

Deep Learning School

бесплатно.



онлайн.



фундаментально.



Deep Learning School

онлайн-школа по искусственному интеллекту
в России





Deep Learning School





Коновалова Нина Петровна

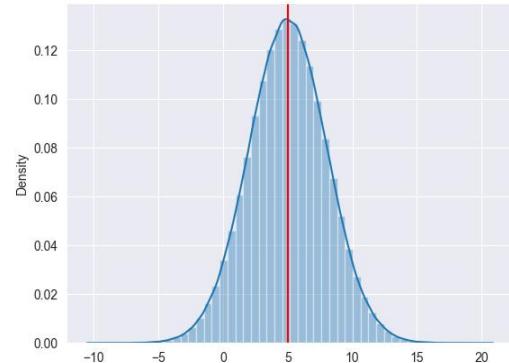
AE и VAE Deep Latent models

Вспомнить всё - матожидание и дисперсия

Рассмотрим случайную величину x с плотностью $p(x)$

Математическое ожидание или «центр тяжести»
характерное среднее значение)

$$\mathbb{E}x = \int xp(x)dx$$



Свойства матожидания:

- **линейность** $\mathbb{E}[aX + bY] = a\mathbb{E}[X] + b\mathbb{E}[Y]$

- **матожидание константы** $\mathbb{E}[c] = c$

Вспомнить всё - дисперсия

Рассмотрим случайную величину x с плотностью $p(x)$

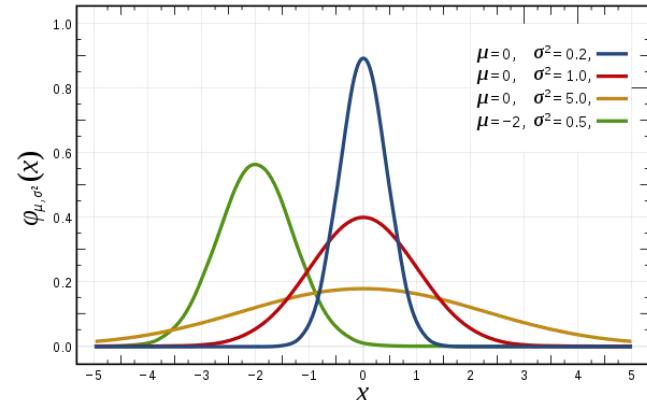
Средний квадрат отклонения от матожидания - **дисперсия**

$$\mathbb{D}x = \mathbb{E}(x - \mathbb{E}x)^2 = \int (x - \mathbb{E}x)^2 p(x) dx$$

Свойства дисперсии:

$$\mathbb{D}[X] \geq 0$$

$$\mathbb{D}[X + Y] = \mathbb{D}[X] + \mathbb{D}[Y] + 2\text{cov}(X, Y)$$



Вспомнить всё - условная вероятность

Пусть x и y - случайные величины с плотностями $p(x)$ и $p(y)$

Совместное распределение

$$p(x, y) \neq p(x)p(y)$$

x, y - независимые

$$p(x, y) = p(x)p(y)$$

Условная плотность вероятности

$$\underbrace{p(x|y)}_{conditional} = \frac{\overbrace{p(x, y)}^{joint}}{\underbrace{p(y)}_{marginal}}$$

Вспомнить всё - условная вероятность

Правило произведения

$$p(x, y, z) = p(x|y, z)p(y|z)p(z)$$

Правило суммирования

$$p(y) = \int p(x, y)dx = \int p(y|x)p(x)dx$$

Немного Байеса в нашу жизнь

Теорема Байеса помогает решать обратные вероятностные задачи

Условная вероятность - хотим получить вероятность x , при условии y

$$p(x|y) = \frac{p(x, y)}{p(y)} = \frac{p(y|x)p(x)}{p(y)}$$

Теорема Байеса - на основе наблюдений x , хотим спрогнозировать y , то есть обратить условную вероятность

$$\underbrace{p(y|x)}_{posterior} = \frac{\overbrace{p(x|y)}^{likelihood} \times \overbrace{p(y)}^{prior}}{\underbrace{p(x)}_{evidence}}$$



Функция правдоподобия и ее максимизация

Задача. Пусть дана выборка из независимых одинаково распределённых величин, известных с точностью до параметра

$$X = \{x_i\}_{i=1}^n \sim p(x|\theta)$$

Необходимо оценить параметр, при котором вероятность получить данный набор величин - максимальна.

Функция правдоподобия:

$$\mathcal{L}(\theta) = \prod_{i=1}^n p(x_i|\theta)$$

$$\theta_{ML} = \arg \max \prod_{i=1}^n p(x_i|\theta)$$

Функция правдоподобия и ее максимизация

Работать с произведениями не очень удобно, поэтому лучше перейти от функции правдоподобия к логарифму функции правдоподобия

Логарифм функции правдоподобия:

$$\log \mathcal{L}(\theta) = \log \prod_{i=1}^n p(x_i|\theta) = \sum_{i=1}^n \log p(x_i|\theta)$$

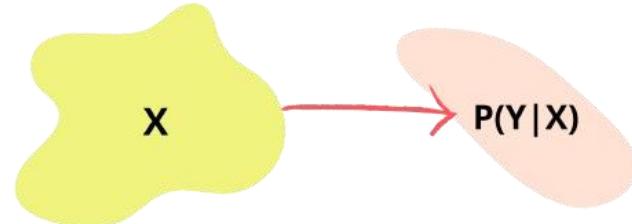
$$\theta_{ML} = \arg \max \sum_{i=1}^n \log p(x_i|\theta)$$

Дискриминативные модели

То, что мы раньше изучали - дискриминативные модели, то есть модели, которые выучивают распределение $P(y|x)$, а если быть точнее - $P(y,W|x)$.

Какими свойствами обладают дискриминативные модели:

- Чаще у имеет более простую структуру
- Мы не генерируем новые объекты



Примеры задач

Например это могут быть следующие задачи:

Классификация



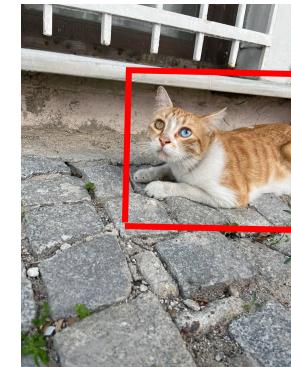
↓
КОТ

Сегментация



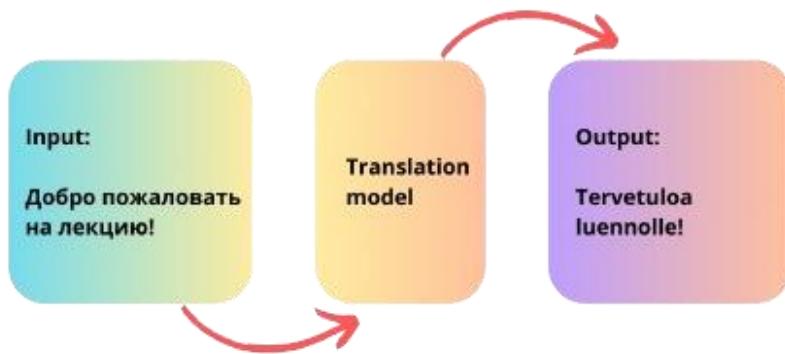
<https://segment-anything.com/demo>

Детекция

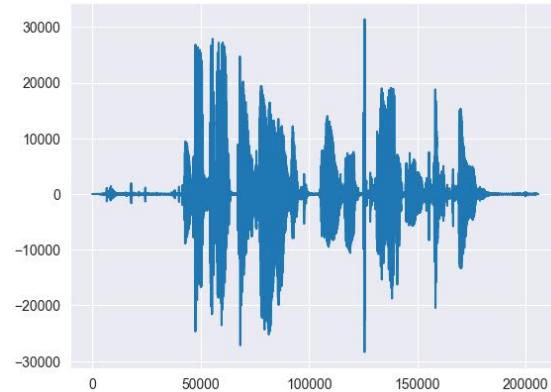


Не обязательно картинки

Задача машинного перевода



Классификация речи



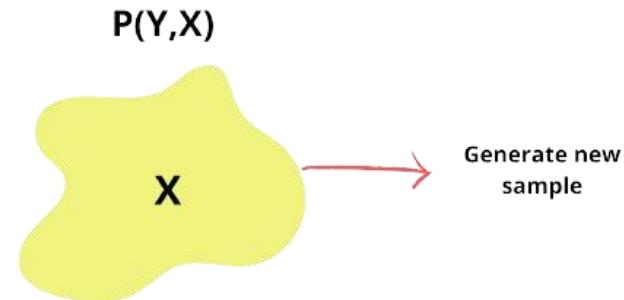
Пол: ж
Возраст: 25 - 30

Генеративные модели

В данной лекции мы с вами познакомимся с генеративными моделями - $P(y, x, W)$.

