Лекция 9

Representation learning, AE, VAE

Introduction

В машинном обучении, необходимо:

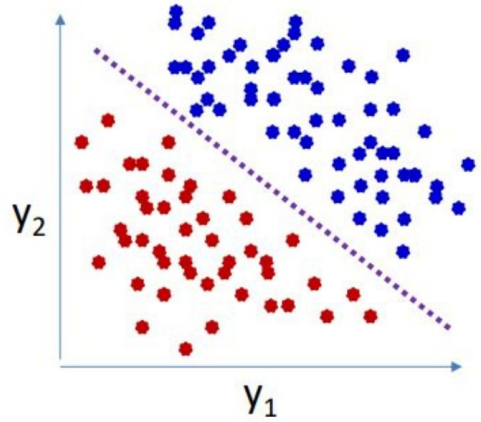
- иметь хорошие признаки (features)
- модель поверх признаков (классификатор, регрессор)

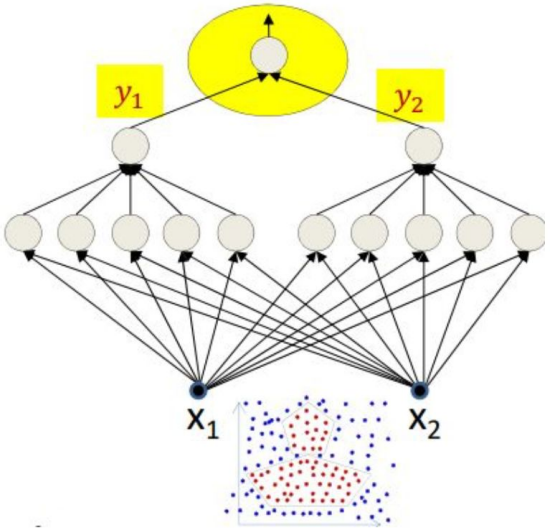
Сегодня поговорим про признаки

Если мы создаем признаки вручную, это называется feature engineering (очень важно в SVM, decision trees, NLP алгоритмах и т.д.)

Если мы делаем это автоматически, это называется **feature learning** или **representation learning**

Пример

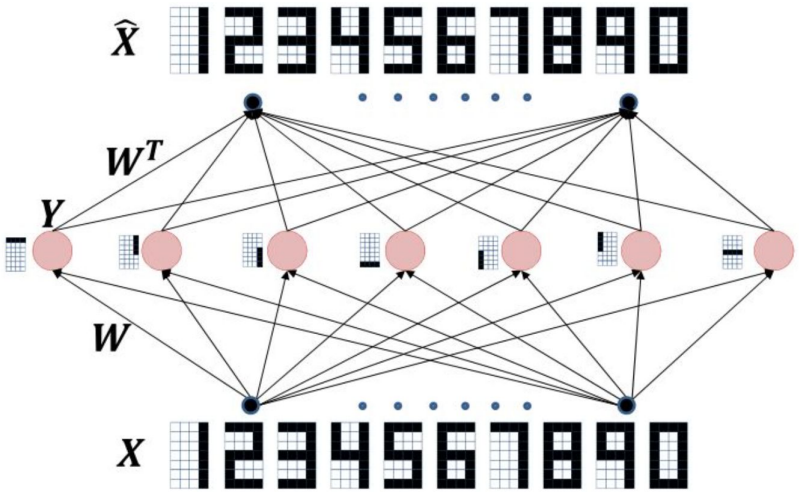




Deep Learning объединяет representation learning и классификацию в одну задачу

Пример

Нейронные сети могут выучить представление данных в unsupervised режиме. Например Autoencoders (Автоенкодеры)



Модель на картинке по сути non-linear PCA

Representation learning как способ думать

- Почему CNN хороши для картинок?
  - Признаки которые вы ищете на картинках обычно translation-invariant
- Почему RNN хороши для последовательностей?
  - Так как зависимости в последовательностях могут быть долговременными
- Почему attention полезен в seq2seq задачах?
  - Потому что слово на выходе зависит от определенных слов на входе

Вместо того, чтобы использовать случайный набор инструментов, моделей для своей задачи, подумайте, какие зависимости в данных вы хотите искать

H: для sentiment analysis CNN подходят лучше чем RNN

Как находить representation

- end-to-end (обычный подход)
- unsupervised (autoencoders)
- на альтернативной задаче
- использовать предобученную модель (н: word embeddings)

Если мы используем готовые представления данных, нас интересуют:

- возможность файнтюнить представления
- фиксировать представления и добавлять больше слоев в сетку

Например

- Transfer learning

  Учим модель на простой задаче, где есть много данных, переносим ее на нашу задачу где данных мало, меняя последний слой сети

Например

- Semi-supervised learning

  Мы хотим научить переводчик Английский-Русский. Одно из решений:
  - Учим encoder-decoder на английских текстах, берем encoder
  - Учим encoder-decoder на русских текстах, берем decoder
  - Учим attention на текстах английский-русский

Unimodal representations

Какие есть уже хорошие предобученные представления данных?

Зависит от модальности: картинки, звук, аудио, видео, ...

- **Images:** берем модель предобученную например на ImageNet (domain-specific! т.е. для медицинских данных ImageNet явно не подойдет)
- **Text:** Word2Vec, BERT, и т.д.
- **Audio:** VGGish
- **Video:** кодируем отдельно звуковой и визуальный ряд
  - в случае визуального ряда, вытаскиваем кадры на фиксированном интервале
  - получаем представления каждого кадра
  - Далее представления можно усреднять либо оставить как последовательность

Multimodal representations

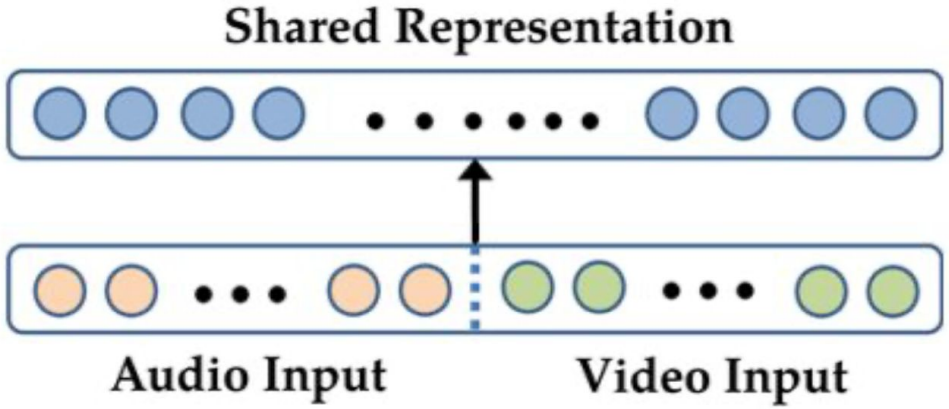
Допустим мы строим вопросно-ответную систему по изображению

Можно легко получить представления каждой модальности в отдельности, картинки и звука

Как теперь будем объединять?

