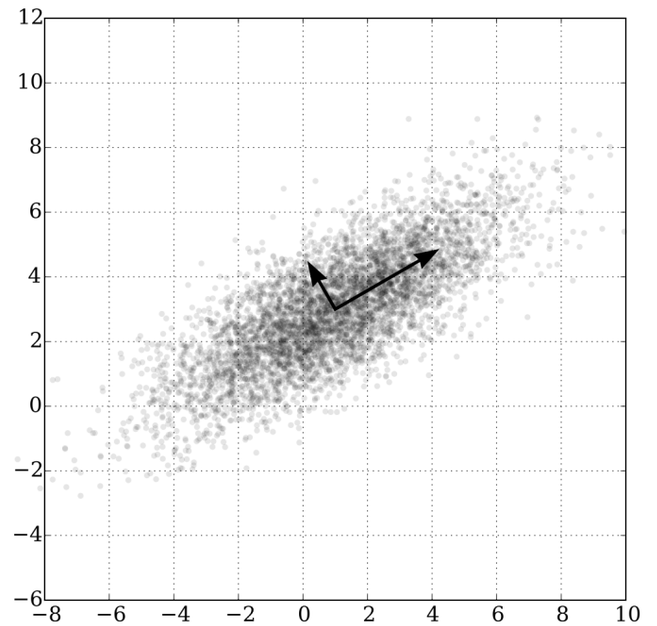
МАШИННОЕ ОБУЧЕНИЕ

AE. VAE. GAN

ПЛАН

- AutoEncoders
- Variational AutoEncoders
- Generative-Adversarial Networks

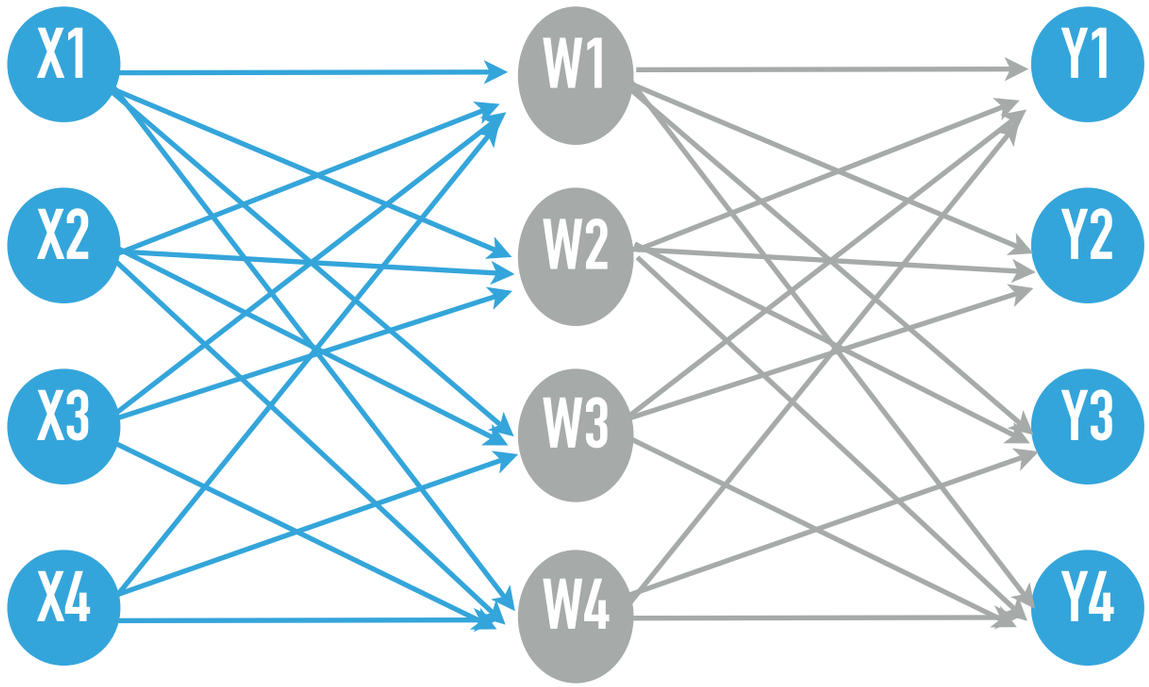
ВСПОМНИМ PCA



- Мы хотим найти такое линейное преобразование (=смену базиса) чтобы базисные векторы были направлены вдоль наибольших дисперсий данных
- Оси с маленькими дисперсиями отбрасываем
- Утверждали что это оптимальное преобразование с точки зрения MSE

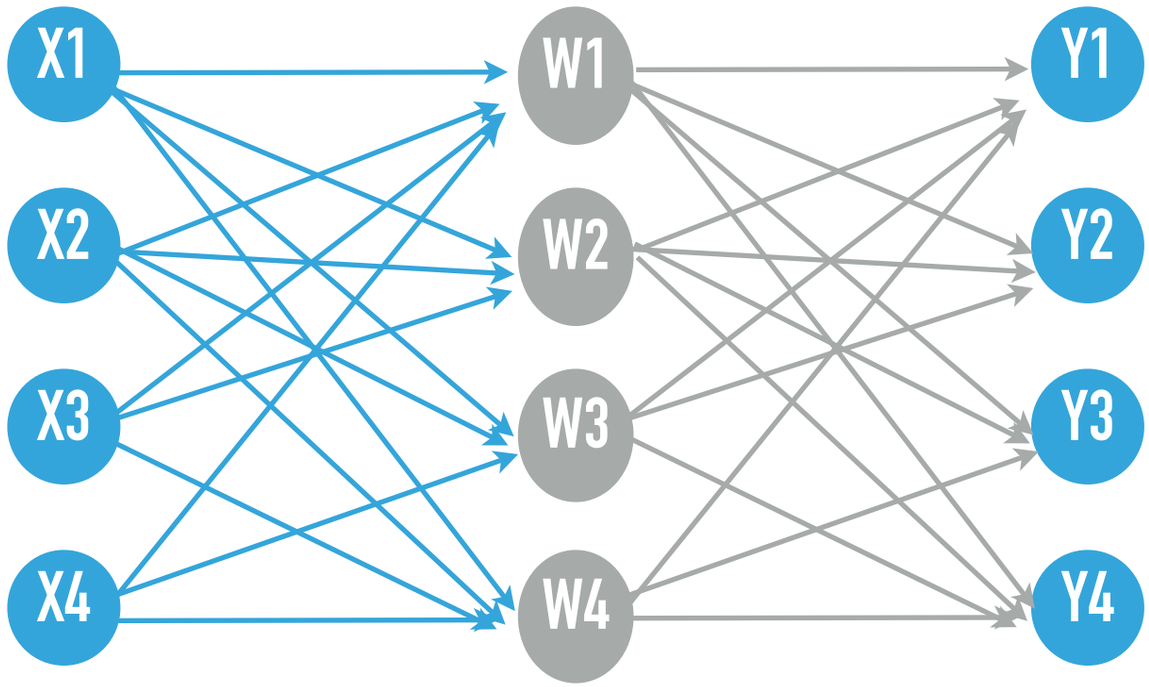
ВСПОМНИМ PCA

- Как повторить тоже самое нейронной сетью?



ВСПОМНИМ PCA

- Как повторить тоже самое нейронной сетью?

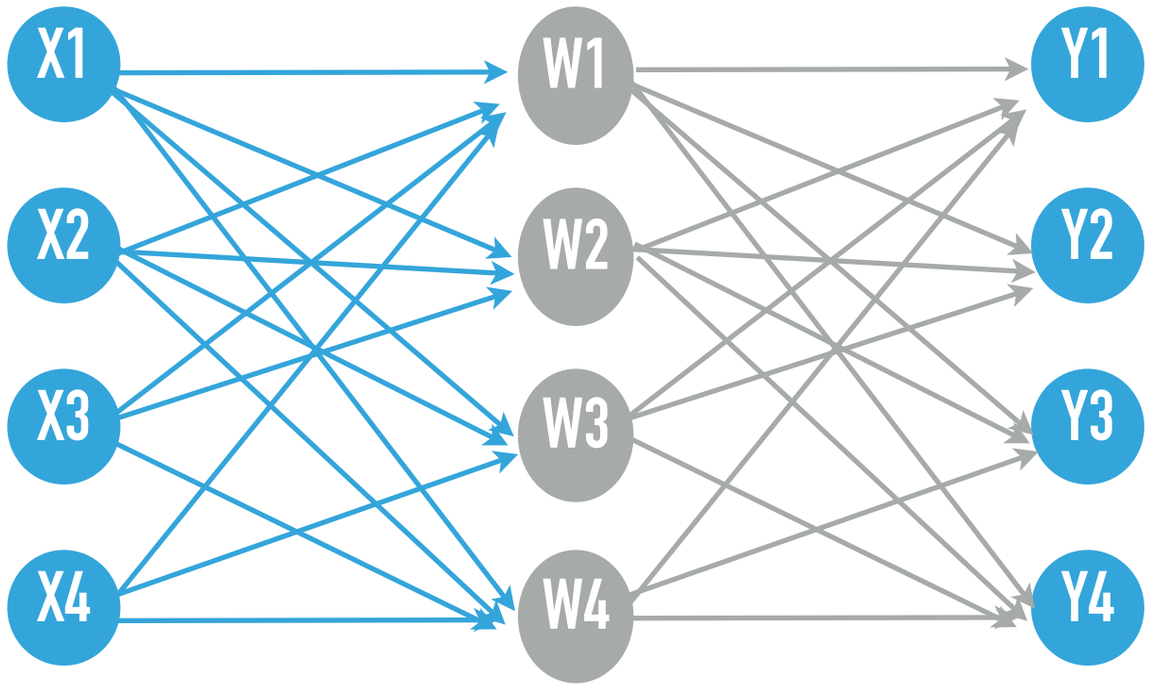


- Рассмотрим сеть с одним скрытым слоем и выходным слоем тех же размерностей что входной слой