Какими свойствами обладают генеративные модели:

- генерируем новые объекты



Примеры задач

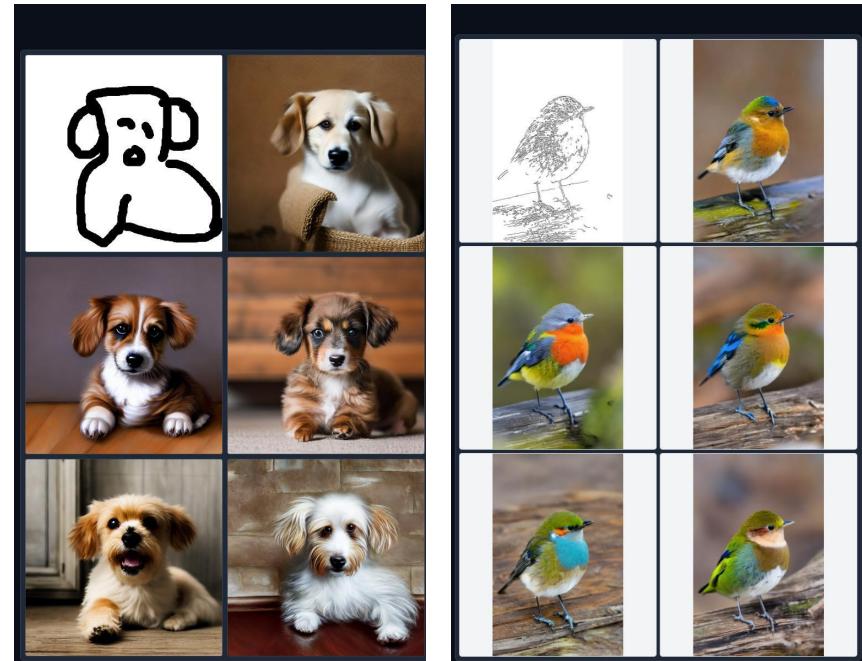


<https://github.com/Stability-AI/generative-models>

Генерация:

- Диффузионные модели
- VAE
- GANs

[https://github.com/llyasviel/ControlNet \[1\]](https://github.com/llyasviel/ControlNet [1])



SORA OpenAI



Не обязательно картинки

Генерация музыки



A funk song with rhythmical guitars using wah pedals, a horn section, a strong bass line and husky vocals.

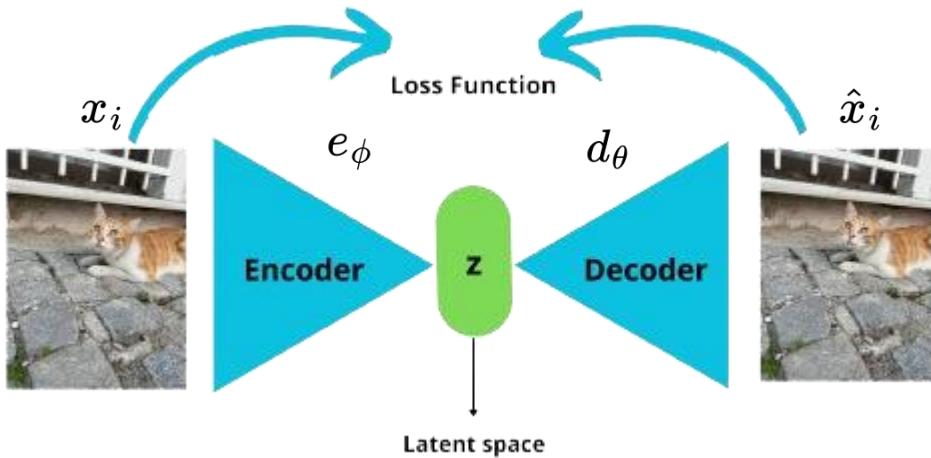
<https://google-research.github.io/noise2music/>

Alpha GO



https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2Fpoint-nine-news%2Fwhat-does-alphago-vs-8dadec65aaaf&psig=AOvVaw2CO_5hx3ytbaMDxoBL1Dt6&ust=1713376266228000&source=images&cd=vfe&opi=89978449&ved=0CBIBQRxqFwoTCNDCoZ0mx4UDFQAAAAAdAAAAABAE

Автокодировщики (autoencoders)

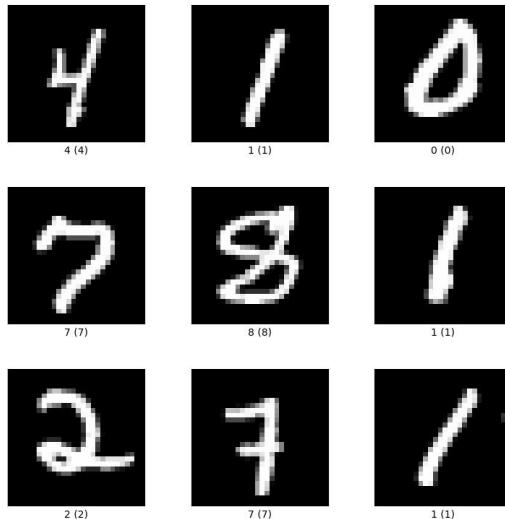


$$\min_{\phi, \theta} \frac{1}{N} \sum_{i=1}^N loss(d_\theta(e_\phi(x_i)), x_i)$$

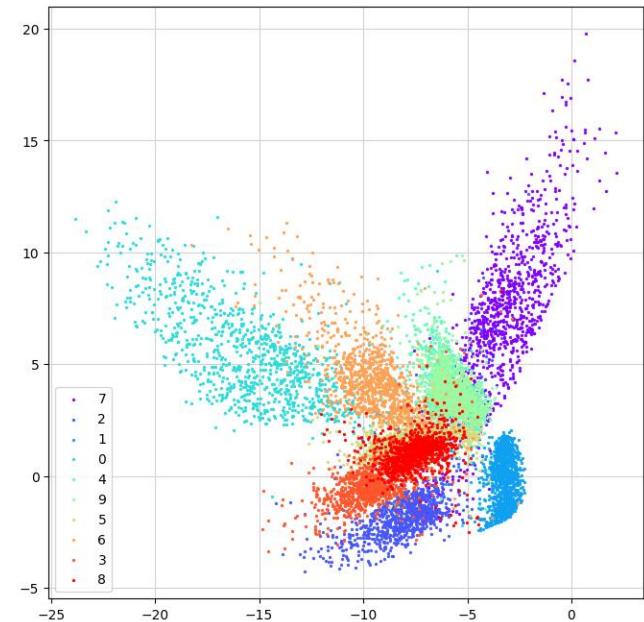
- Состоит из двух основных частей
 - **энкодер** e_ϕ - кодирует в латентное пространство
 - **декодер** d_θ - декодирует обратно
- На латентное пространство не накладываются дополнительных условий
- Не можем сэмплировать из латентного пространства
- Важно выбрать хорошую функцию потерь, чаще всего используется MSE