Shallow representations

Просто конкатенируем!

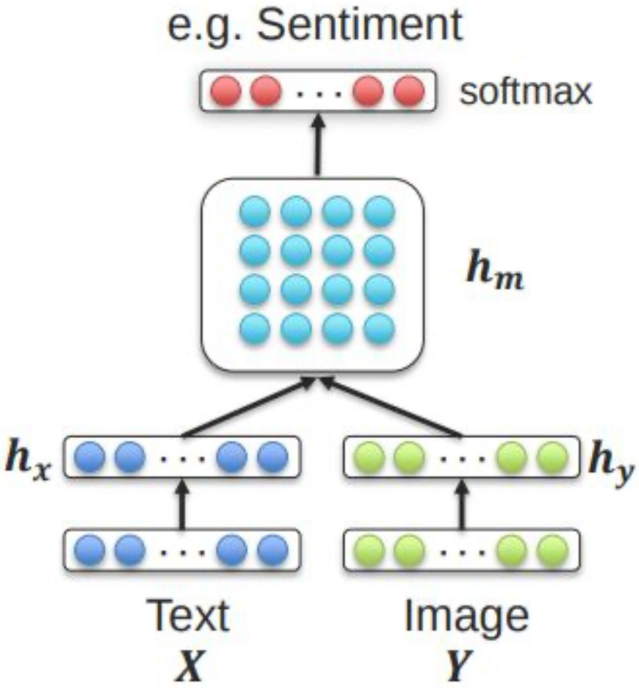


Плюсы: Очень просто

Минусы: Не улавливаем большую часть взаимодействия между модальностями -> weak representation

Bilinear Pooling

Считаем outer product двух векторов
и получаем большой мультимодальный
вектор на выходе



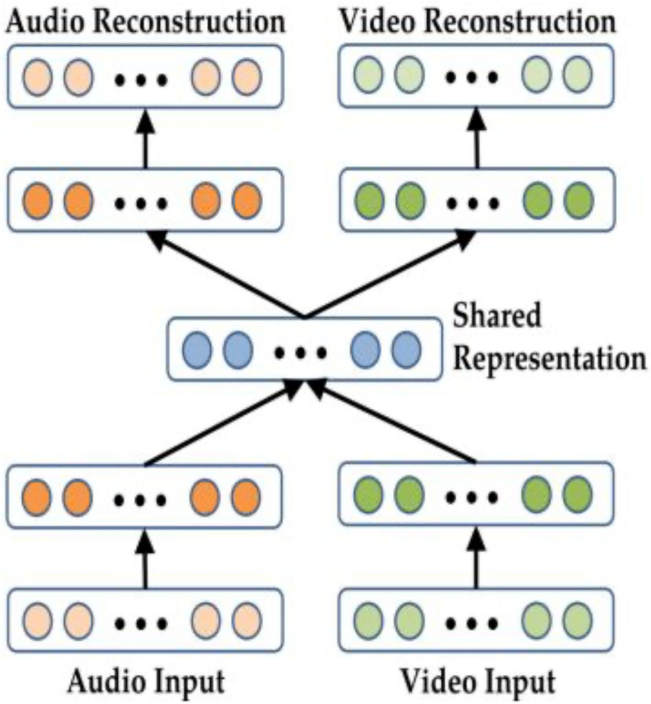
Плюсы:

- Улавливаем почти каждый паттерн
- Отлично для “дополняющих” друг друга модальностей, т.е. которые содержат разную информацию

Минусы:

- Можем получить много лишней информации (redundancy)
- Untrackable? Смотрим Bilinear Pooling или Tensor Fusion

Autoencoders



Единое представление для нескольких модальностей!

Учим в unsupervised режиме. Когда применяем представлений для другой задачи оставляем только Encoder.

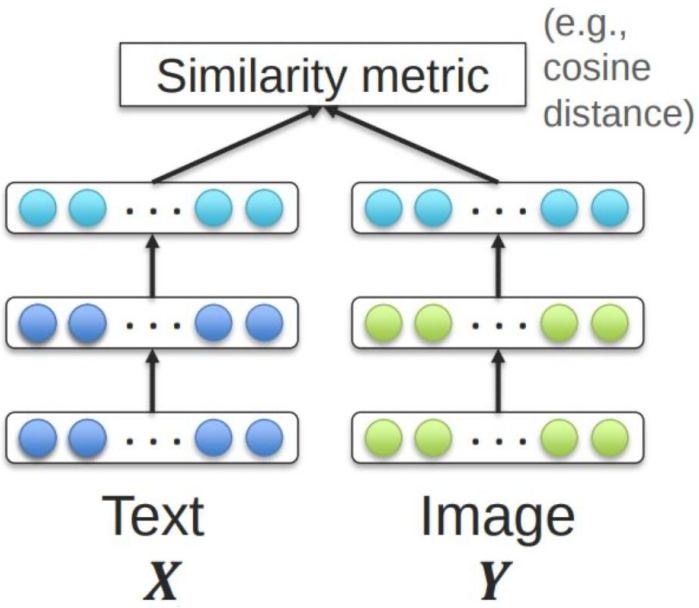
Плюсы:

- robust representation
- если тренируем правильно, можно восстанавливать недостающие модальности

Минусы:

Требует отдельной процедуры обучения, т.к. есть decoder и зачастую не state-of-the-art по сравнению с pooled или coordinated representation

