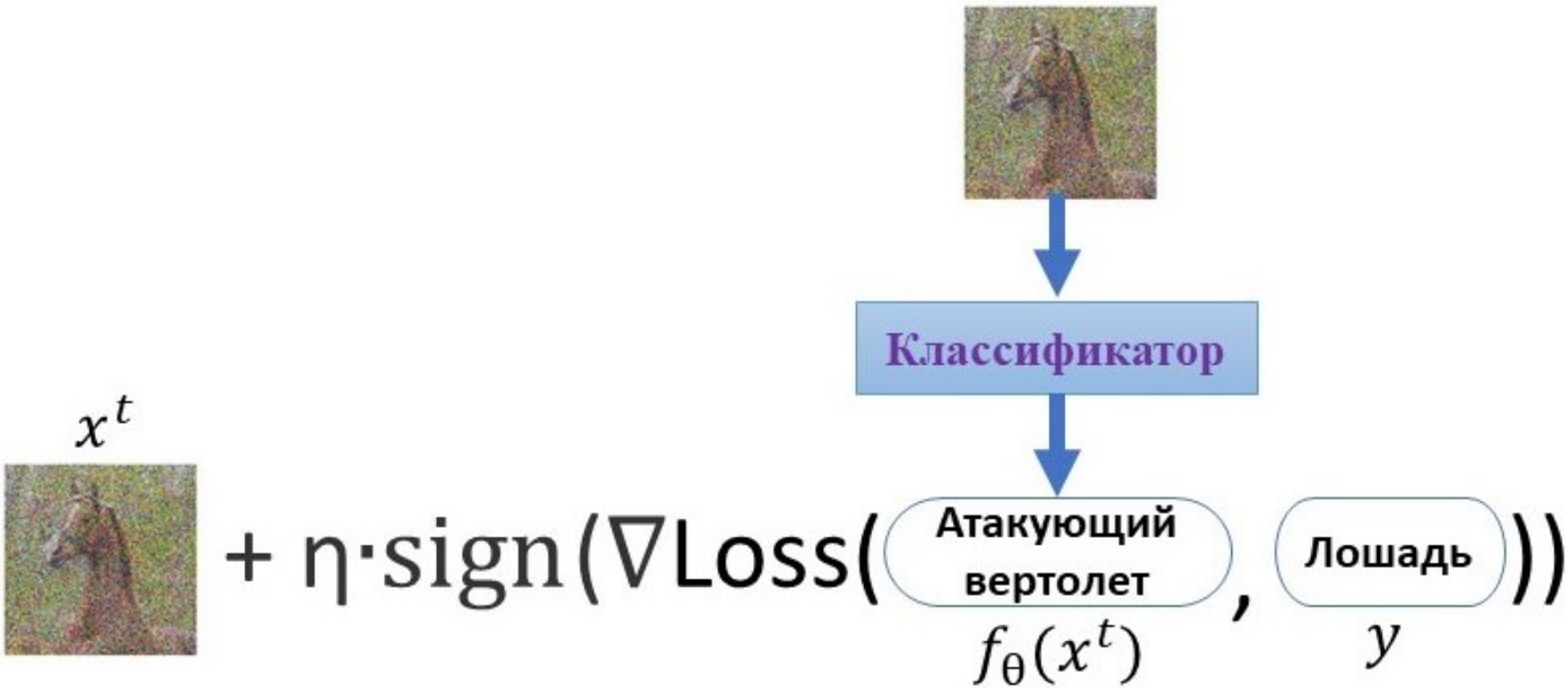
$$\|x_{adv} - x\|_\infty \leq \varepsilon$$

Проектирование на этот шар делается для того, чтобы сохранить невидимость изменений для человека, ограничивая их интенсивность.

Таким образом, если новый градиентный шаг, который стремится сделать изображение «более атакующим», выходит за пределы этого шара, его компоненты подлежат ограничению (т. е. проекции) так, чтобы максимальное отклонение в любом пикселе не превышало ε.