Уменьшение размерности

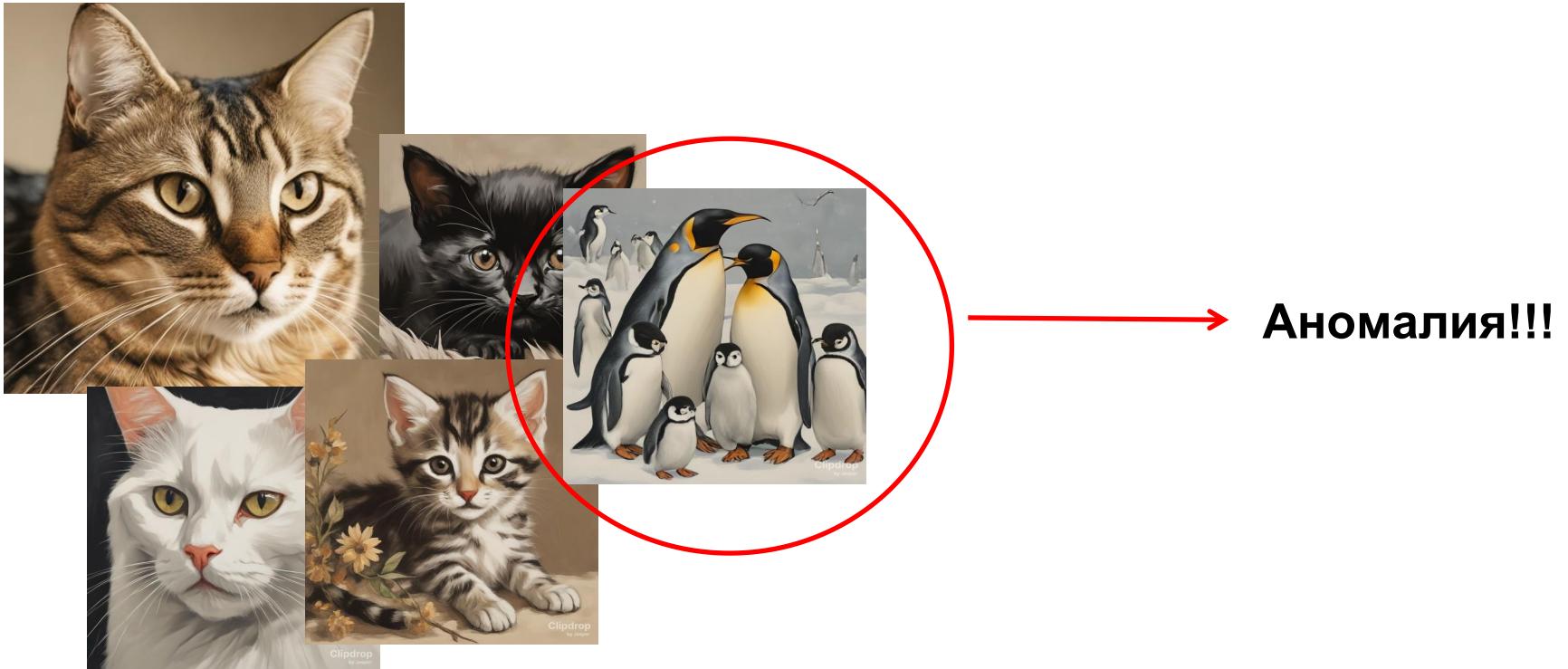
Датасет MNIST - 28 x 28



Латентное пространство MNIST - 1 x 2

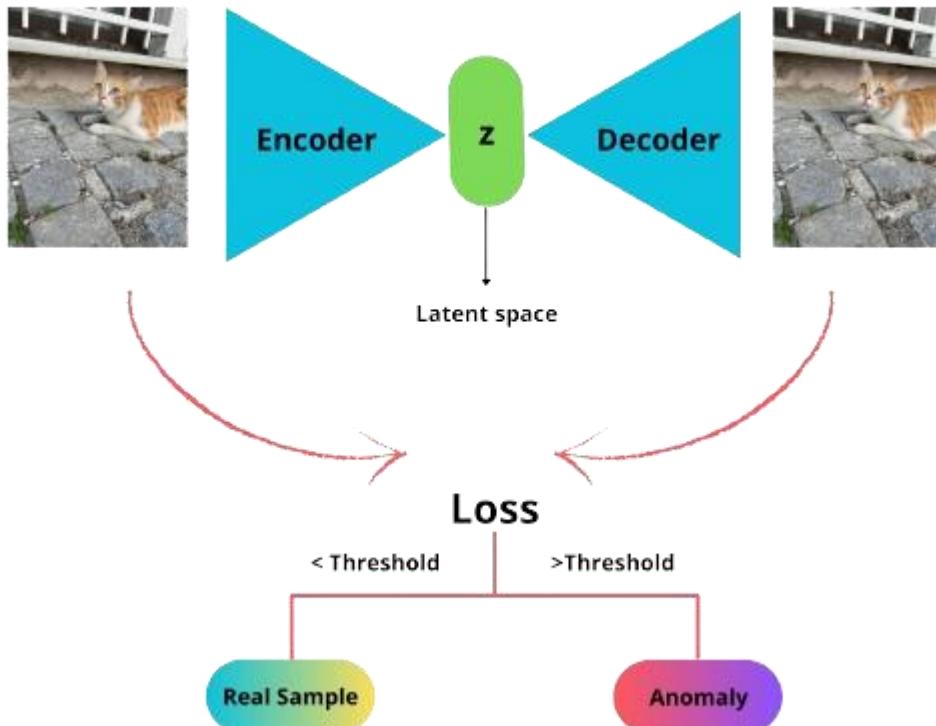


Детекция аномалий

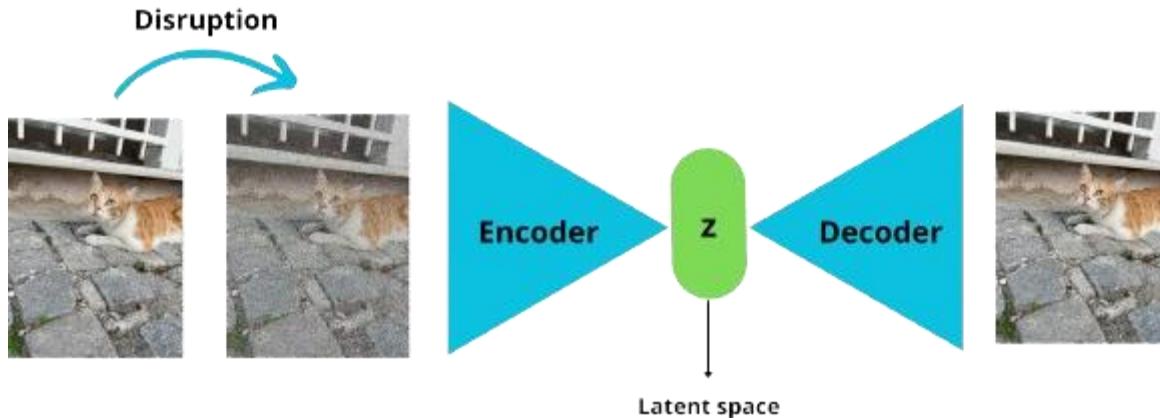


Изображения сгенерированы с помощью [SDXL Turbo](#)

Детекция аномалий



Denoising autoencoder



- Энкодер получает на вход немного зашумленное изображение
- Декодер восстанавливает исходное изображение

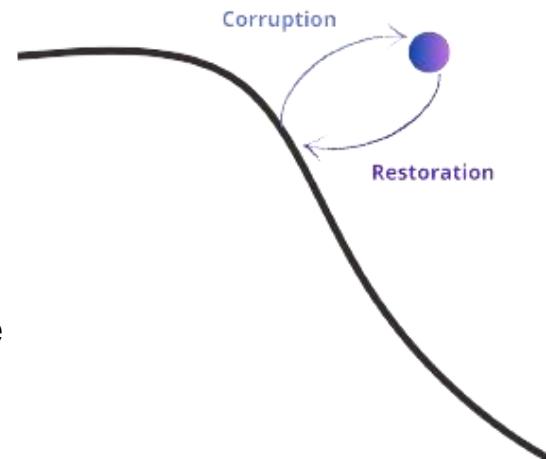


Image Inpainting



[2] Image restoration using convolutional auto-encoders with symmetric skip connections