Coordinated representation

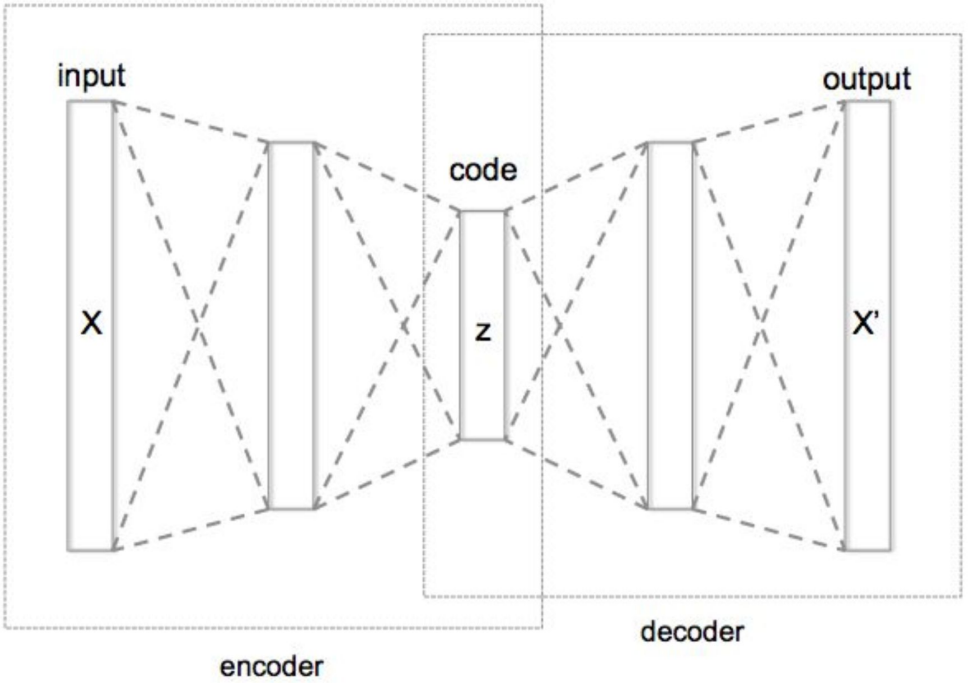


Подробнее об autoencoders

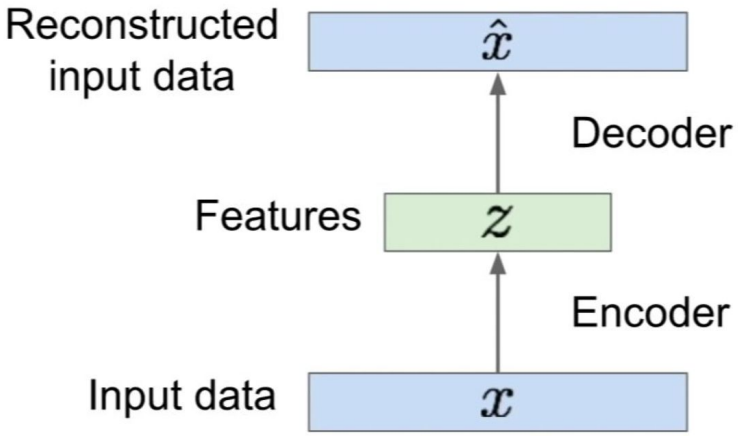
Autoencoder

Выполняет две задачи:

- Representation learning
- Dimensionality reduction



Autoencoder

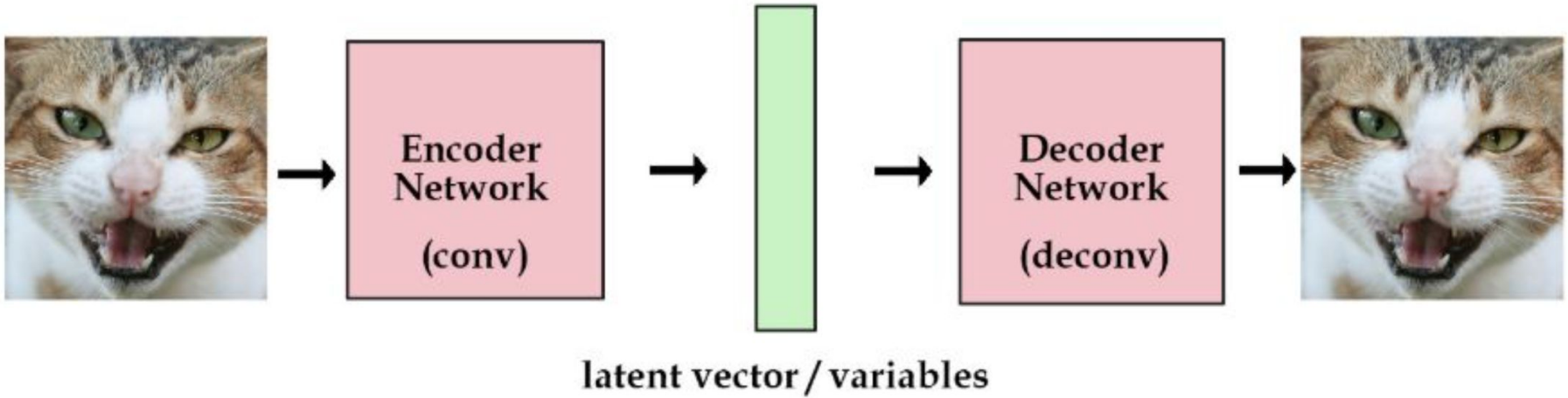


- входные данные имеют большую размерность
- encoder уменьшает размерность
- decoder восстанавливает оригинальные данные
- возможные архитектуры включают:
  - Linear layers + нелинейность
  - Conv, Deconv
  - LSTM, RNN, GRU, etc.
- L2 loss, VGG loss
- обычно очень полезны когда пытаемся извлечь важные признаки

Скрытое представление проходит через bottleneck
Узнали?