- Все функции активации линейные

ВСПОМНИМ PCA

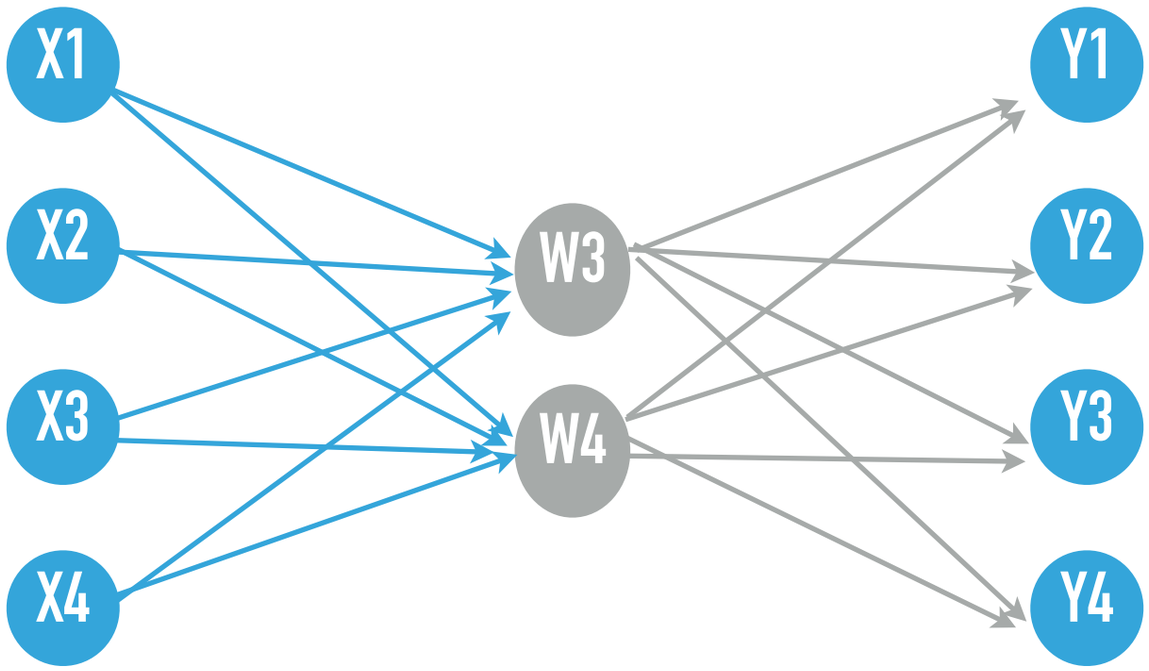
Как повторить тоже самое нейронной сетью?



$$X = (x_1, x_2, x_3, x_4)$$
$$Y = (y_1, y_2, y_3, y_4)$$
$$W = (w_1, w_2, w_3, w_4)$$
$$\hat{X} = Y^t W^t X$$

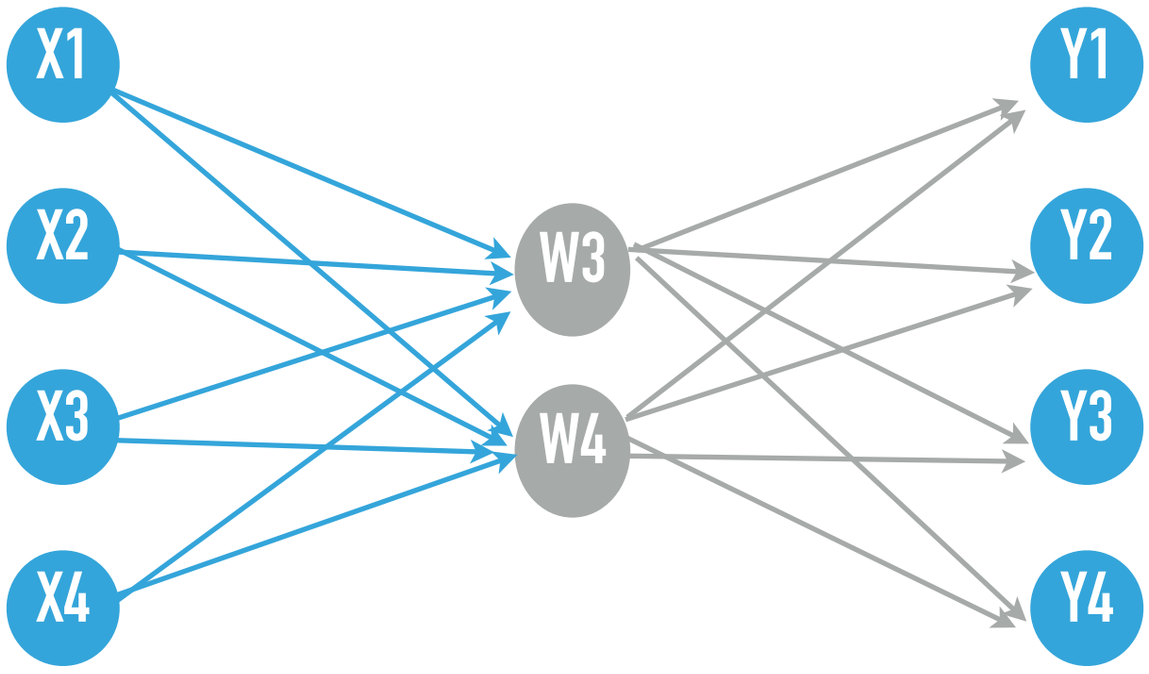
ВСПОМНИМ PCA

- Уменьшая размерность скрытого слоя мы находим оптимальное в среднеквадратичном смысле приближение наших данных



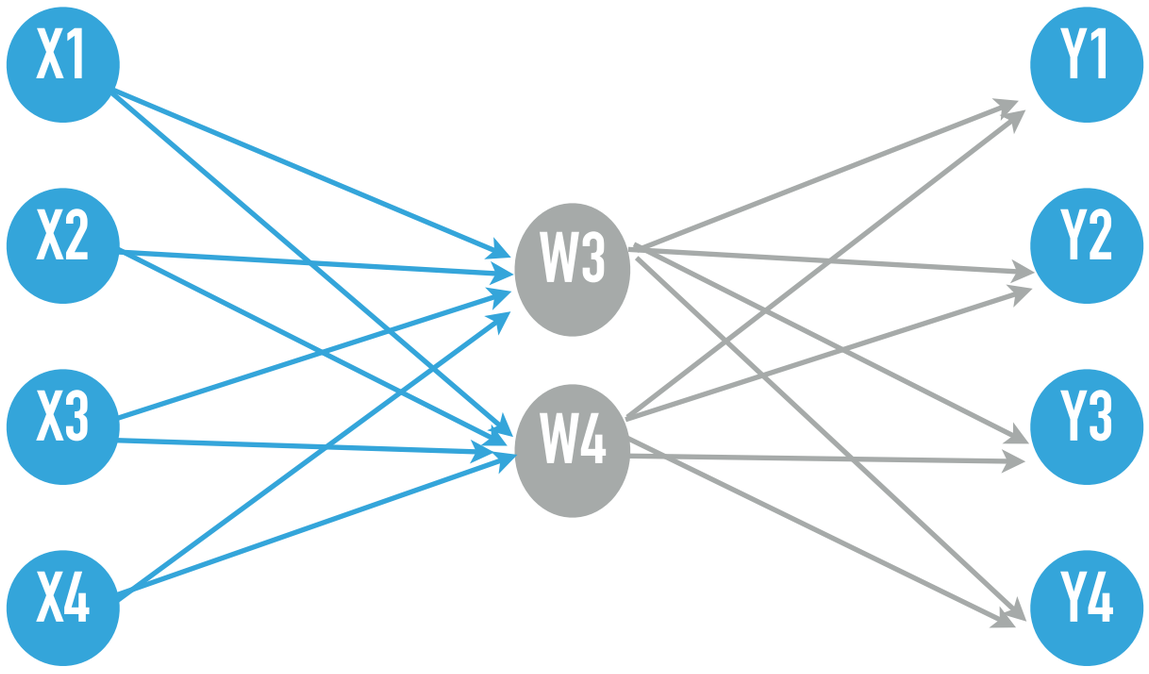
ВСПОМНИМ PCA

- А что если не ограничивать себя линейными функциями активации и одним скрытым слоем?