Другие улучшения

Image super-resolution

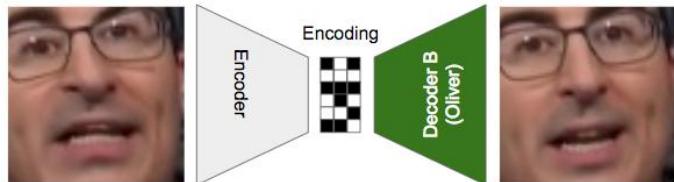
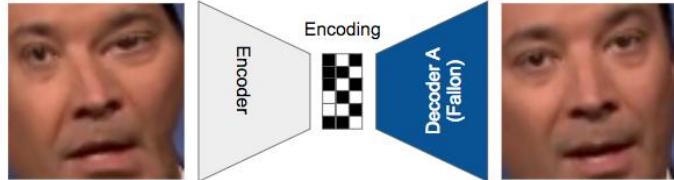


Image deblurring



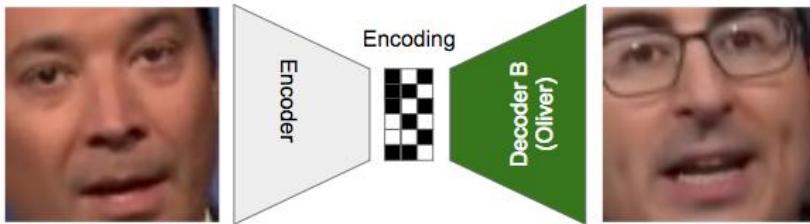
[2] Image restoration using convolutional auto-encoders with symmetric skip connections

DeepFakes



Обучение.

В данной постановке мы обучаем один общий энкодер и два различных декодера



Валидация.

Для валидации мы с помощью энкодера кодируем одно лицо, а декодируем другое.

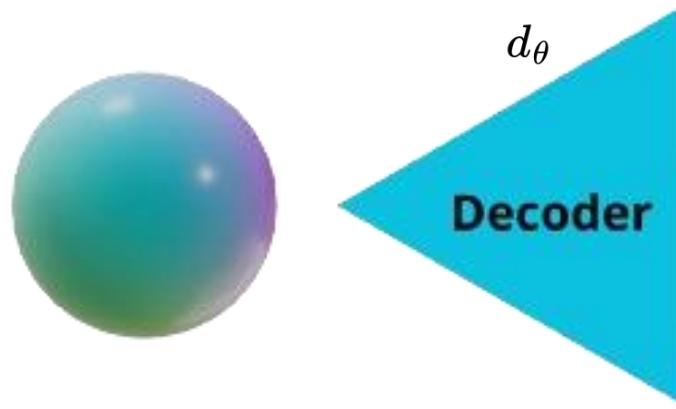
DeepFakes



[3] Gaurav Oberoi blog

Generative Latent Optimization (GLO)

$$\mathcal{Z} = \mathcal{B}(r, d, p) = \{z \in \mathbb{R}^d : \|z\|_p \leq r\}$$

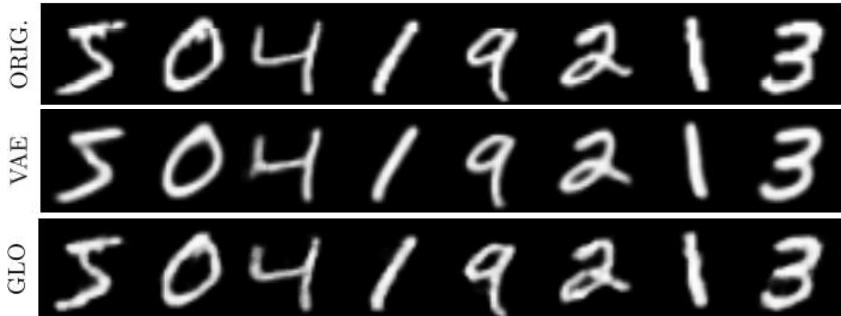


Важные особенности:

- нет энкодера
- латентное пространство
- d -мерный шар
определенного радиуса
- латентные векторы тоже
оптимизируются

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N \left[\min_{z_i \in \mathcal{Z}} \text{loss}(d_\theta(z_i), x_i) \right]$$

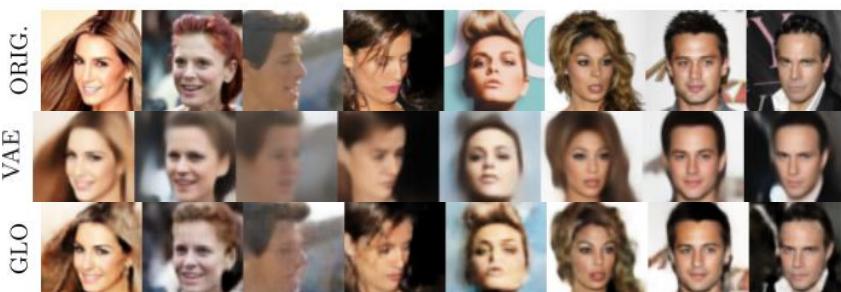
GLO - реконструкция



(a) MNIST



(b) SVHN



(c) CelebA-64



(d) CelebA-128

GLO - интерполяция

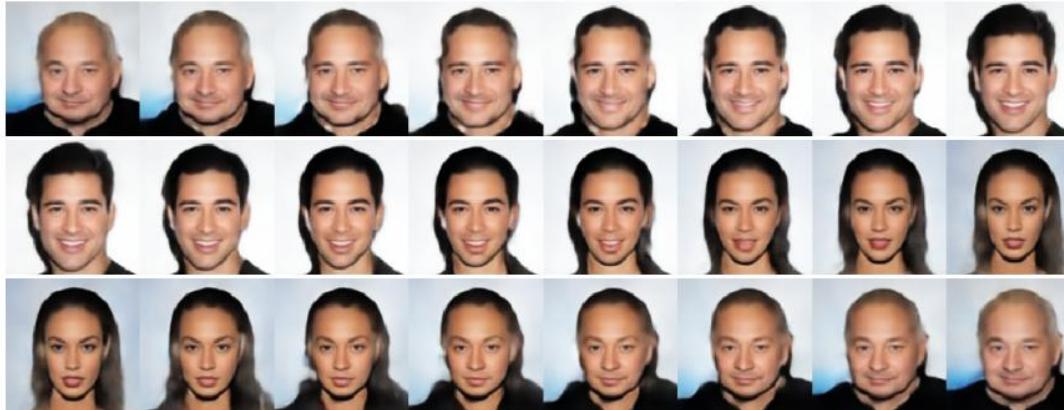


Figure 2. Illustration of interpolations obtained with our model on the CelebA dataset. We construct a path between 3 images to verify that paths do not collapse to an “average” representation in the middle of the interpolation.

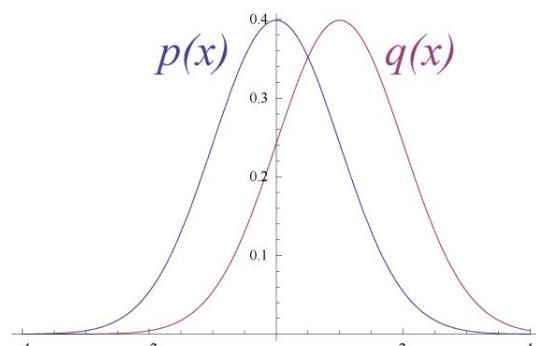
GLO - генерация



KL - дивергенция или как нам сравнить распределения

Для непрерывного распределения

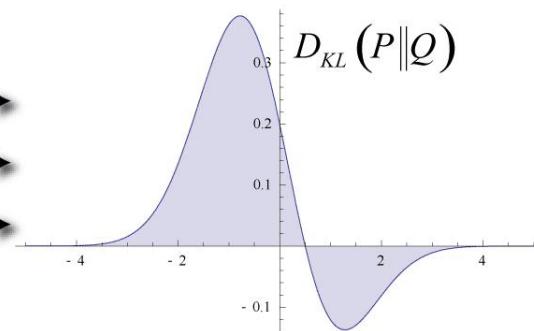
$$D_{KL}(p\|q) = \int p(x) \log \frac{p(x)}{q(x)} dx$$