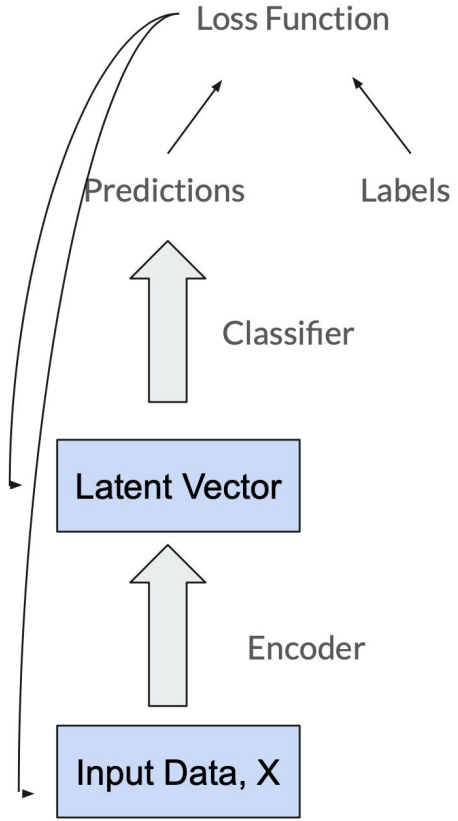
https://vk.com/weirdreparametrizationtrick

Autoencoder



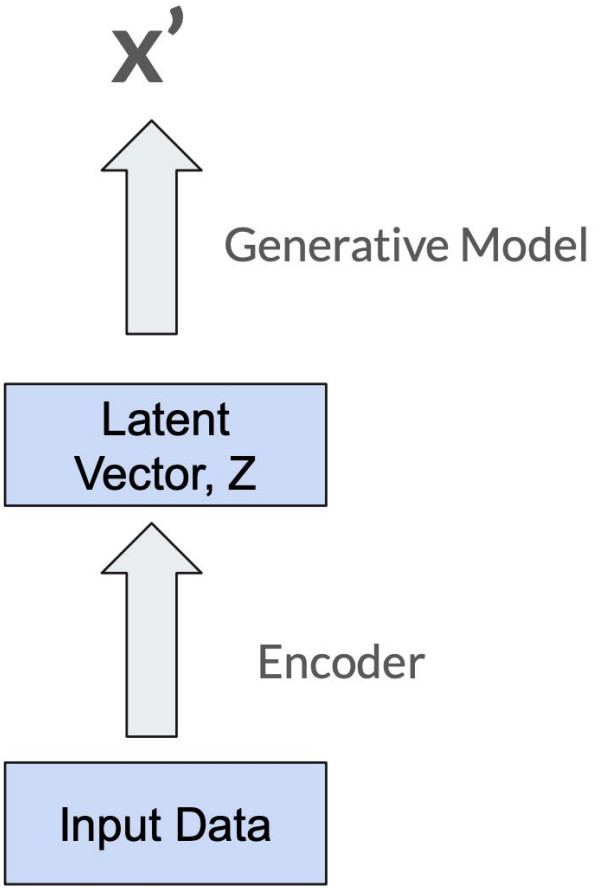
Autoencoder

- можно применять в supervised learning задачах
- оставляем encoder для извлечения признаков
- убираем decoder
- объединяем с классификатором, файнтюним
- имеет смысл когда имеется большое количество неразмеченных данных и мало размеченных



Autoencoder: минусы

- Что если мы захотим генерировать новые данные?
- Сгенерируем сами z и подадим на вход декодеру
- z содержит хорошие признаки
  - В случае лиц, z может содержать форму носа, ширину бровей и т.д.

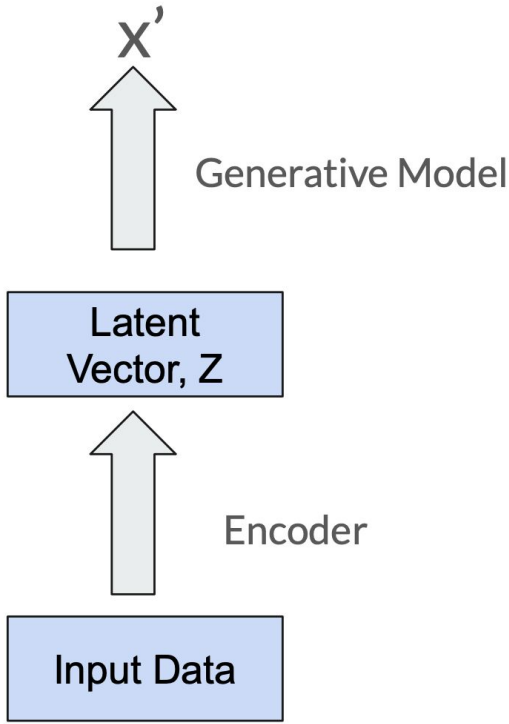


Генеретивная модель на основе AE

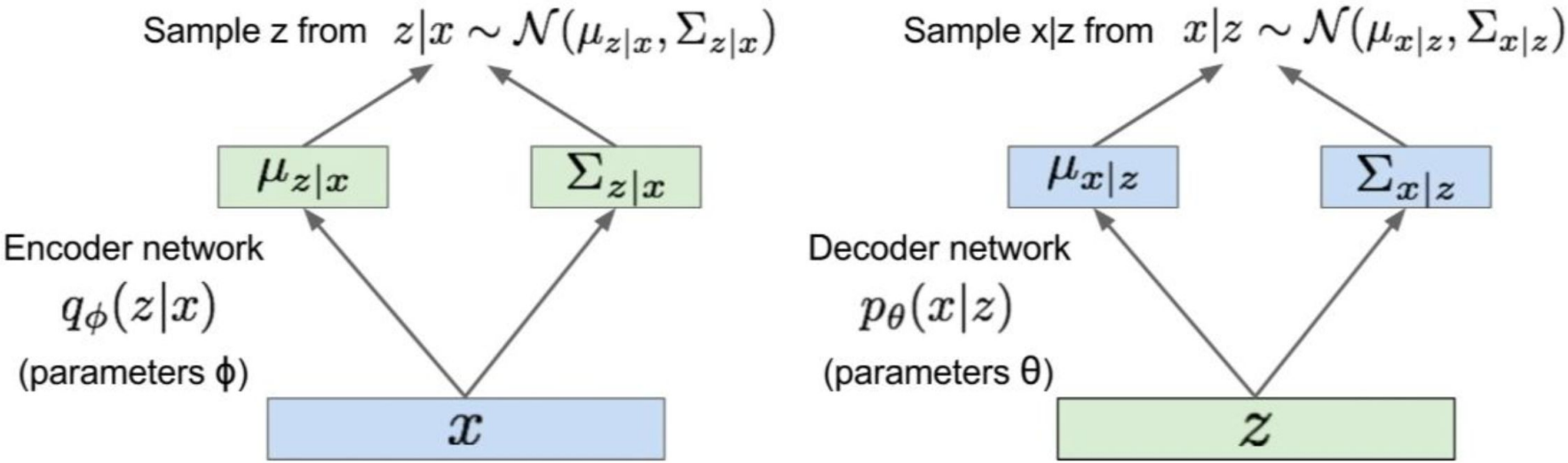
- Наложим априорное распределение на z - prior p(z)
- Сеть заставим учить условное распределение p(x'|z)
- Учим сеть максимизировать likelihood данных из трейна

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

- **Проблема:** как считать p(x|z) для любого z?