ПРОВЕДЕМ НЕЛИНЕАРИЗАЦИЮ :)

- Так мы и приходим к автоэнкодерам: моделям, которые по входу пытаются восстановить его же, но пропустив через пространство меньшей размерности



ЗАМЕЧАНИЯ

- Пропустив входной вектор через энкодер мы получаем вектор меньшей размерности, по которому должен быть восстановим исходный вектор
- То есть все те же латентные факторы что рассматривали во многих других задачах
- Никакой разметки нам не нужно
- Природа входных данных может быть абсолютно любой

ЗАМЕЧАНИЯ

- И тут у нас все заведомо нелинейное!

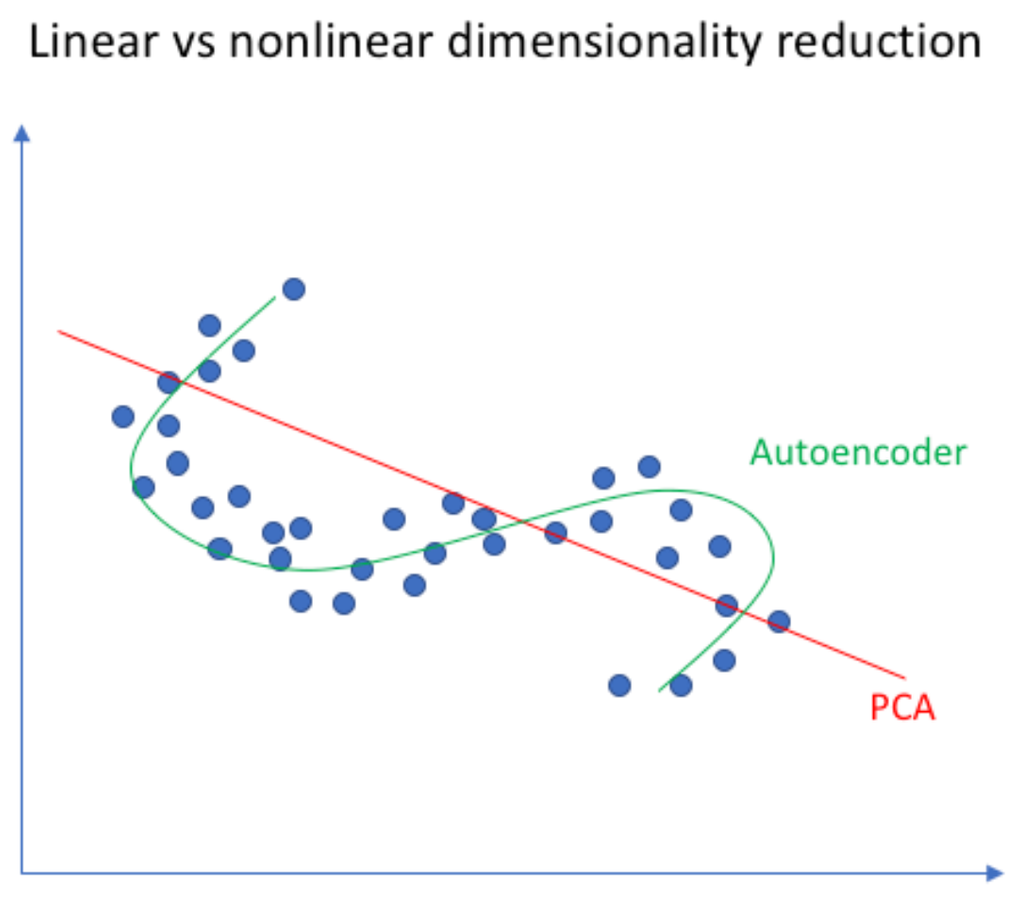
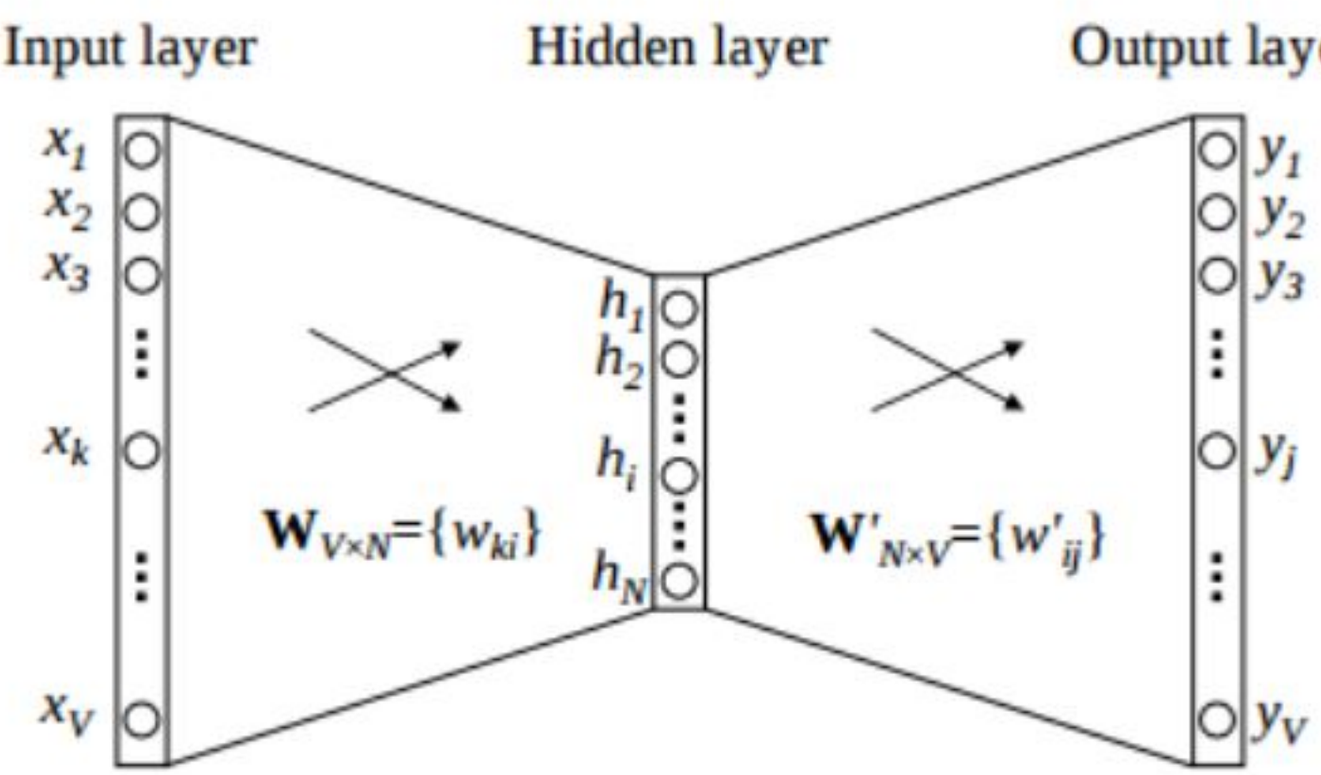


image source

НОВА ЛИ ИДЕЯ?

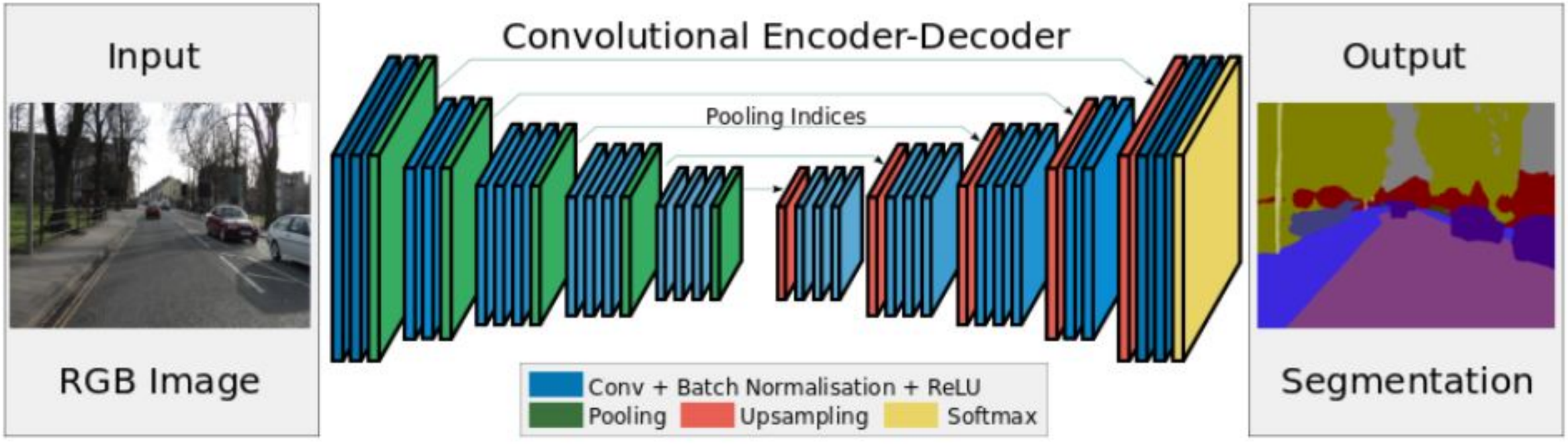
- (Для тех кто знакомы) похоже, например, на word2vec:



word2vec model architecture

НОВА ЛИ ИДЕЯ?