Original Gaussian PDF's

Для дискретного распределения

$$D_{KL}(p\|q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$



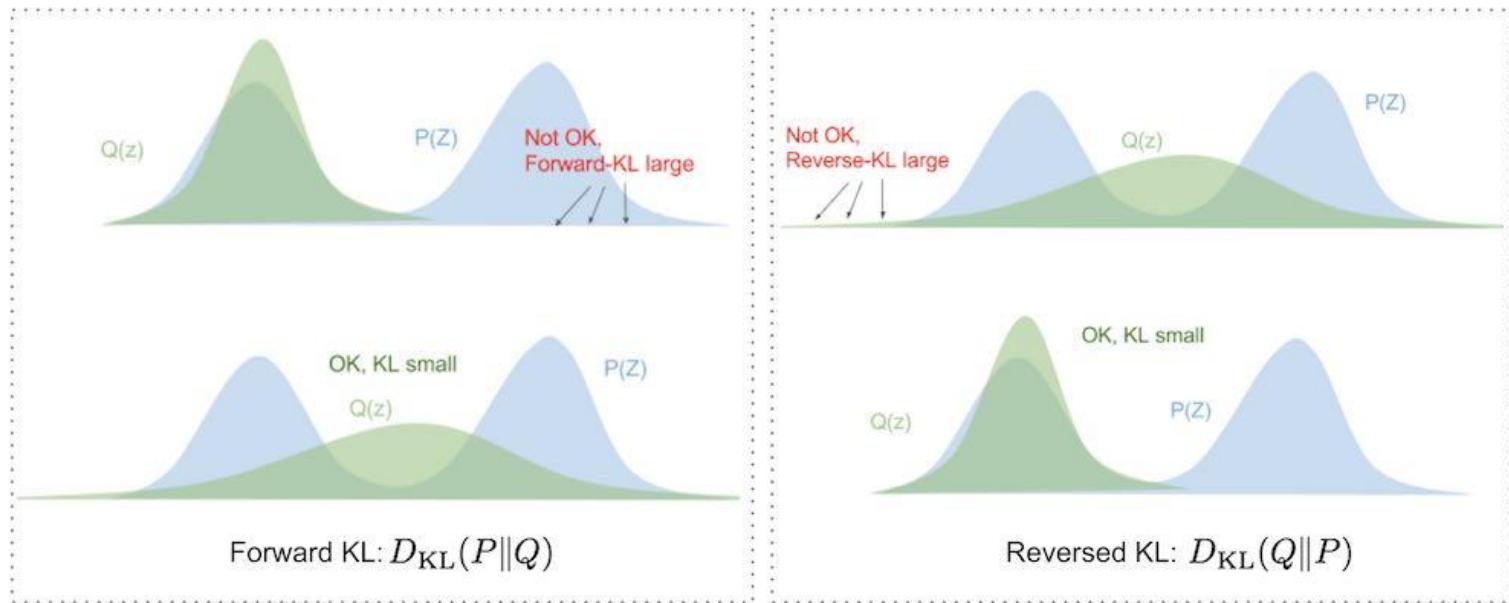
KL Area to be Integrated

KL - дивергенция или как нам сравнить распределения

Свойства KL-дивергенции

- Неотрицательность: $D_{KL}(p\|q) \geq 0$
- Нессимметричность: $D_{KL}(p\|q) \neq D_{KL}(q\|p)$

KL - дивергенция или как нам сравнивать распределения



<https://blog.evjang.com/2016/08/variational-bayes.html>

$$D_{KL}(p\|q) \neq D_{KL}(q\|p)$$

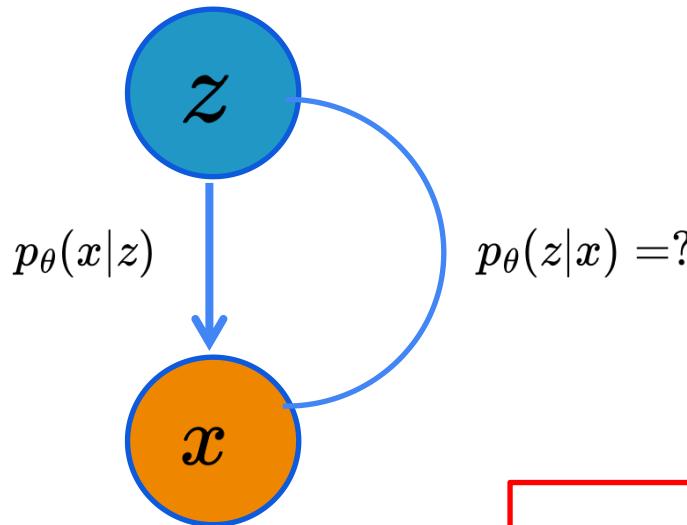
Вариационные автокодировщики (VAE)

В идеале хотим выучить

$$\log p(x) = \frac{1}{N} \sum_{i=1}^N \left(\int \log p_\theta(x_i|z)p(z)dz \right) \rightarrow \max_\theta$$

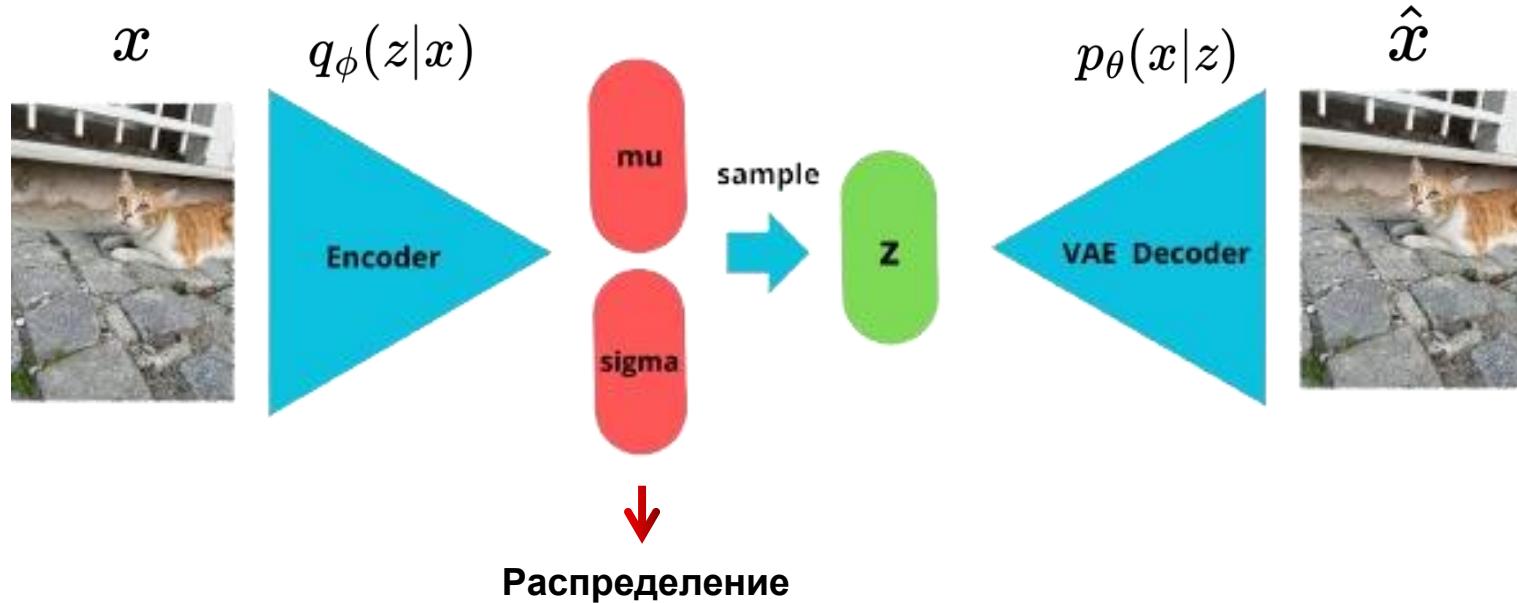
генератор

latents prior



Идея: учим $q(z|x, \phi)$ вместо $p_\theta(z|x)$
 $q(z_i|x_i, \phi) \approx p(z_i|x_i, \theta)$