Проще говоря, **мы не даем x_{adv} выйти за пределы ε-окрестности реального изображения x.**

Итерационный шаг PGD



Проективный градиентный спуск на основе диффузии (Diff-PGD)

Метод **Diff-PGD** [11] комбинирует традиционный PGD с преимуществами диффузионной модели для генерации реалистичных атакующих образцов. Этот метод включает 4 расширения:

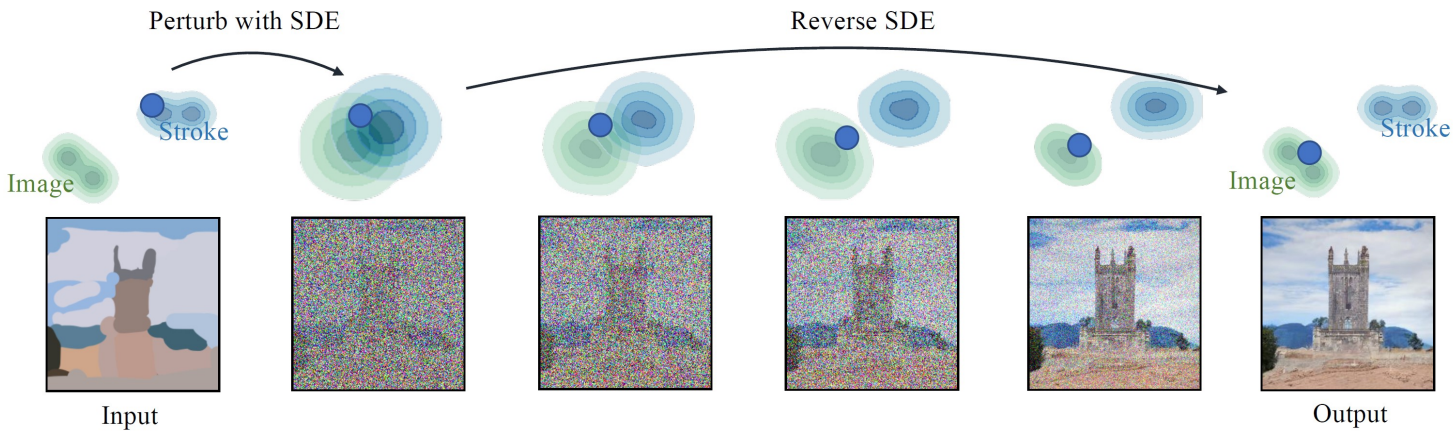
1. **Базовый метод Diff-PGD для глобальной атаки:** Атака по всему изображению, как в случае с обычным PGD.
2. **Региональные атаки (Diff-rPGD):** Внесение изменений только в определенные области изображения с использованием маски.
3. **Кастомизированные атаки, основанные на стиле (Diff-PGD с поддержкой стиля):** Генерация атакующих образцов с учетом стилевых ограничений, таких как стиль заданного образца.
4. **Атаки на физический мир** - сделать изображения устойчивым к плохому освещению, углу обзора и т.д.

Вспомним SDEdit

Метод стохастического дифференциального редактирования (SDEdit) [7] использует диффузионную модель для редактирования изображения путем добавления шума и последующего обратного денойзинга:

$$\text{SDEdit}(x, K) = R_\varphi(\dots R_\varphi(R_\varphi(x_K, K), K-1)\dots, 0),$$

где сначала выполняется K шагов диффузии, а затем K шагов обратного денойзинга. Такой процесс можно использовать, например, для очистки скрытых образцов от шумов.



Базовый Diff-PGD для глобальной атаки

В Diff-PGD вместо оптимизации функции потерь по исходному изображению x проводится оптимизация по очищенному изображению x_0, которое восстанавливается с использованием SDEdit:

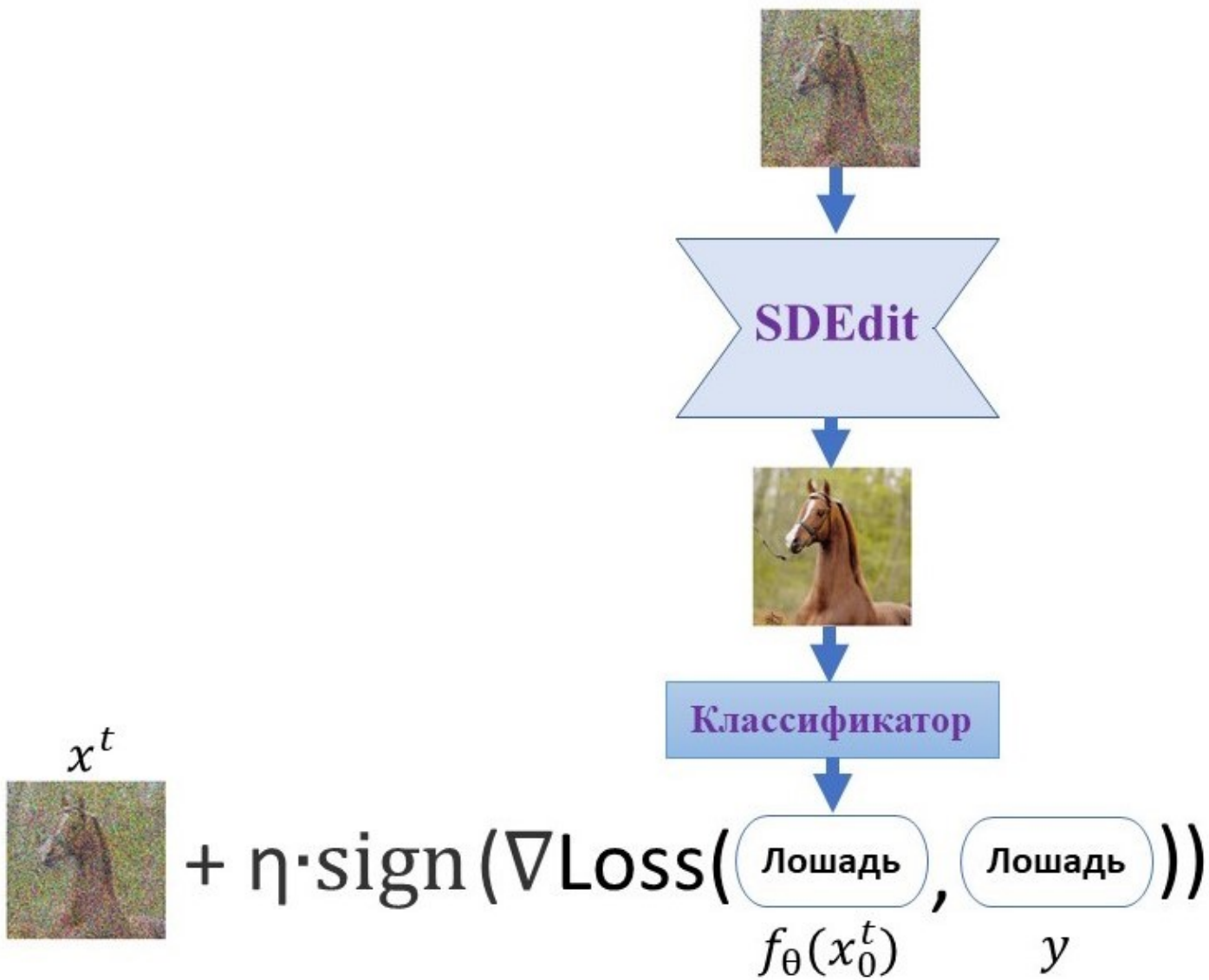
$$x_0^t = \text{SDEdit}(x^t, K),$$

где K — количество шагов обратного денойзинга. Это позволяет сохранить основные свойства естественных изображений, так как на каждом шаге оптимизации используется денойзинг с использованием диффузионной модели, что уменьшает шум и делает образец более реалистичным.
Итерационный шаг Diff-PGD тогда можно записать как:

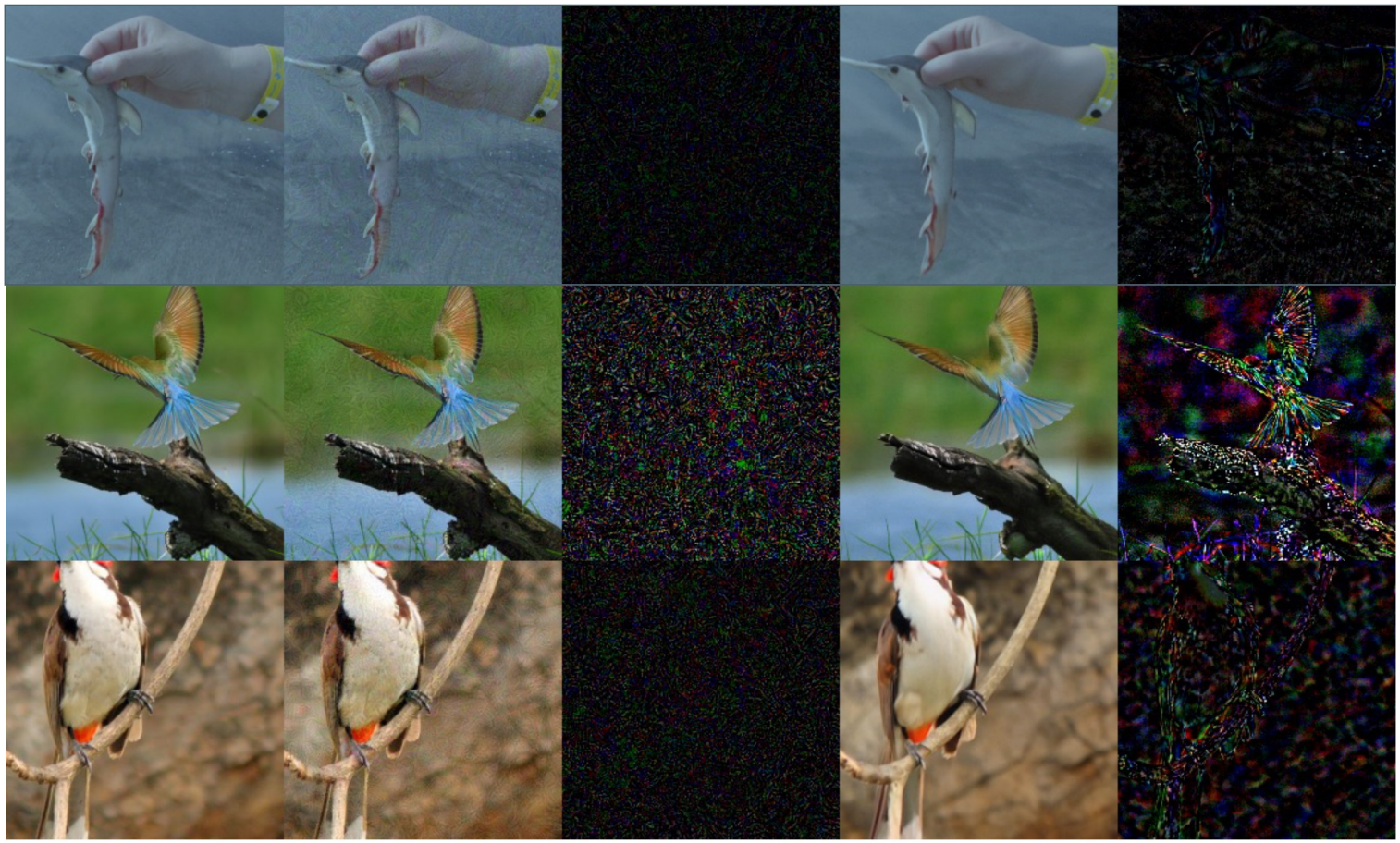
$$x^{t+1} = P_{B_\infty(x,\epsilon)}\left[x^t + \eta \cdot \text{sign}(\nabla_{x^t} l(f_\theta(x_0^t), y))\right],$$

где $x_0^t = \text{SDEdit}(x^t, K)$.