- Или даже модели Semantic Segmentation:



НО ВСЕ ЖЕ НЕТ

- В word2vec предсказывается контекст слова (или слово по контексту). Фактически это был классификатор
- В autoencoder мы именно реконструируем сам объект. Любой природы

НО ВСЕ ЖЕ НЕТ

- В word2vec предсказывается контекст слова (или слово по контексту). Фактически это был классификатор
- В autoencoder мы именно реконструируем сам объект. Любой природы
- На самом деле word2vec тоже можно использовать не только для слов, но наличие контекста принципиально

НО ВСЕ ЖЕ НЕТ

- В word2vec предсказывается контекст слова (или слово по контексту). Фактически это был классификатор
- В autoencoder мы именно реконструируем сам объект. Любой природы
- На самом деле word2vec тоже можно использовать не только для слов, но наличие контекста принципиально
- А в сегментации у нас вообще были лейблы

ВИДЫ АВТОЭНКОДЕРОВ

- Задача любого автоэнкодера - восстановить данные, но не переобучиться. По способу борьбы с переобучением есть разные виды автоэнкодеров:

ВИДЫ АВТОЭНКОДЕРОВ

- Задача любого автоэнкодера - восстановить данные, но не переобучиться. По способу борьбы с переобучением есть разные виды автоэнкодеров:
  - Неполные (undercomplete)
  - Разреженные (sparse)
  - Шупоподавляющий (denoising)

ВИДЫ АВТОЭНКОДЕРОВ

- Задача любого автоэнкодера - восстановить данные, но не переобучиться. По способу борьбы с переобучением есть разные виды автоэнкодеров:
  - Неполные (undercomplete) <--- уже рассмотрели
  - Разреженные (sparse)
  - Шупоподавляющий (denoising)

ВИДЫ АВТОЭНКОДЕРОВ

- Задача любого автоэнкодера - восстановить данные, но не переобучиться. По способу борьбы с переобучением есть разные виды автоэнкодеров:
  - Неполные (undercomplete)
  - Разреженные (sparse)
  - Шумоподавляющий (denoising)

РАЗРЕЖЕННЫЕ АВТОЭНКОДЕРЫ

- Разрешим любую размерность латентного пространства (даже выше исходного пространства), но будем требовать зануления активаций

РАЗРЕЖЕННЫЕ АВТОЭНКОДЕРЫ

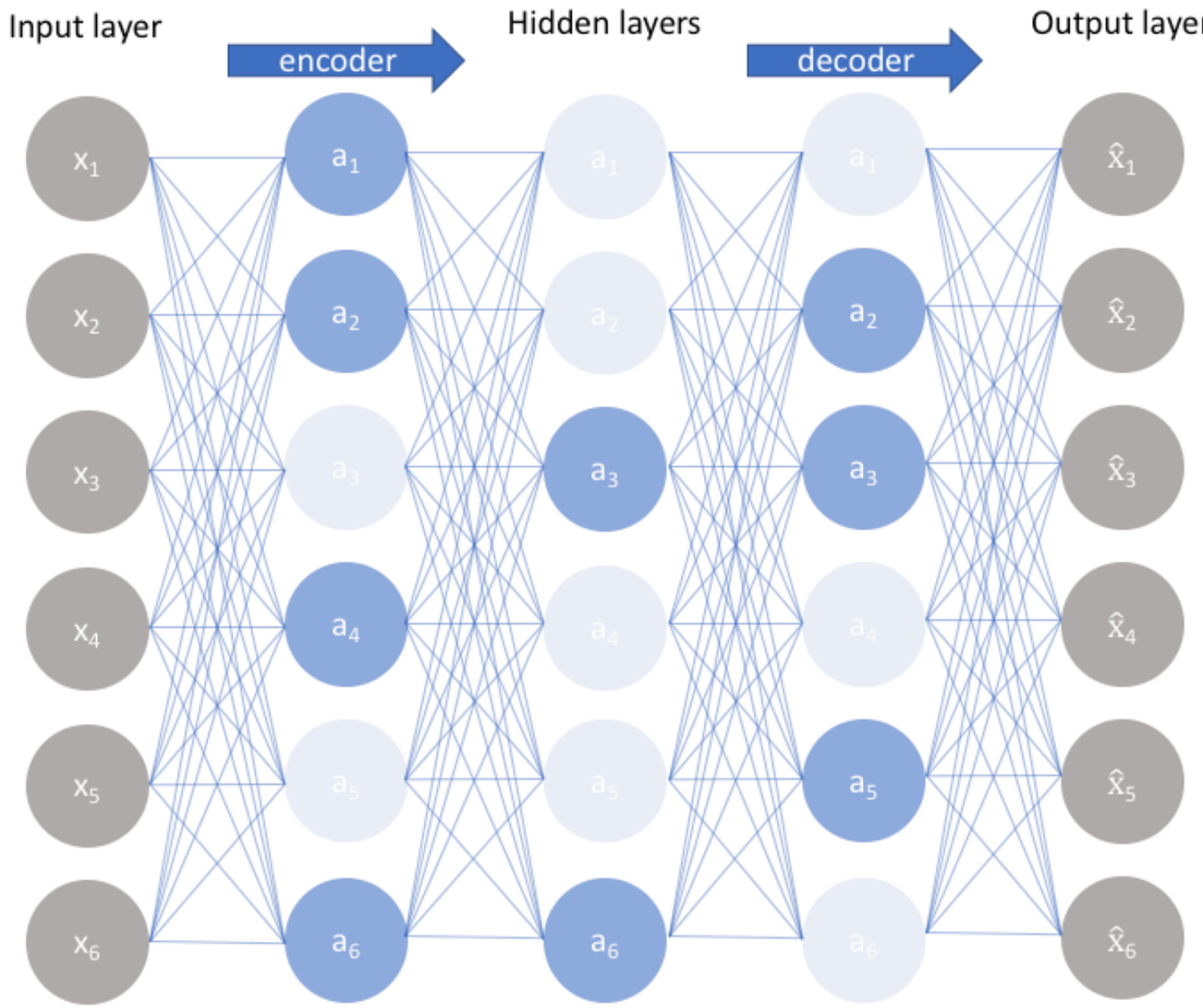


image source

ЧТО ЗНАЧИТ ЗАНУЛЯТЬ АКТИВАЦИИ?

- Пусть a_i^h - выход из i-го нейрона h-го скрытого слоя

- Тогда будем рассматривать такой лосс:

$$L(x, \hat{x}) + \lambda \sum_i \|a_i^h\|$$

где L - ошибка реконструкции

ЧТО ЗНАЧИТ ЗАНУЛЯТЬ АКТИВАЦИИ?

- Пусть a_i^h - выход из i-го нейрона h-го скрытого слоя

- Либо рассмотрим среднее значение этого слоя на батче данных:

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} a_i^h(x)$$

Мы хотим чтобы это среднее равнялось нулю. Это возможно если активации, в основном, будут близки к нулю

РАССТОЯНИЕ КУЛЬБАКА-ЛЕЙБЛЕРА

- Наши лоссы были завязаны на конкретные значения целевой и предсказанной величин
- А иногда хочется чтобы не сами величины, а их распределения принимали желаемый вид

РАССТОЯНИЕ КУЛЬБАКА–ЛЕЙБЛЕРА

- Пусть P, Q - два распределения с плотностями p(x), q(x) соответственно. Тогда расстоянием (дивергенцией) Кульбака-Лейблера называют следующую величину:

РАССТОЯНИЕ КУЛЬБАКА–ЛЕЙБЛЕРА

- Пусть P, Q - два распределения с плотностями p(x), q(x) соответственно. Тогда расстоянием (дивергенцией) Кульбака-Лейблера называют следующую величину:

$$D_{\mathrm{KL}}(P \parallel Q) = \int_X p \log \frac{p}{q} \, \mathrm{d}\mu.$$

РАССТОЯНИЕ КУЛЬБАКА-ЛЕЙБЛЕРА

- Пусть P, Q - два распределения с плотностями p(x), q(x) соответственно. Тогда расстоянием (дивергенцией) Кульбака-Лейблера называют следующую величину:

$$D_{\mathrm{KL}}(P \parallel Q) = \int_X p \log \frac{p}{q} \, \mathrm{d}\mu.$$

- А в дискретном случае:

$$D_{KL}(P \parallel Q) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

РАССТОЯНИЕ КУЛЬБАКА–ЛЕЙБЛЕРА

- Пусть P, Q - два распределения с плотностями p(x), q(x) соответственно. Тогда расстоянием (дивергенцией) Кульбака-Лейблера называют следующую величину:

$$D_{\mathrm{KL}}(P \parallel Q) = \int_X p \log \frac{p}{q} \, \mathrm{d}\mu.$$

- А в дискретном случае:

$$D_{KL}(P \parallel Q) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

- Эта величина не является метрикой, т.к. нет симметричности и не выполнено неравенство треугольника

ВЕРНЕМСЯ К РАЗРЕЖЕННЫМ АВТОЭНКОДЕРАМ

- Мы хотим чтобы активации были нулевыми. Например, можем хотеть чтобы они подчинялись распределению Бернулли. Введем для этого следующий лосс:

$$\sum_{j=1}^{l^{(h)}} \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j}$$

ВИДЫ АВТОЭНКОДЕРОВ

- Задача любого автоэнкодера - восстановить данные, но не переобучиться. По способу борьбы с переобучением есть разные виды автоэнкодеров:
  - Неполные (undercomplete)
  - Разреженные (sparse)
  - Шупоподавляющий (denoising)

ШУМОПОДАВЛЯЮЩИЕ АВТОЭНКОДЕРЫ

- Добавим шум на наши изображения, а предсказывать будем исходные версии без шума
- За счет того что теперь не предсказываем само изображение "запоминать" его нет смысла

ШУМОПОДАВЛЯЮЩИЕ АВТОЭНКОДЕРЫ

- Добавим шум на наши изображения, а предсказывать будем исходные версии без шума

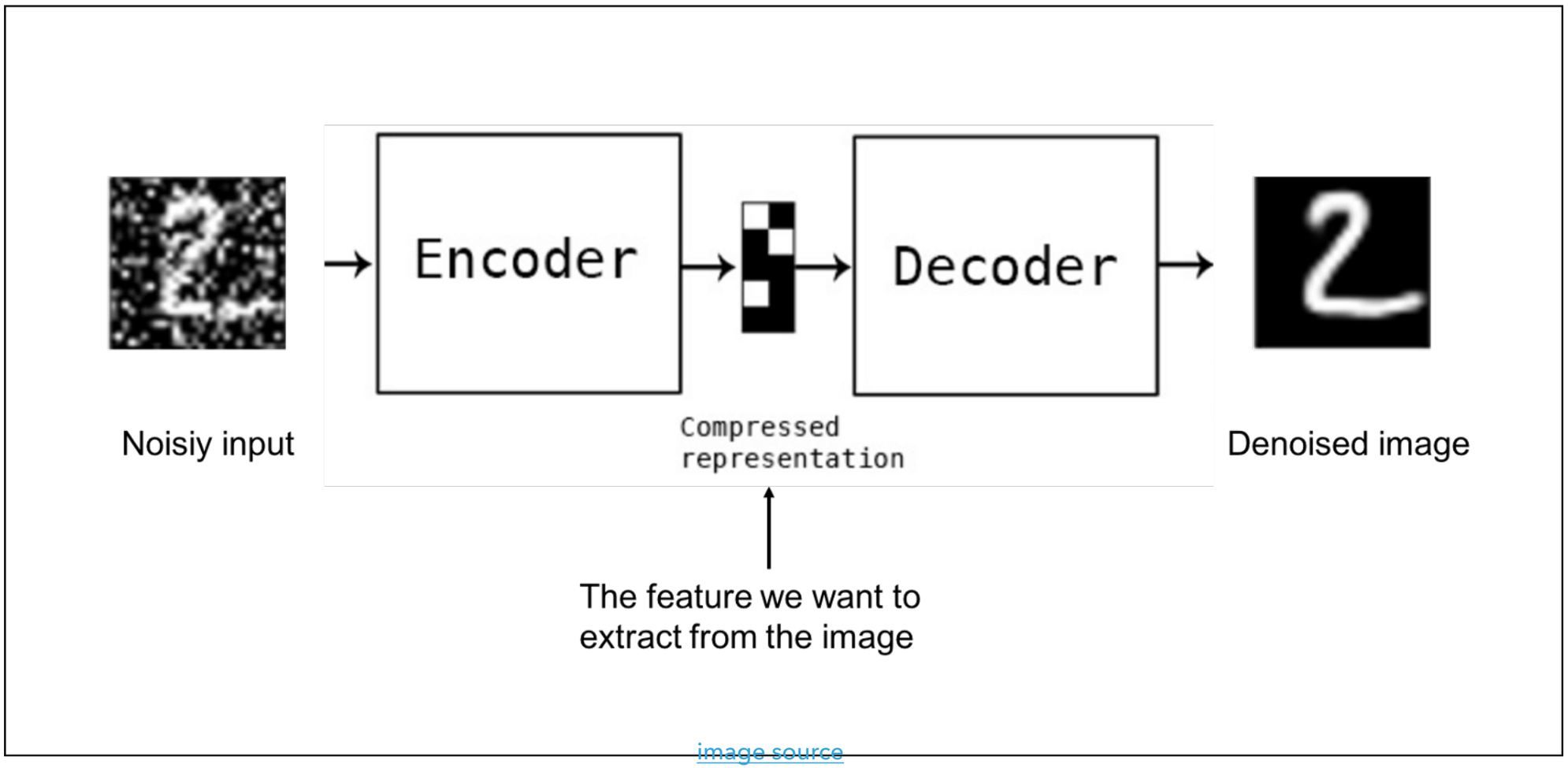


image source

ПРИЛОЖЕНИЯ АВТОЭНКОДЕРОВ

- Понижение размерности данных / создание фичей
- Сжатие данных
- Избавление от шумов
- Детектирование аномалий

ПРИЛОЖЕНИЯ АВТОЭНКОДЕРОВ

- Понижение размерности данных / создание фичей
  - Пропустили через encoder -> имеем фичи
- Сжатие данных
- Избавление от шумов
- Детектирование аномалий

ПРИЛОЖЕНИЯ АВТОЭНКОДЕРОВ

- Понижение размерности данных / создание фичей
- Сжатие данных
  - Обменялись декодерами. Передали человеку результат энкодера -> он декодировал
- Избавление от шумов
- Детектирование аномалий

ПРИЛОЖЕНИЯ АВТОЭНКОДЕРОВ

- Понижение размерности данных / создание фичей
- Сжатие данных
- Избавление от шумов
  - Обучили denoising autoencoder -> пропускаем зашумленное изображение через него целиком
- Детектирование аномалий

ПРИЛОЖЕНИЯ АВТОЭНКОДЕРОВ

- Понижение размерности данных / создание фичей
- Сжатие данных
- Избавление от шумов
- Создание фичей
- Детектирование аномалий
  - Обучили автоэнкодер на нормальных данных. Подаем на вход аномалию -> у декодера будет высокая ошибка восстановления

ДИСКРИМИНАТИВНЫЕ VS ГЕНЕРАТИВНЫЕ МОДЕЛИ

Пусть мы решаем задачу классификации

Нас интересует отображение f: X -> Y или P(Y|X)

ДИСКРИМИНАТИВНЫЕ VS ГЕНЕРАТИВНЫЕ МОДЕЛИ

Пусть мы решаем задачу классификации

Нас интересует отображение f: X -> Y или P(Y|X)

Дискриминативный (discriminative) подход:

- Предполагаем функциональную форму для P(Y|X)
- Обучаем параметры P(Y|X)

ДИСКРИМИНАТИВНЫЕ VS ГЕНЕРАТИВНЫЕ МОДЕЛИ

Пусть мы решаем задачу классификации

Нас интересует отображение f: X -> Y или P(Y|X)

Дискриминативный (discriminative) подход:
- Предполагаем функциональную форму для P(Y|X)
- Обучаем параметры P(Y|X)

Генеративный (generative) подход:
- Предполагаем функциональную форму для P(X|Y)
- Оцениваем P(Y) по данным, обучаем P(X|Y)
- По теореме Байеса получаем P(Y|X)

МОЖНО ЛИ ГЕНЕРИРОВАТЬ ДАННЫЕ АВТОЭНКОДЕРОМ?

- Автоэнкодер переводит данные в латентное пространство
- Можно ли взять точку сразу из этого пространства, отдать декодеру и получить новый семпл?

МОЖНО ЛИ ГЕНЕРИРОВАТЬ ДАННЫЕ АВТОЭНКОДЕРОМ?

- Автоэнкодер переводит данные в латентное пространство
- Можно ли взять точку сразу из этого пространства, отдать декодеру и получить новый семпл?
- Но пространства, получаемые автоэнкодером, слишком рванные :(
- От них и не ожидалось что они будут непрерывными

МОЖНО ЛИ ГЕНЕРИРОВАТЬ ДАННЫЕ АВТОЭНКОДЕРОМ?