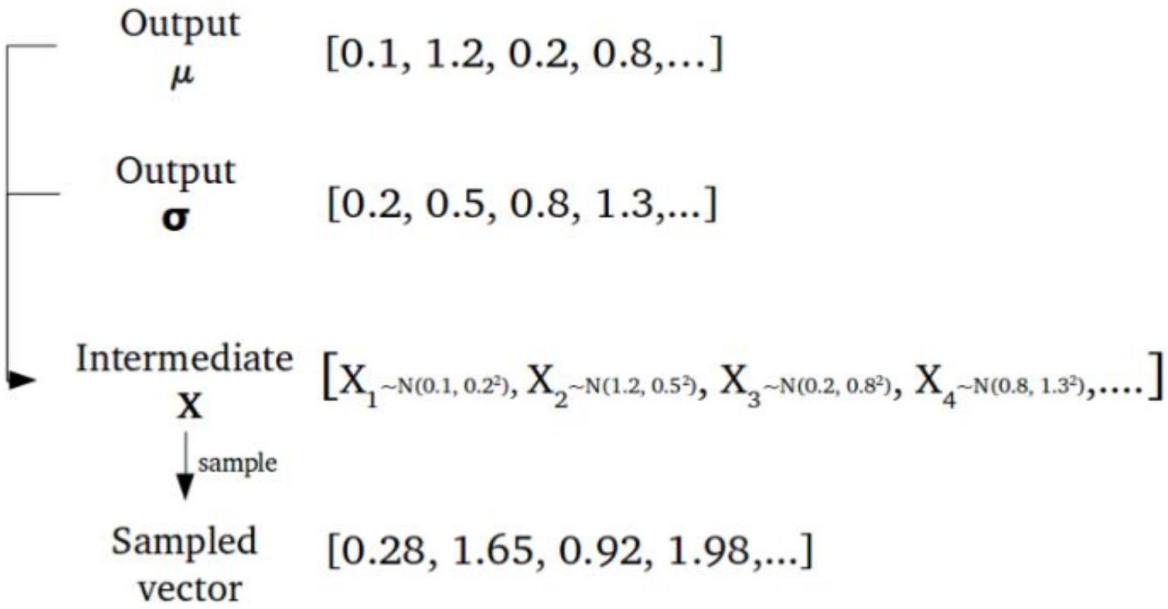


VAE


Sample z from $z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$
$\mu_{z|x}$
$\Sigma_{z|x}$
Encoder network
$q_\phi(z|x)$
(parameters ϕ)
x
Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$
$\mu_{x|z}$
$\Sigma_{x|z}$
Decoder network
$p_\theta(x|z)$
(parameters θ)
z

VAE

Каждый раз когда семплим из распределения, получаем разный x



VAE

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && (\text{Bayes' Rule})\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && (\text{Multiply by constant})\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && (\text{Logarithms})\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}
$$

Можем оценить с помощью сэмплирования

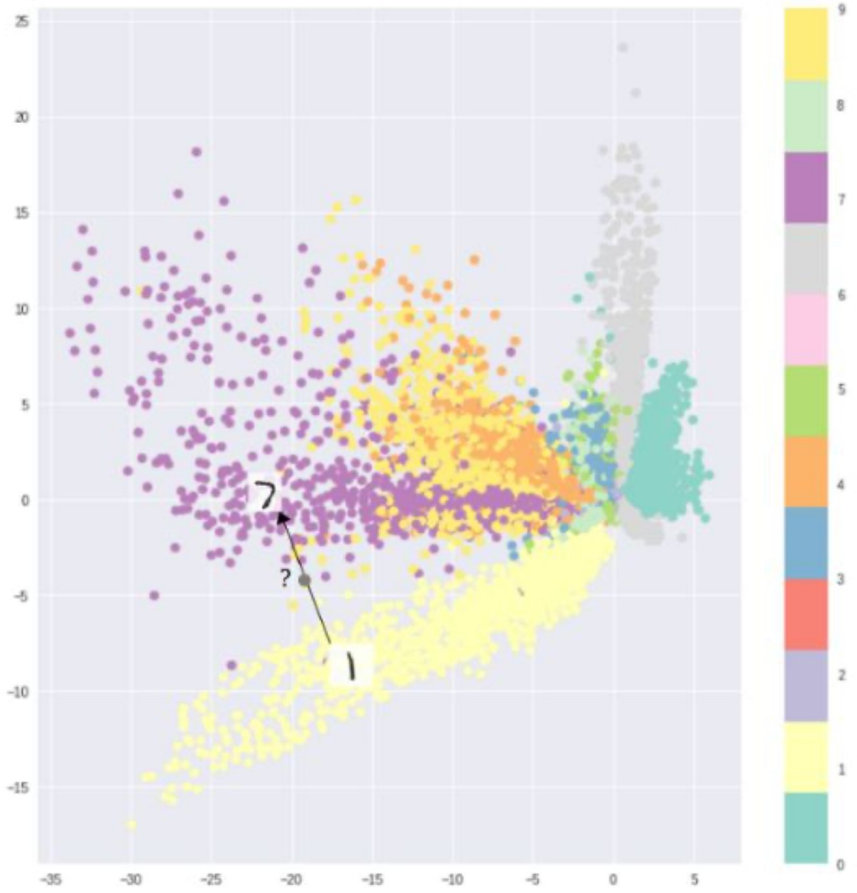
KL дивергенция между двумя гауссовскими распределениями имеет простую аналитическую форму