Итерационный шаг Diff-PGD



Примеры редактирования изображения при помощи классического PGD и Diff-PGD



x x_{PGD} δ_{PGD} x_0^n δ_0^n

Примеры редактирования изображения при помощи классического PGD и Diff-PGD

Расширение Diff-PGD для атак в определенных регионах (Diff-rPGD)

Diff-rPGD нацелена на изменение только регионов, определённых маской M, сохраняя остальные части изображения без изменений:

$$(1 - M) \circ x_{\text{adv}} = (1 - M) \circ x,$$

где $\circ$ обозначает поэлементное умножение, а M — бинарная маска. Когда $M = 1$ на всём изображении, Diff-rPGD сводится к стандартному Diff-PGD.
При итеративном процессе SDEdit используется стратегия замены (replacement), чтобы промежуточные образцы в обратном процессе диффузии лучше сочетались с остальной частью изображения:

$$x_i^t = M \circ x_i^t + (1 - M) \circ q(x_i^t | x^t), \tag{1}$$

где $q(x_i^t | x^t)$ — распределение прямого процесса диффузии.

Алгоритм Diff-rPGD

Algorithm 1 Diff-rPGD

Require: Target classifier f_θ, original image x, mask M, denoiser D_ϕ, # of reverse SDEdit steps K, iterations n, stepsize η, clip value ϵ (when $M = 1$ it reduces to Diff-PGD)

$x^0 = x_0^0 = x$, $x_c = x$

for $t = 0, 1, 2, \ldots, n-1$ **do**

  $x_K^t \sim q(x_K^t | x^t)$ ▷ Sample x_K^t from $q(x_K^t | x^t)$ in each PGD iteration

  for $i = K-1, \ldots, 0$ **do**

    $x_i^t = R_\phi(x_{i+1}^t)$ ▷ Apply denoiser R_ϕ to x_{i+1}^t in each SDEdit iteration

    $x_i^t \sim M \circ x_i^t + (1 - M) \circ q(x_i^t | x^t)$ ▷ Sample from masked combination of x_i^t and $q(x_i^t | x^t)$

  end for

  $g = \nabla_{x^t} l[f_\theta(M \circ x_0^t + (1 - M) \circ x_c)]$ ▷ Compute the gradient

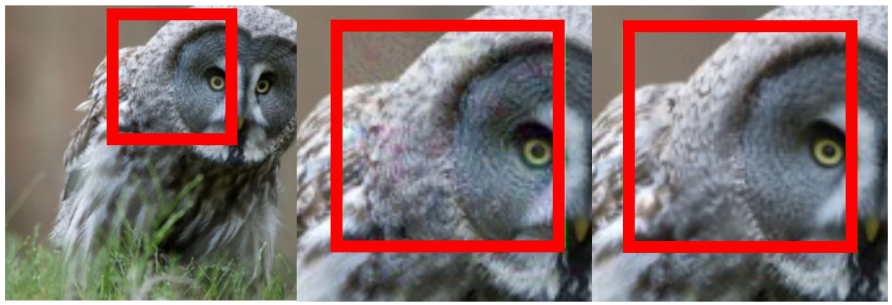
  $x^{t+1} = \Pi_{x,\epsilon}[x^t + \eta\, M \circ \mathrm{sign}(g)]$ ▷ PGD update

end for

$x_0^n = \mathrm{rSDEdit}(x^n)$ ▷ Apply reverse SDEdit to the final x^n

return x^n, x_0^n ▷ Return the final adversarial example and the denoised version

x $(+)\, x_{rPGD}$ $(+)\, x_0^n$



x $(+)\, x_{rPGD}$ $(+)\, x_0^n$

Кастомизированные атаки с определенным стилем

Сначала создается промежуточный образец $\hat{x}_s$, который приближен к стилю x_s с использованием функции потерь стиля $l_s(x, x_s)$, после чего применяется Diff-PGD для получения атакующего образца x_{adv}. Функция потерь стиля определяется как:

$$l_s(x, x_s) = \sum_{h \in H_s} \|G(f_h(x)) - G(f_h(x_s))\|_2^2,$$

где $G(f_h(\cdot))$ — матрица Грама, описывающая промежуточные стилистические признаки изображения, а H_s — слои сети для извлечения признаков стиля.

Алгоритм генерации атакующего образца определённого стиля

Для генерации скрытого образца применяется следующая двухэтапная процедура:

1. Оптимизация стиля для получения промежуточного образца $\hat{x}_s$, близкого по стилю к x_s.
2. Применение Diff-rPGD для создания скрытого образца на основе $\hat{x}_s$.