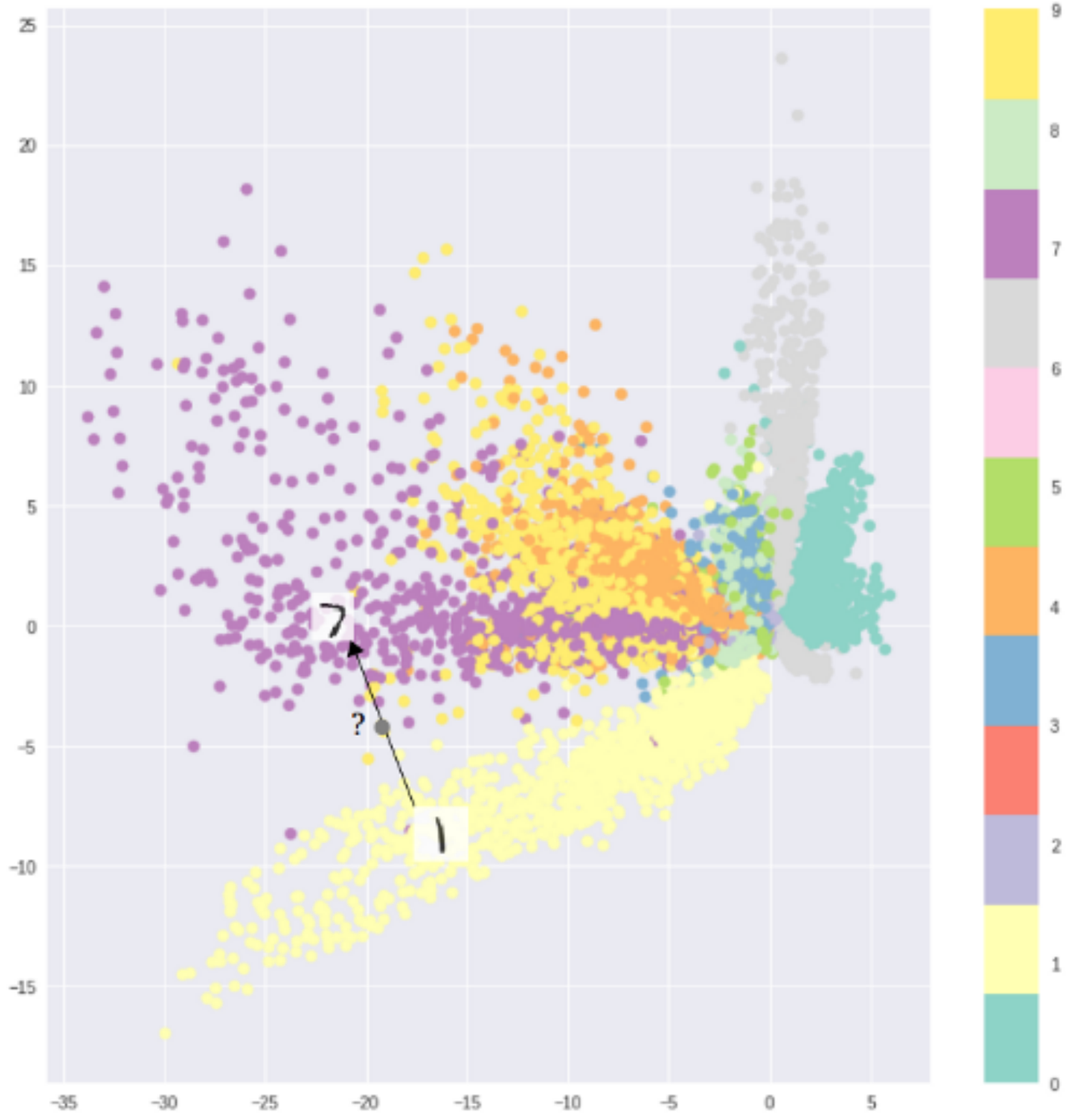


image source

ВАРИАЦИОННЫЕ АВТОЭНКОДЕРЫ

- Давайте заставим автоэнкодер находить "хорошие" пространства:
  - Пусть энкодер возвращает два вектора: вектор средних и вектор стандартных отклонений
  - Декодер будет семплировать нормальное распределение с этими параметрами и декодировать полученную точку