Вариационные автокодировщики (VAE)



$$q(z_i|x_i, \phi) = \prod_{j=1}^d \mathcal{N}(z_{i,j}|\mu_j(x_i), \sigma_j^2(x_i))$$

Обучение VAE

$$\begin{aligned}\log p(x|\theta) &= \int q(z|x, \phi) \log p(x|\theta) dz = \int q(z|x, \phi) \log \frac{p(x, z|\theta)}{p(z)} dz = \\ &\quad \int q(z|x, \phi) \log \frac{p(x, z|\theta)q(z|x, \phi)}{p(z)q(z|x, \phi)} dz = \\ &= \underbrace{\int q(z|x, \phi) \log \frac{q(z|x, \phi)}{p(z)} dz}_{D_{KL}(q(z|x, \phi) || p(z)) \geq 0} + \underbrace{\int q(z|x, \phi) \log \frac{p(x, z|\theta)}{q(z|x, \phi)} dz}_{ELBO} \\ \mathcal{L}_{ELBO} &= \int q(z) \log \frac{p(x, z)}{q(z)} dz \leq \log p(x)\end{aligned}$$

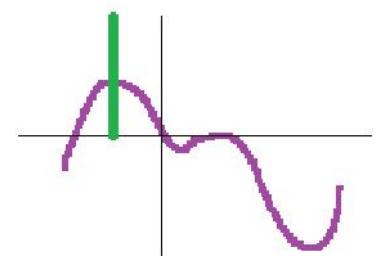
ELBO

$$\mathcal{L}_{ELBO}(\phi, \theta) = \int q(z|x, \phi) \log \frac{p(x|z, \theta)p(z|\theta)}{q(z|x, \phi)} dz \rightarrow \max_{\theta, \phi}$$

$$\mathcal{L}_{ELBO}(\phi, \theta) = \int q(z|x, \phi) \log p(x|z, \theta) dz + \int q(z|x, \phi) \log \frac{p(z|\theta)}{q(z|x, \phi)} dz =$$

$$= \sum_{i=1}^n \int q(z_i|x_i, \phi) \log p(x_i|z_i, \theta) dz_i - \sum_{i=1}^n D_{KL}(q(z_i|x_i, \phi) \| p(z_i))$$

Reconstruction **Regularization**



$\log p(x_i|z_i, \theta)$

KL-дивергенция

Для двух нормальных распределений, одно из которых с нулевым средним и единичной дисперсией KL-дивергенцию можно расписать следующим образом:

$$q(z|x, \phi) \sim \mathcal{N}(z|\mu, \sigma^2 I)$$

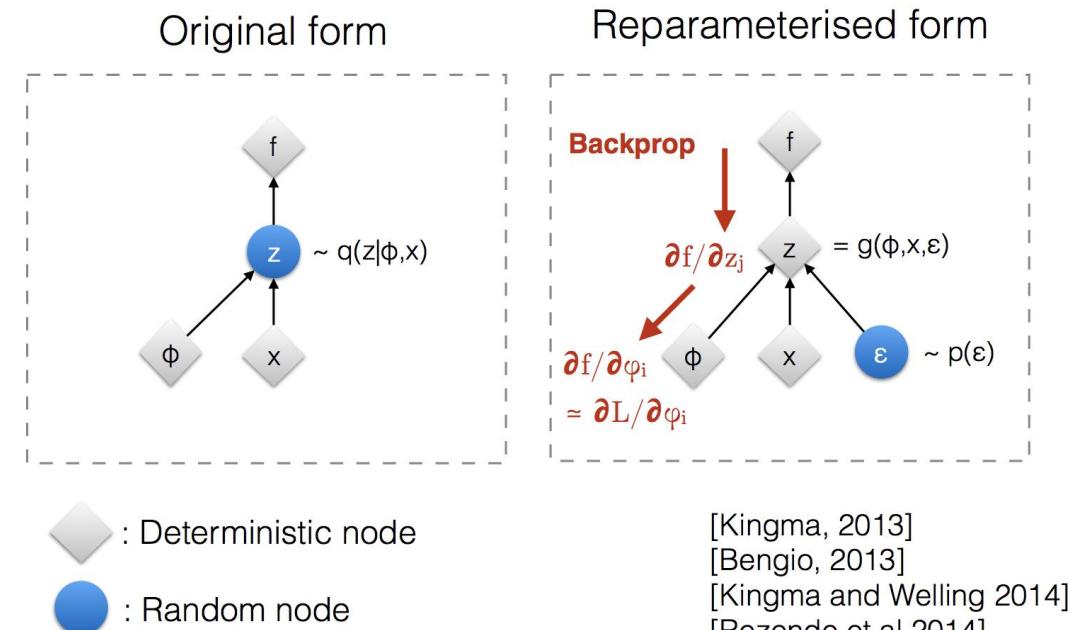
$$p(z) \sim \mathcal{N}(z|0, I)$$

$$D_{KL}(q(z|x, \phi) | p(z)) = \frac{1}{2} \sum_i (\mu_i^2 + \sigma_i^2 - 1 - \log \sigma_i^2)$$

Сэмплирование

Проблема: необходимо сэмплировать из распределения, а это стохастический процесс и мы не можем пропустить градиенту

Решение:
репараметризационный трюк
- представим нашу случайную величину как детерминистическую, зависящую от случайной величины



[Kingma, 2013]
[Bengio, 2013]
[Kingma and Welling 2014]
[Rezende et al 2014]

Результаты

Реконструкция



Сэмплирование



Латентное пространство VAE



6	6	6	6	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	4	4	4	2	2	2	2	0	0	0	0	0	0	0	0	0	0	0	0
9	2	2	2	2	2	2	2	3	5	5	5	6	0	0	0	0	0	0	0
9	9	2	2	2	2	2	2	3	3	5	5	5	5	0	0	0	0	0	0
9	9	9	2	2	2	2	2	3	3	3	3	3	5	5	5	5	5	5	3
9	9	9	4	2	2	2	2	3	3	3	3	3	5	5	5	5	5	5	3
9	9	9	9	9	2	2	2	3	3	3	3	3	5	5	5	5	5	5	3
9	9	9	9	9	9	3	3	3	3	3	3	3	3	3	3	3	3	3	7
7	9	9	9	9	9	9	3	3	3	3	3	3	3	3	3	3	3	3	7
7	9	9	9	9	9	9	8	8	8	8	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	8	8	8	8	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	8	8	8	8	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	8	8	8	8	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	9	8	8	8	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	9	9	8	8	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	9	9	9	8	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	9	9	9	9	8	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	9	9	9	9	9	8	8	8	8	8	8	8	7
7	9	9	9	9	9	9	9	9	9	9	9	9	8	8	8	8	8	8	7
7	9	9	9	9	9	9	9	9	9	9	9	9	9	8	8	8	8	8	7
7	9	9	9	9	9	9	9	9	9	9	9	9	9	9	8	8	8	8	7
7	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	8	8	8	7
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7

GitHub с различным автокодировщиками

Beta-VAE (Code , Config)	Link		
Disentangled Beta-VAE (Code , Config)	Link		

<https://github.com/AntixK/PyTorch-VAE?tab=readme-ov-file>

Image editing using VAE



<https://rtoledo.me/post/2021-05-31-edit-face-attributes-using-vae/edit-face-attributes-using-vae/>

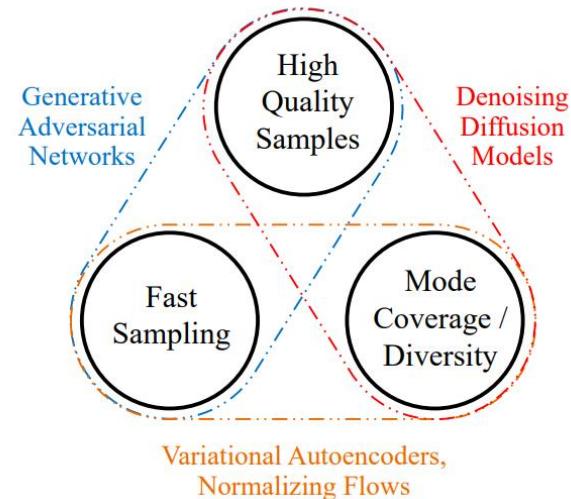
Проблемы VAE

Pros:

- Вариативность генерации
- Высокая скорость

Cons:

- «Замыленные» изображение, низкое качество



[6] DENOISING DIFFUSION GANS

Дополнительные главы - VQ-VAE

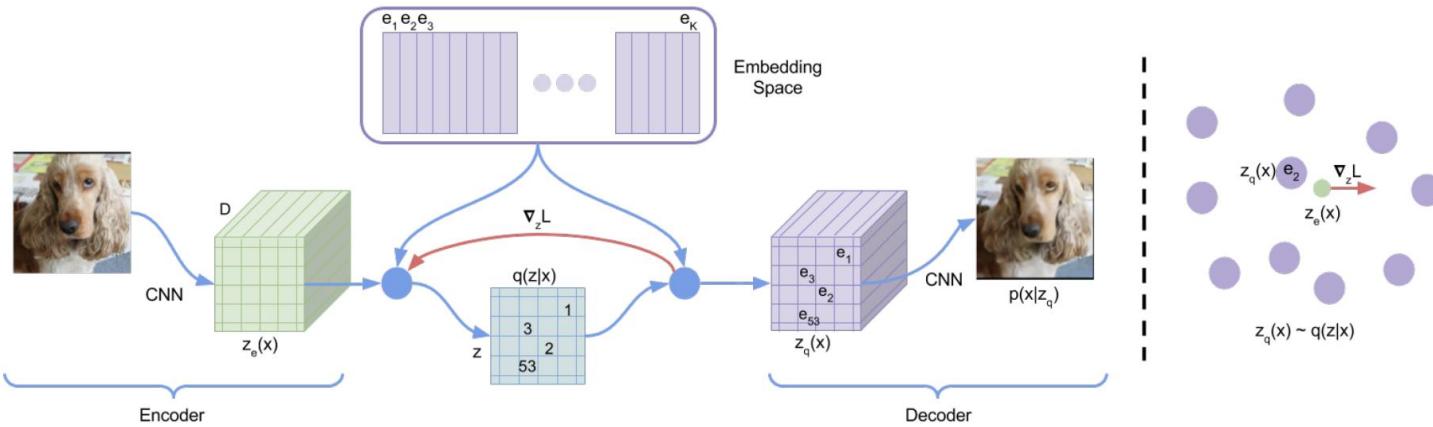
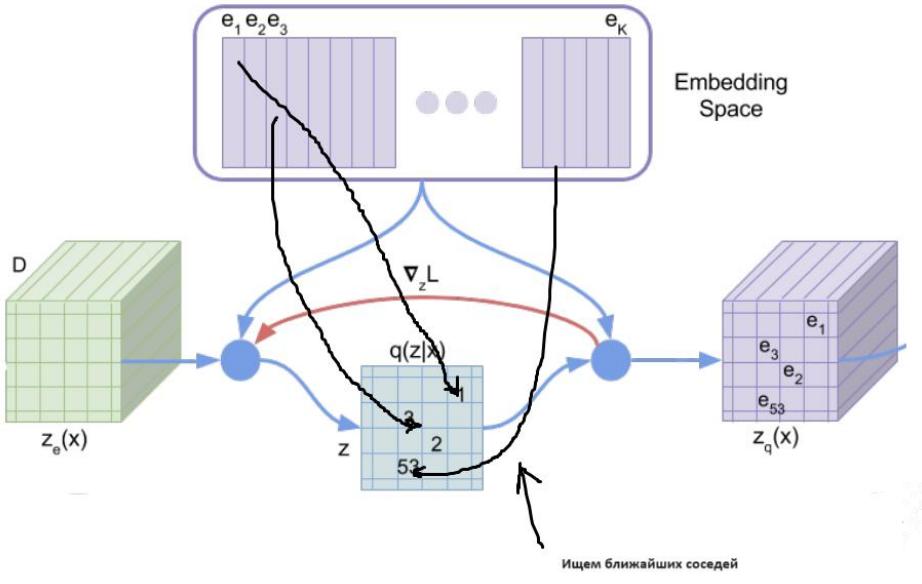


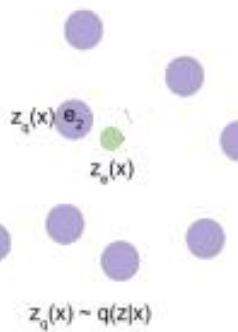
Figure 1: Left: A figure describing the VQ-VAE. Right: Visualisation of the embedding space. The output of the encoder $z(x)$ is mapped to the nearest point e_2 . The gradient $\nabla_z L$ (in red) will push the encoder to change its output, which could alter the configuration in the next forward pass.

VQ-VAE - кодирование



Для каждого закодированного вектора ищем ближайшего соседа из нашего обучаемого словаря

$$z_q(x) = e_k, k = \arg \min_j \|z_e(x) - e_j\|_2$$



VQ-VAE - обучение

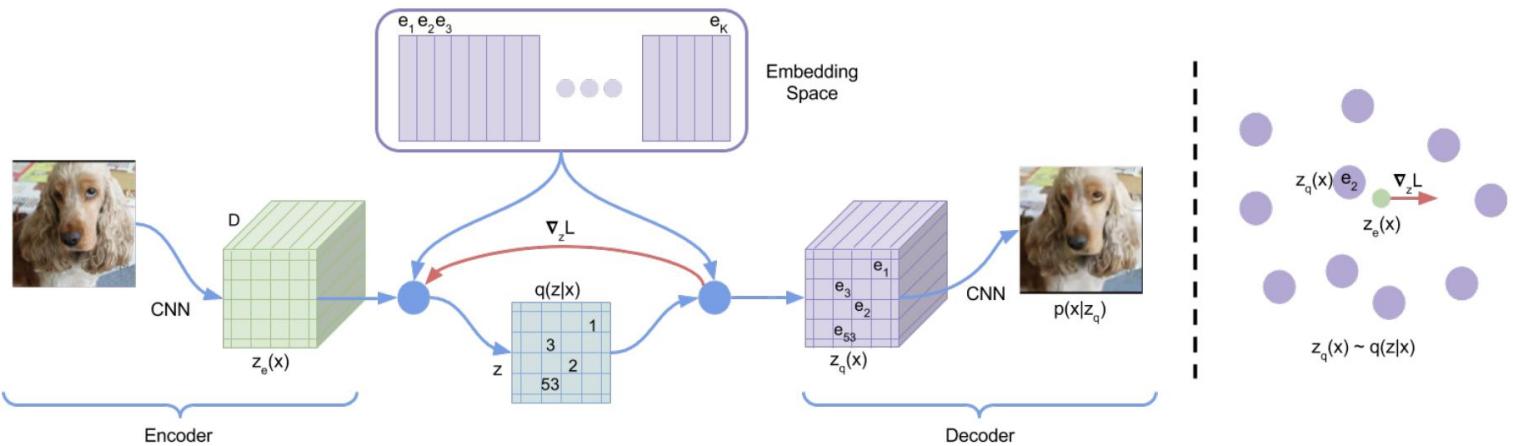
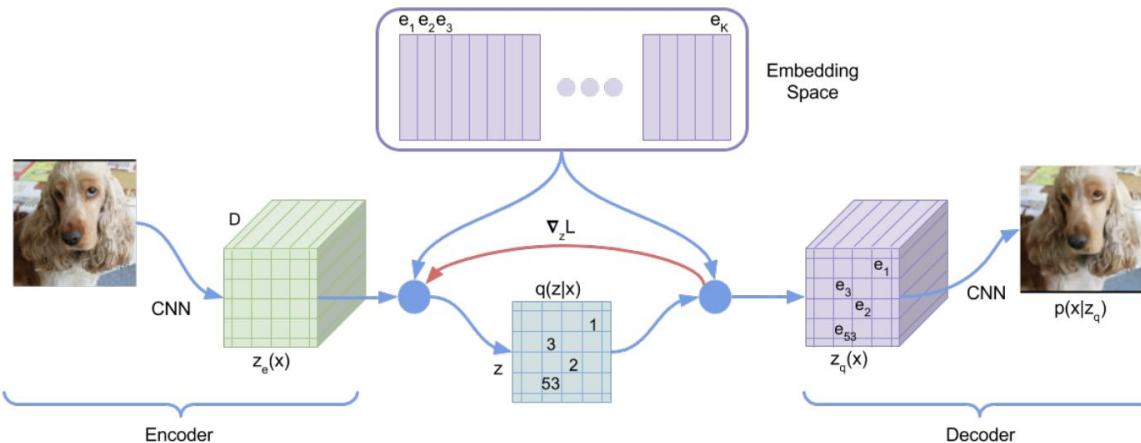