   Здесь мы используем тот же принцип проекции на шар. Итоговый атакующий образец x_{adv} не отдалится от стилевого $\hat{x}_s$ больше чем на ε:

$$\|x_{adv} - \hat{x}_s\|_\infty \leq \varepsilon$$

Атаки в физическом мире при помощи Diff-PGD

Цель: создать **физический объект** (например, наклейку или патч), который при размещении в реальной сцене «обманывает» целевую модель.

Хотим повысить устойчивость к физическим условиям (изменение освещения или угла обзора) $\Rightarrow$ вставляем **«физический адаптер»**.

«Физический адаптер» для Diff-PGD

Цифровые атаки: ограничиваем норму ℓ_∞
Физический адаптер: применяем T - набор случайных преобразований, которые симулируют изменения окружения.

$$l_{\text{Diff-Phys}}(x) = l_{\text{adv}}\left(\mathbb{E}_{\tau\sim T}\left[\tau(\text{SDEdit}(x, K))\right]\right),$$

где τ - случайные трансформации (изменения фона, сдвиги, освещенность и масштаб). Эти изменения делают патч более устойчивым в физическом мире.

Процесс генерации патча для атаки в физическом мире

1. Определяется начальная область на изображении, где будет располагаться патч;
2. Применяются физические трансформации T (изменение масштаба, яркости и т.д. для симуляции различных условий);
3. Оптимизация патча в цифровой симуляции,
4. Печать патча и размещение в реальной среде для тестирования на целевой модели классификации.

Сравнение генерации патчей для физического мира разными методами

- Зеленая рамка - исходные картинка и объект,
- Синяя рамка - AdvPatch,
- Красная рамка - AdvCam,
- Черная рамка - Diff-Phys.

Подведем итоги

- **Диффузионные модели** - мощный инструмент для генерации изображений: качественно, разнообразно, относительно быстро;
- Можно делать **Text-to-Image** генерацию, определенным образом давая диффузионной модели эмбеддинг текста.
- Диффузионные модели используются для **защиты** систем компьютерного зрения. В рамках этой задачи используется множество методом редактирования изображений.
- Диффузионные модели используются для **атаки** систем компьютерного зрения (например, в методе Diff-PGD), что помогает добиться большей реалистичности атакующих образцов

Список литературы

[1] 2023 Brooks et al. *InstructPix2Pix: Learning to Follow Image Editing Instructions*. URL: http://arxiv.org/abs/2211.09800.

[2] 2023 Gandikota et al. *Concept Sliders: LoRA Adaptors for Precise Control in Diffusion Models*. URL: http://arxiv.org/abs/2311.12092.

[3] 2024 Haas et al. *Discovering Interpretable Directions in the Semantic Latent Space of Diffusion Models*. URL: http://arxiv.org/abs/2303.11073.

[4] 2022 Hertz et al. *Prompt-to-Prompt Image Editing with Cross Attention Control*. URL: http://arxiv.org/abs/2208.01626.

[5] 2020 Ho et al. *Denoising Diffusion Probabilistic Models*. URL: https://arxiv.org/abs/2006.11239.

[6] 2022 J. Ho & T. Salimans. *Classifier-free Diffusion Guidance*. URL: https://arxiv.org/abs/2207.12598.

Список литературы

[7] 2022 Meng et al. *SDEdit: guided image synthesis and editing with Stochastic Differential Equations*. URL: https://arxiv.org/pdf/2108.01073.

[8] 2023 Mokady et al. *Null-text Inversion for Editing Real Images using Guided Diffusion Models*. URL: https://arxiv.org/pdf/2211.09794.

[9] 2021 Song et al. *Denoising Diffusion Implicit Models*. URL: https://arxiv.org/abs/2010.02502.

[10] 2021 Swen Gowal et al. *Improving Robustness using Generated Data*. URL: https://arxiv.org/pdf/2110.09468.

[11] 2024 Xue et al. *Diffusion-Based Adversarial Sample Generation for Improved Stealthiness and Controllability*. URL: https://arxiv.org/pdf/2305.16494.