ВАРИАЦИОННЫЕ АВТОЭНКОДЕРЫ

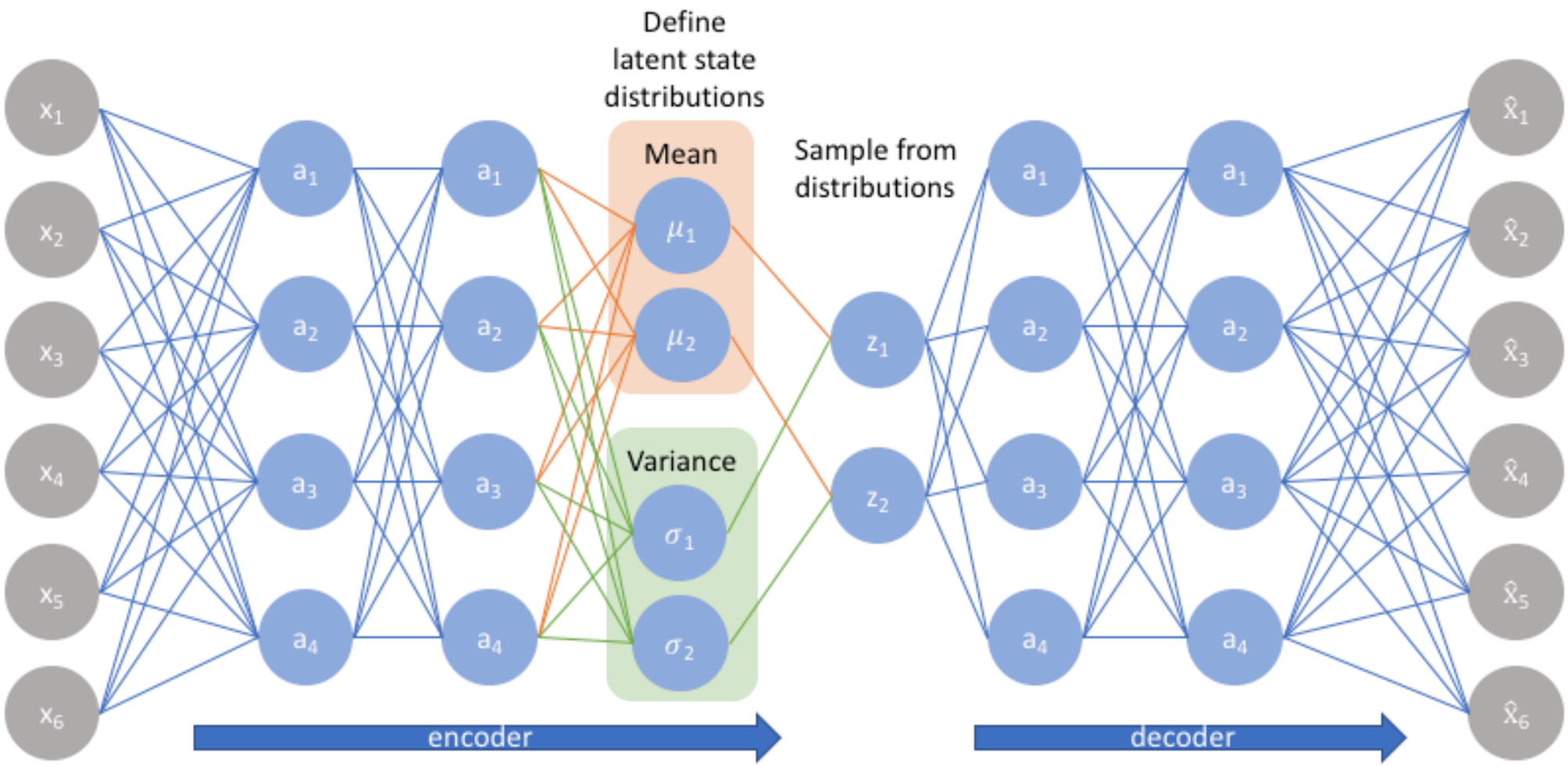


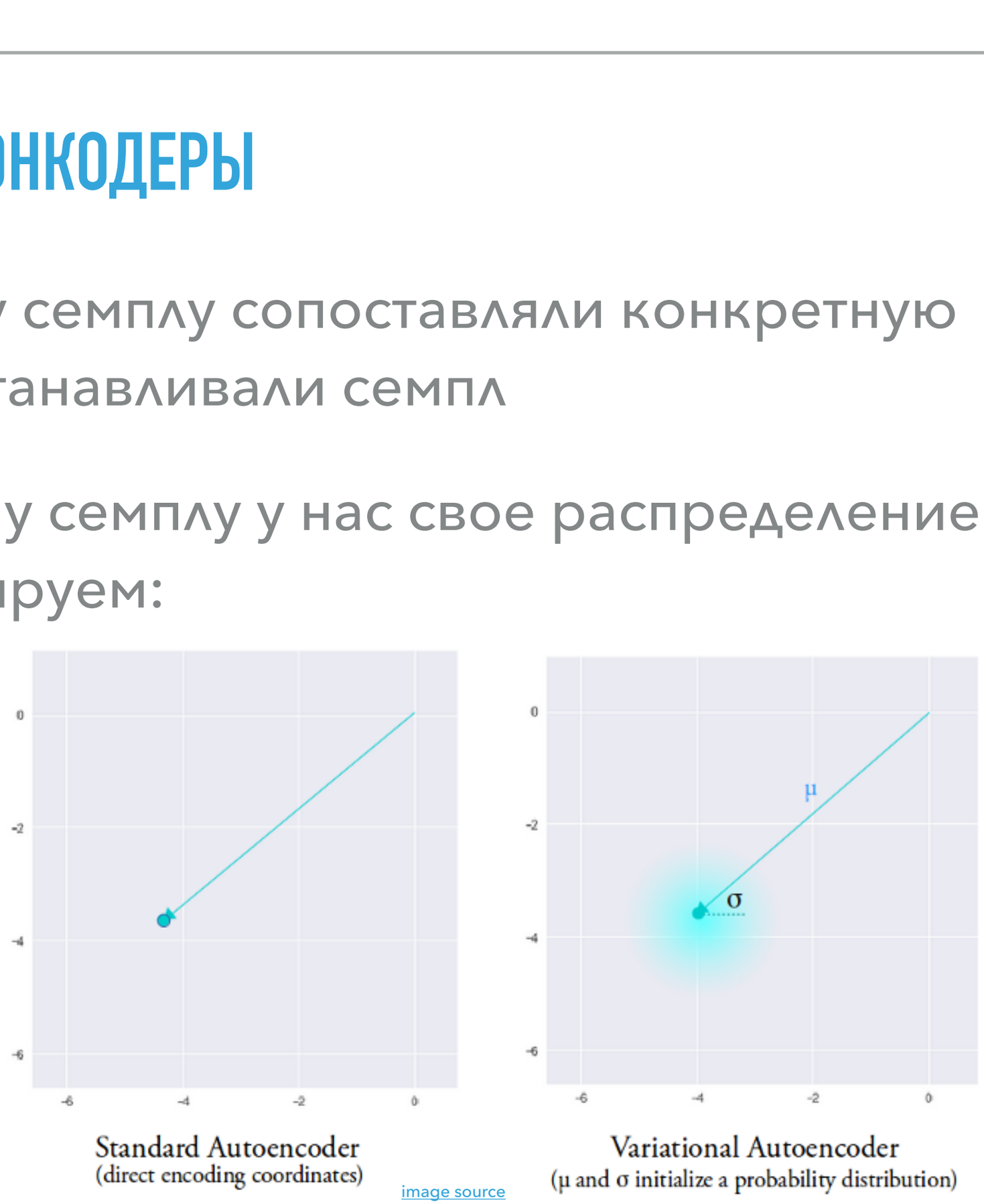
image source

ВАРИАЦИОННЫЕ АВТОЭНКОДЕРЫ

- Раньше мы каждому семплу сопоставляли конкретную точку и по ней восстанавливали семпл
- А теперь по каждому семплу у нас свое распределение из которого мы семплируем:

ВАРИАЦИОННЫЕ АВТОЭНКОДЕРЫ

- Раньше мы каждому семплу сопоставляли конкретную точку и по ней восстанавливали семпл
- А теперь по каждому семплу у нас свое распределение из которого мы семплируем:



Standard Autoencoder
(direct encoding coordinates)

image source

Variational Autoencoder
(μ and σ initialize a probability distribution)

ВАРИАЦИОННЫЕ АВТОЭНКОДЕРЫ

- Раньше мы каждому семплу сопоставляли конкретную точку и по ней восстанавливали семпл
- А теперь по каждому семплу у нас свое распределение из которого мы семплируем
- Получается что даже подавая один и тот же семпл декодеру будут приходить разные точки, и он должен уметь работать с "островком" вокруг каждой точки

ПОЛУЧИЛИ ЛИ ТО ЧТО ХОТЕЛИ?

- Теперь у нас локально вокруг каждой точки "хорошее" пространство. Но что если mu и sigma разных семплов будут далеко друг от друга?

ПОЛУЧИЛИ ЛИ ТО ЧТО ХОТЕЛИ?

- Теперь у нас локально вокруг каждой точки "хорошее" пространство. Но что если mu и sigma разных семплов будут далеко друг от друга?

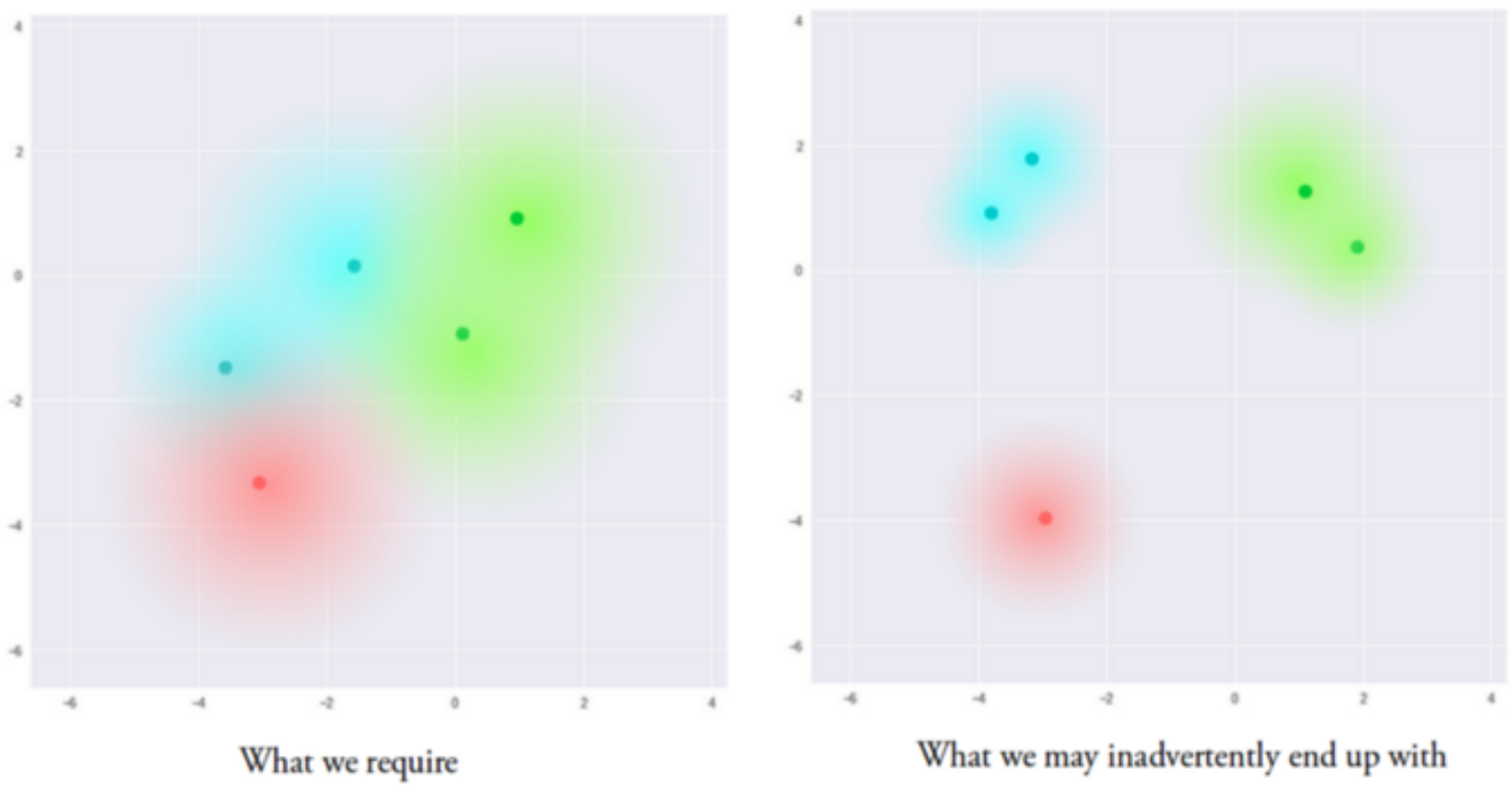


image source

ПОЛУЧИЛИ ЛИ ТО ЧТО ХОТЕЛИ?

- С точки зрения лосса реконструкци картина вполне возможна: ему выгодно точкам выбрать как можно меньшие дисперсии и раскидать подальше друг от друга

ПОЛУЧИЛИ ЛИ ТО ЧТО ХОТЕЛИ?