По определению >= 0

Autoencoder

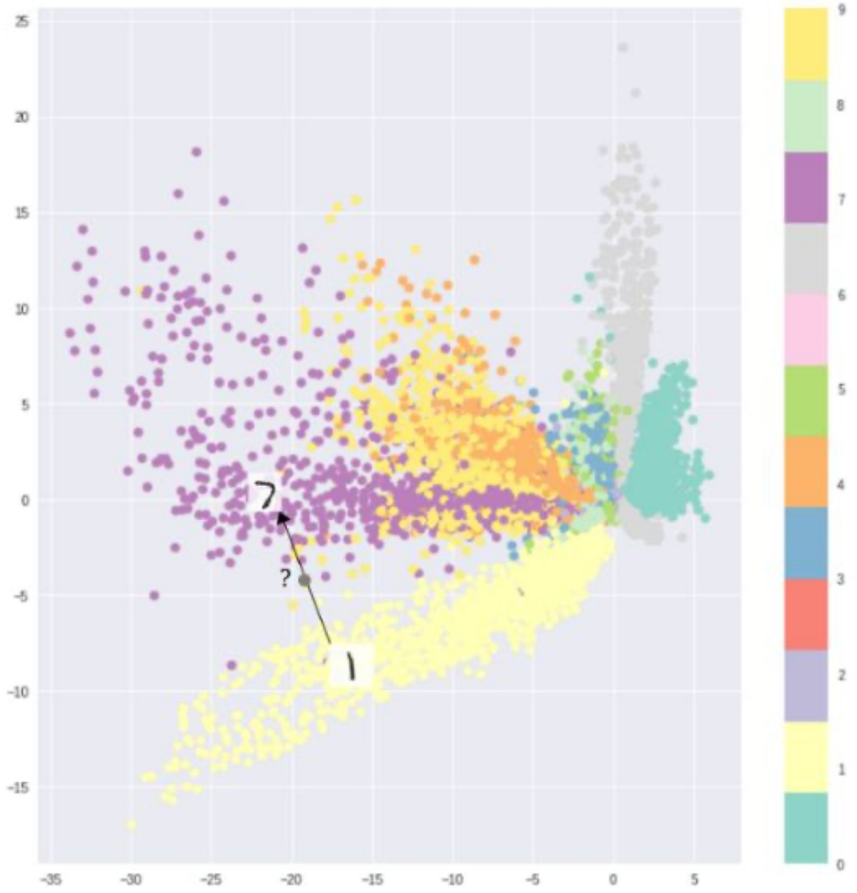
Латентное пространство (там где лежит z), может не являться непрерывным

В этом и состоит проблема генерации новых изображений



Autoencoder: минусы

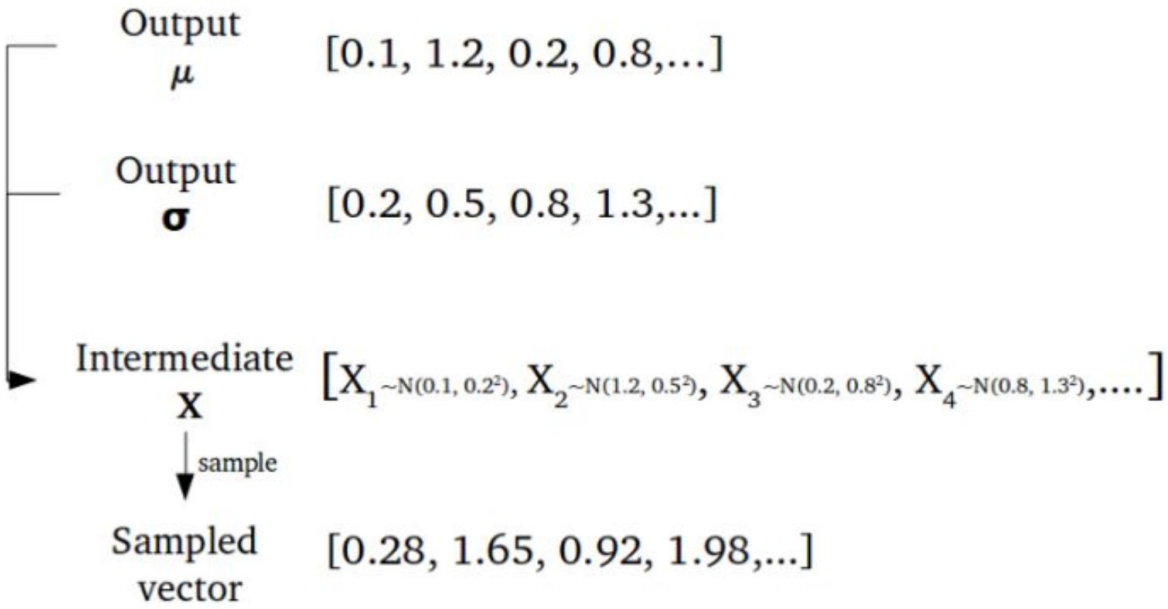
Если случайно генерировать z,
нейросеть не поймет что генерировать на выходе



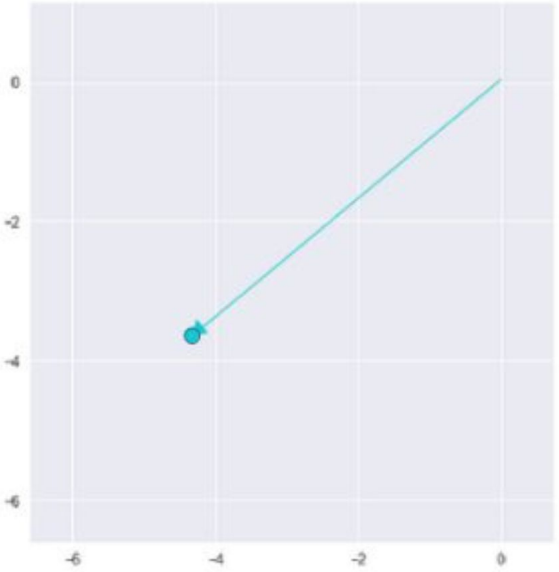
VAE

- Encoder дает на выходе два вектора размерности n, один соответствует mean, второй std