Figure 1: Left: A figure describing the VQ-VAE. Right: Visualisation of the embedding space. The output of the encoder $z(x)$ is mapped to the nearest point e_2 . The gradient $\nabla_z L$ (in red) will push the encoder to change its output, which could alter the configuration in the next forward pass.

$$\mathcal{L} = \boxed{\log p(x|z_q(x))} + \boxed{\|sg[z_e(x)] - e\|_2^2} + \boxed{\beta \|z_e(x) - sg[e]\|_2^2}$$

Реконструкция Обучаем элементы словаря Обучаем энкодер

VQ-VAE - обучение



Реконструкция: качество восстановленного изображения.

$$\log p(x|z_q(x))$$

Для пропуска градиента (backpropagation) используется метод **Straight-through gradient estimation** (градиент по вектору слова просто копируется в градиент по вектору энкодера)

VQ-VAE - обучение

Оптимизация векторов словаря

$$\|sg[z_e(x)] - e\|_2^2$$

Оптимизация энкодера

$$\beta \|z_e(x) - sg[e]\|_2^2$$

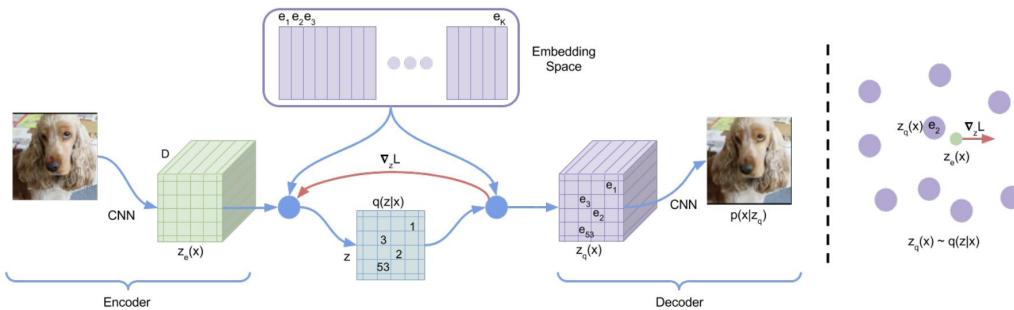
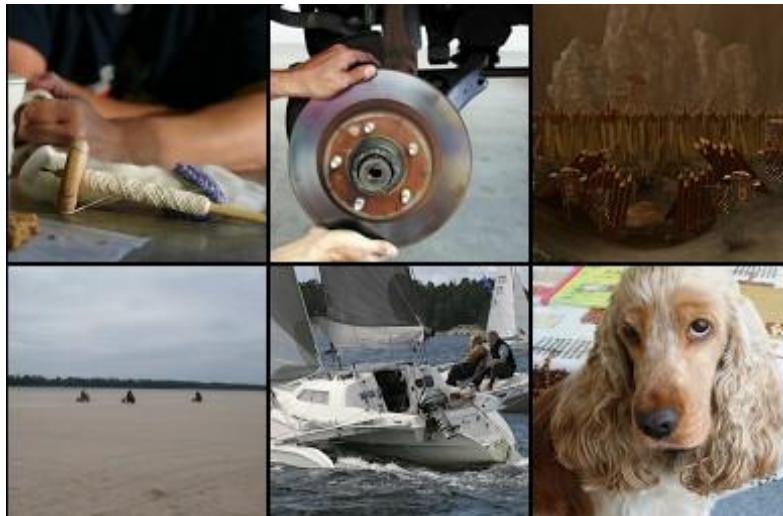


Figure 1: Left: A figure describing the VQ-VAE. Right: Visualisation of the embedding space. The output of the encoder $z(x)$ is mapped to the nearest point e_2 . The gradient $\nabla_z L$ (in red) will push the encoder to change its output, which could alter the configuration in the next forward pass.

VQ-VAE - результаты



Реальные изображения



Реконструкция VQVAE

Pixel-CNN prior

Учим свёрточную нейросеть, которая по предыдущим кодам (пикселям) предсказывает распределение вероятностей следующего.

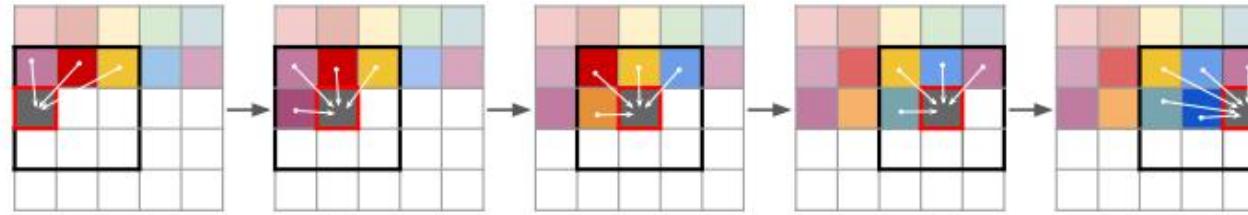
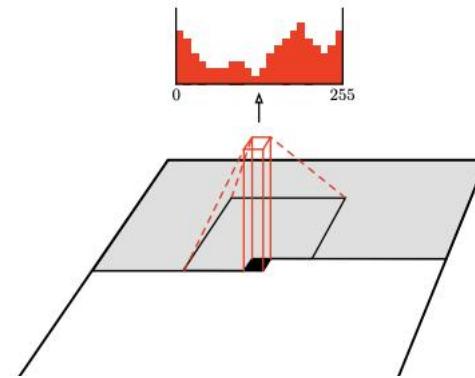
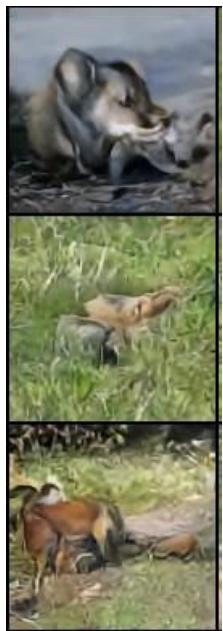


Figure 3. Sliding attention window.

- [8] Taming transformers for high-resolution image synthesis
- [9] Pixel recurrent neural networks

VQ-VAE - генерация





Коновалова Нина

Спасибо
за внимание!

@AfeliaN
konovalova.np@phystech.edu

Bibliography

- [1] Zhang L., Rao A., Agrawala M. Adding conditional control to text-to-image diffusion models //Proceedings of the IEEE/CVF International Conference on Computer Vision. – 2023. – C. 3836-3847.
- [2] Mao X. J., Shen C., Yang Y. B. Image restoration using convolutional auto-encoders with symmetric skip connections //arXiv preprint arXiv:1606.08921. – 2016.
- [3] <https://goberoi.com/exploring-deepfakes-20c9947c22d9>
- [4] Bojanowski P. et al. Optimizing the latent space of generative networks //arXiv preprint arXiv:1707.05776. – 2017.
- [5] <https://blog.evjang.com/2016/08/variational-bayes.html>
- [6] Xiao Z., Kreis K., Vahdat A. Tackling the generative learning trilemma with denoising diffusion gans //arXiv preprint arXiv:2112.07804. – 2021.
- [7] Van Den Oord A. et al. Neural discrete representation learning //Advances in neural information processing systems. – 2017. – T. 30.

Bibliography

- [8] Esser P., Rombach R., Ommer B. Taming transformers for high-resolution image synthesis //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. – 2021. – C. 12873-12883.
- [9] Van Den Oord A., Kalchbrenner N., Kavukcuoglu K. Pixel recurrent neural networks //International conference on machine learning. – PMLR, 2016. – C. 1747-1756.