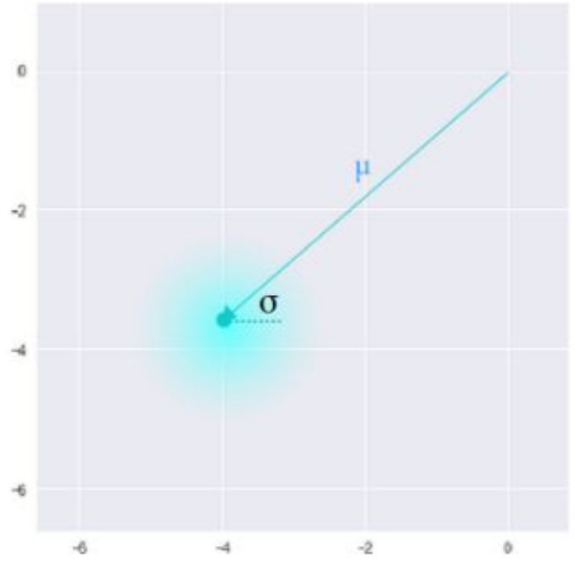
- Стохастичесая генерация, для фиксированного x, mean и std одинаковые, латентный вектор z разный в результате сэмплинга



VAE vs AE

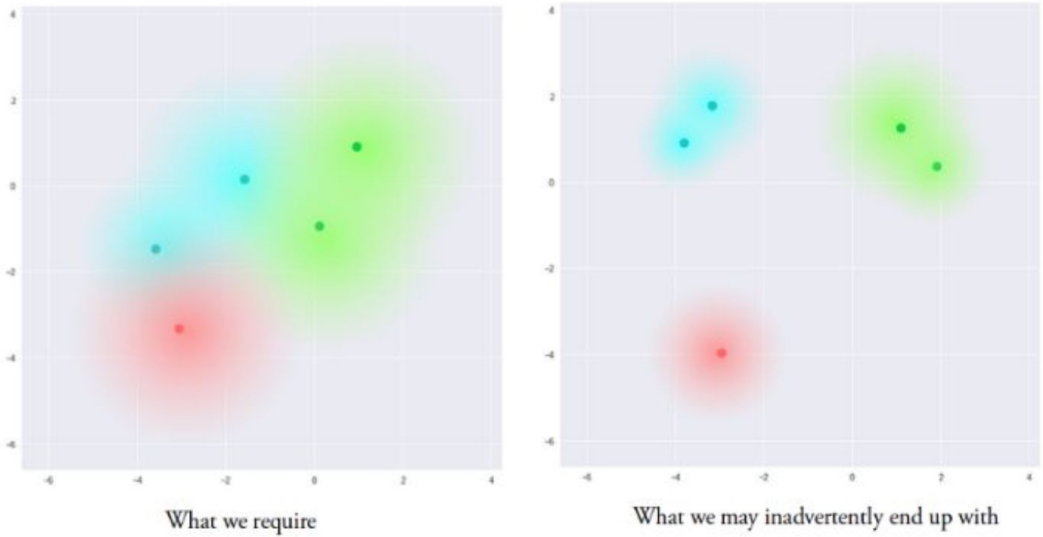


Standard Autoencoder
(direct encoding coordinates)



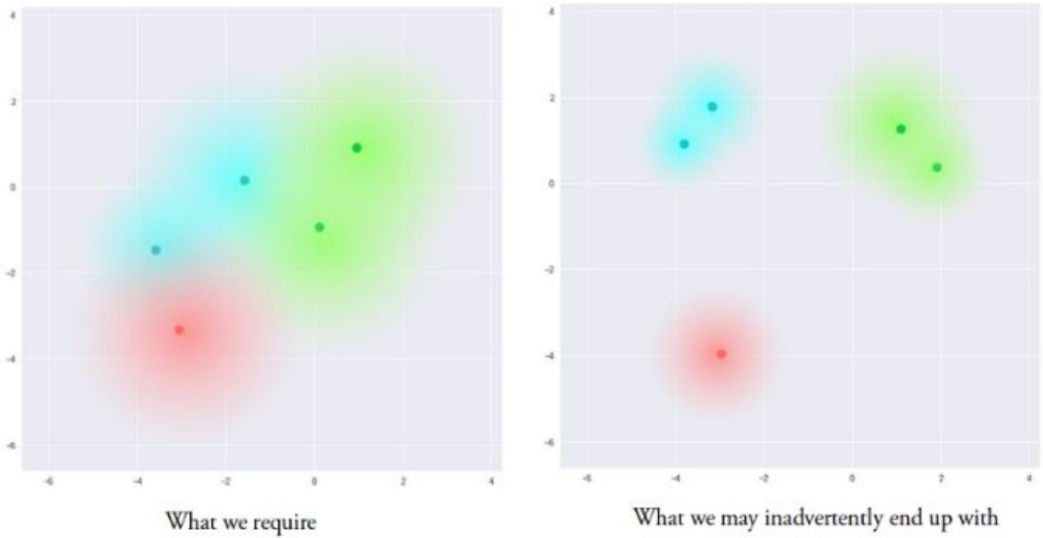
Variational Autoencoder
(μ and σ initialize a probability distribution)

Проблема все равно остается



- все равно имеем пустоты между кластерами
- остатется шанс, что сеть не понимает что ей генерить

Проблема все равно остается



- нет ограничений на mean и variance
- encoder может выучить разные mean для разных классов, а потом минимизировать variance
- в результате меньше неопределенность для декодера

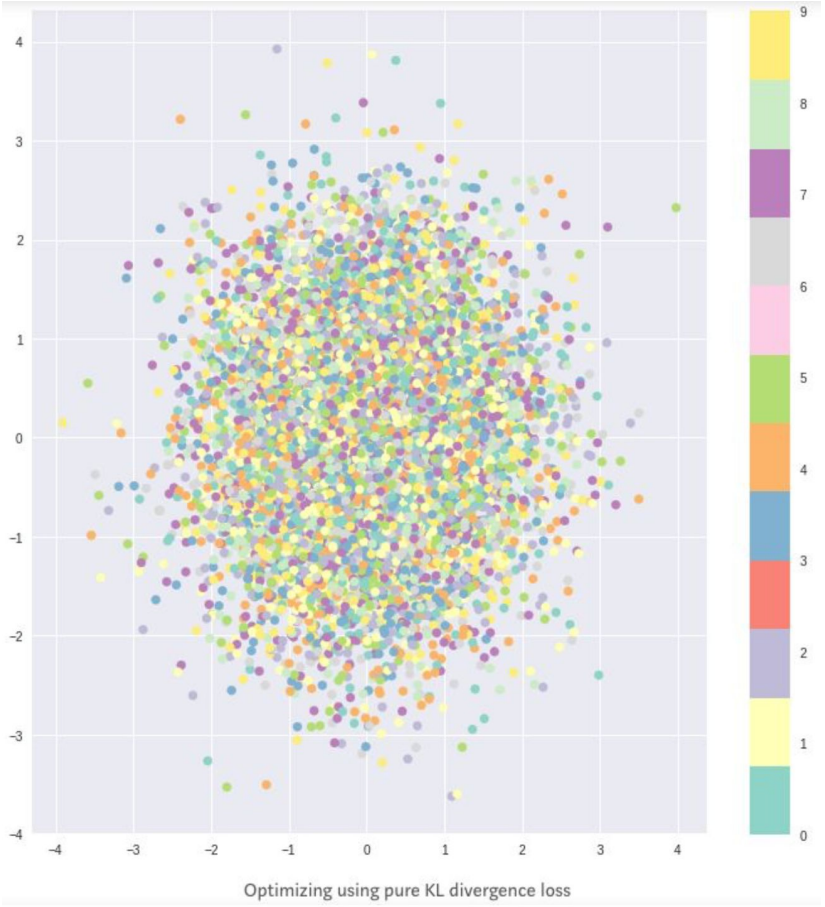
KL divergence

- Измеряет расстояние между двумя распределениями
- Оптимизация по KL divergence означает оптимизацию параметров распределения таким образом, что оно станет похожим на target распределение
- В случае если у X компоненты x_i ~ N(mu_i, sigma_i) и target распределение имеет вид стандартного нормального KL имеет вид:

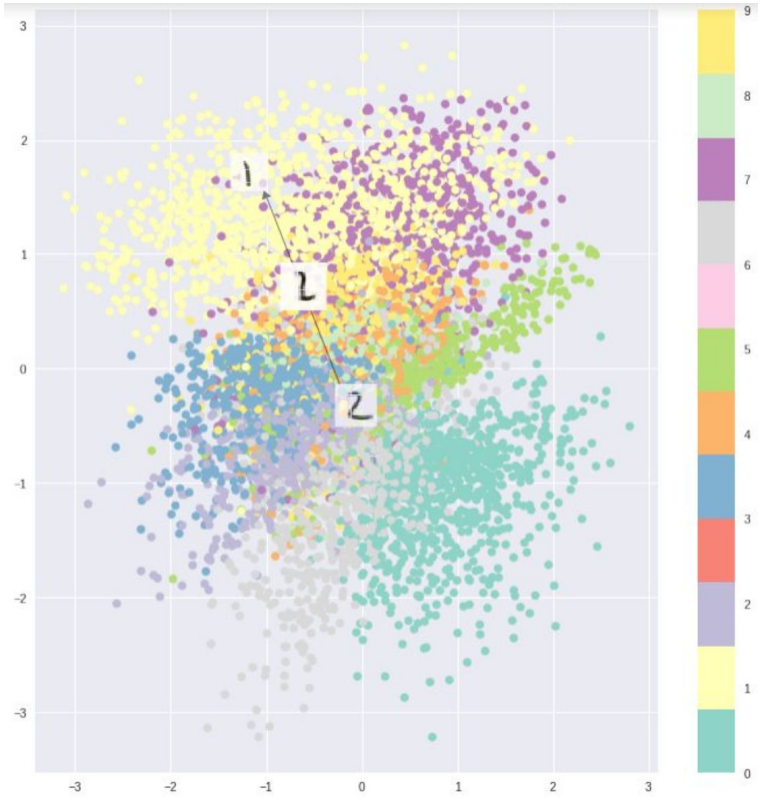
$$\sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1$$

KL divergence

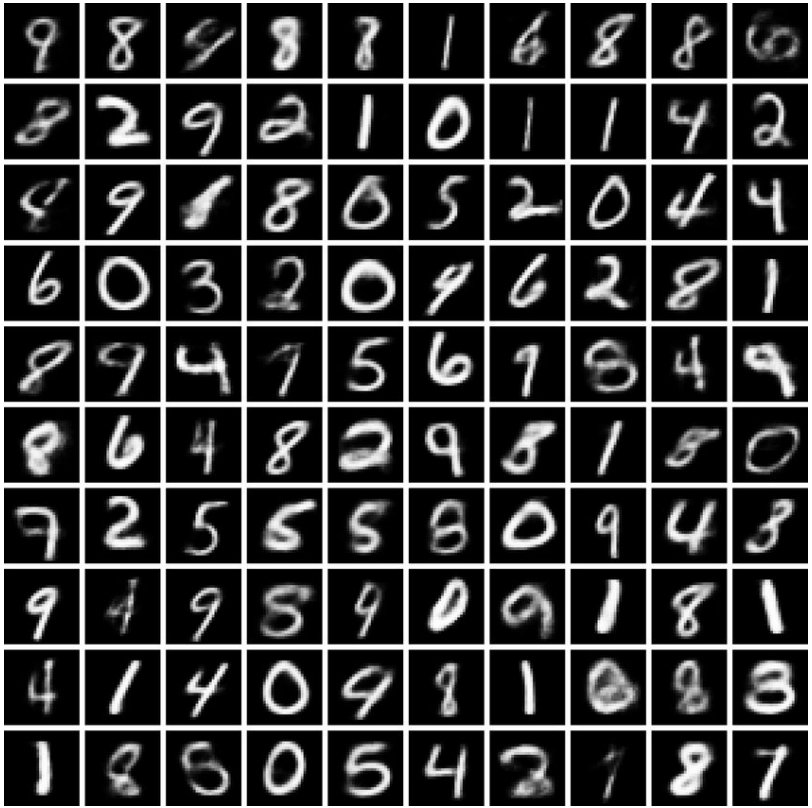
- Заставляет encoder распределять векторы z равномерно между модами target распределения
- Не различает разные классы, не учитывает похожесть объектов одного класса



KL + reconstruction loss



VAE: пример генерации



VAE: пример генерации