- С точки зрения лосса реконструкци картина вполне возможна: ему выгодно точкам выбрать как можно меньшие дисперсии и раскидать подальше друг от друга
- Добавим за это наказание!

ПОЛУЧИЛИ ЛИ ТО ЧТО ХОТЕЛИ?

- С точки зрения лосса реконструкци картина вполне возможна: ему выгодно точкам выбрать как можно меньшие дисперсии и раскидать подальше друг от друга
- Добавим за это наказание!
- Потребуем чтобы каждая компонента латентного пространства вела себя как N(0, 1)

ЛОСС VAE

- Тогда лосс будет складываться из двух слагаемых:
  - Ошибка реконструкции (как у обычного автоэнкодера)
  - Штраф за отклонение от N(0, 1). Добьемся этого расстоянием Кульбака-Лйблера:

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

ЛОСС VAE

- Если бы мы оставили только KL, то картина была бы такой (на MNIST):

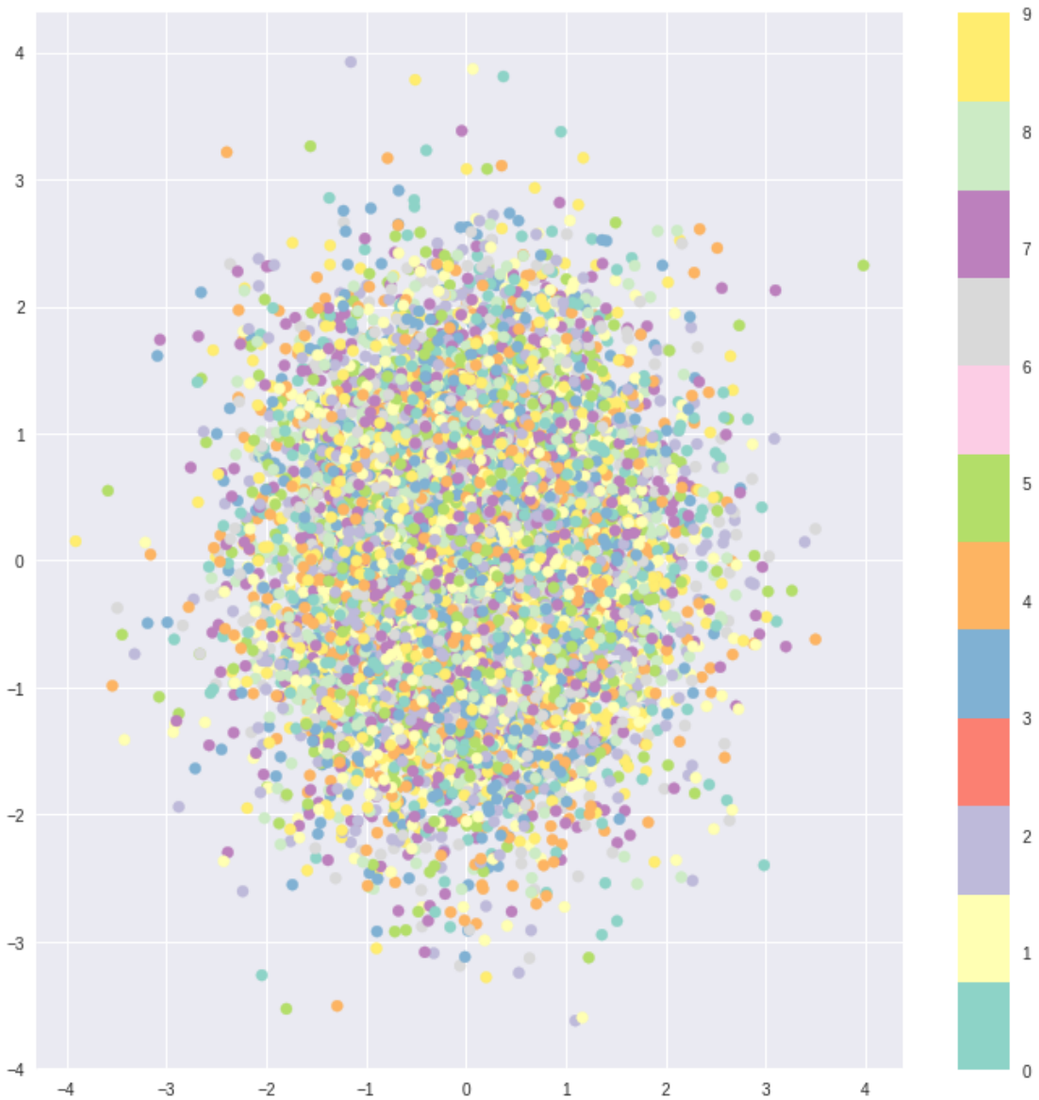


image source

ЛОСС VAE

- А если оба слагаемых, то как мы и хотели:

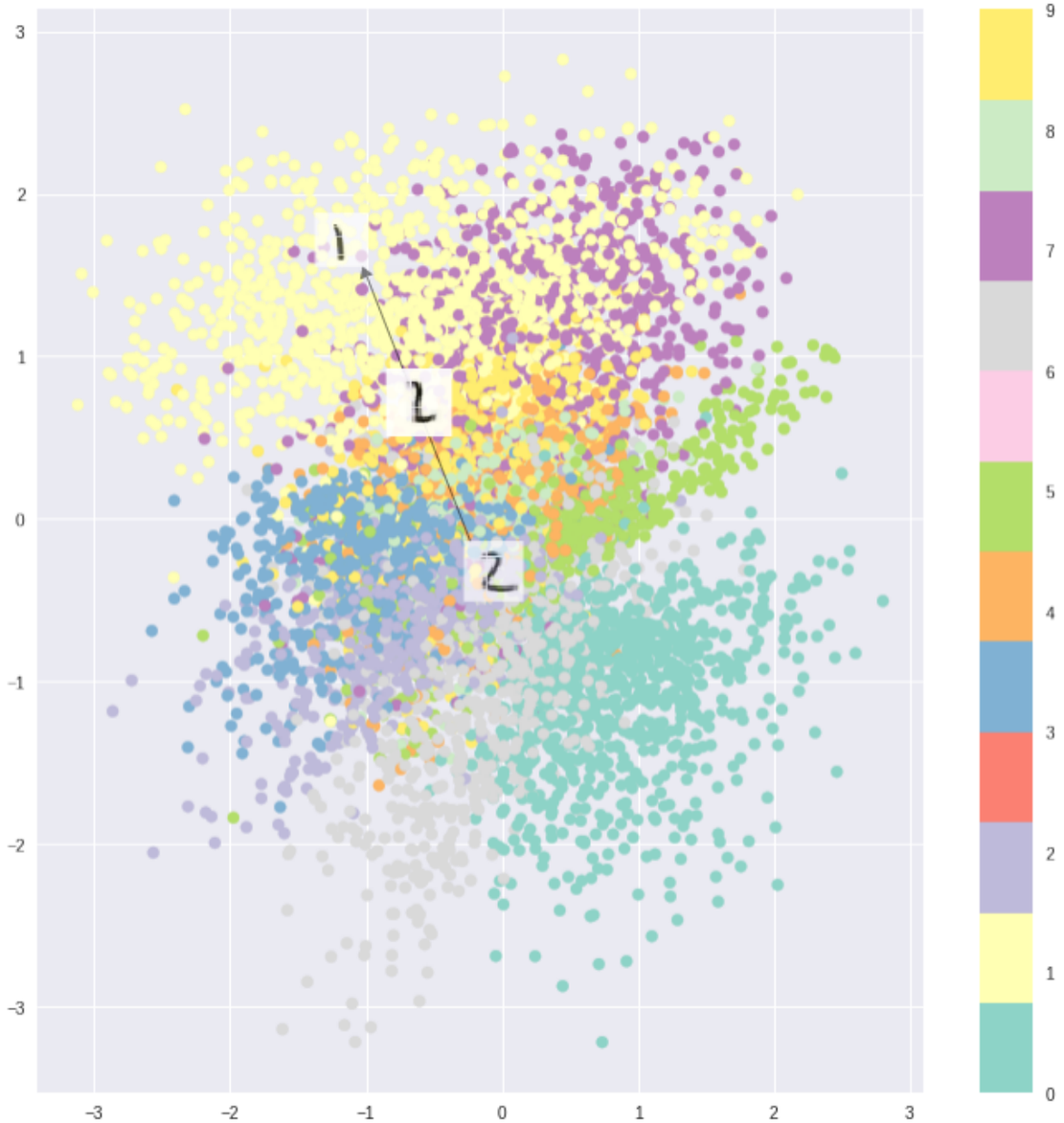


image source

А ЧТО С BACKPROPAGATION?

Мы же не можем пропустить градиент через семплирование

REPARAMETERIZATION TRICK

Будем семплировать N(0, 1) и смещать/нормировать

Тогда это просто как еще одна переменная, которую будем игнорировать во время обратного хода

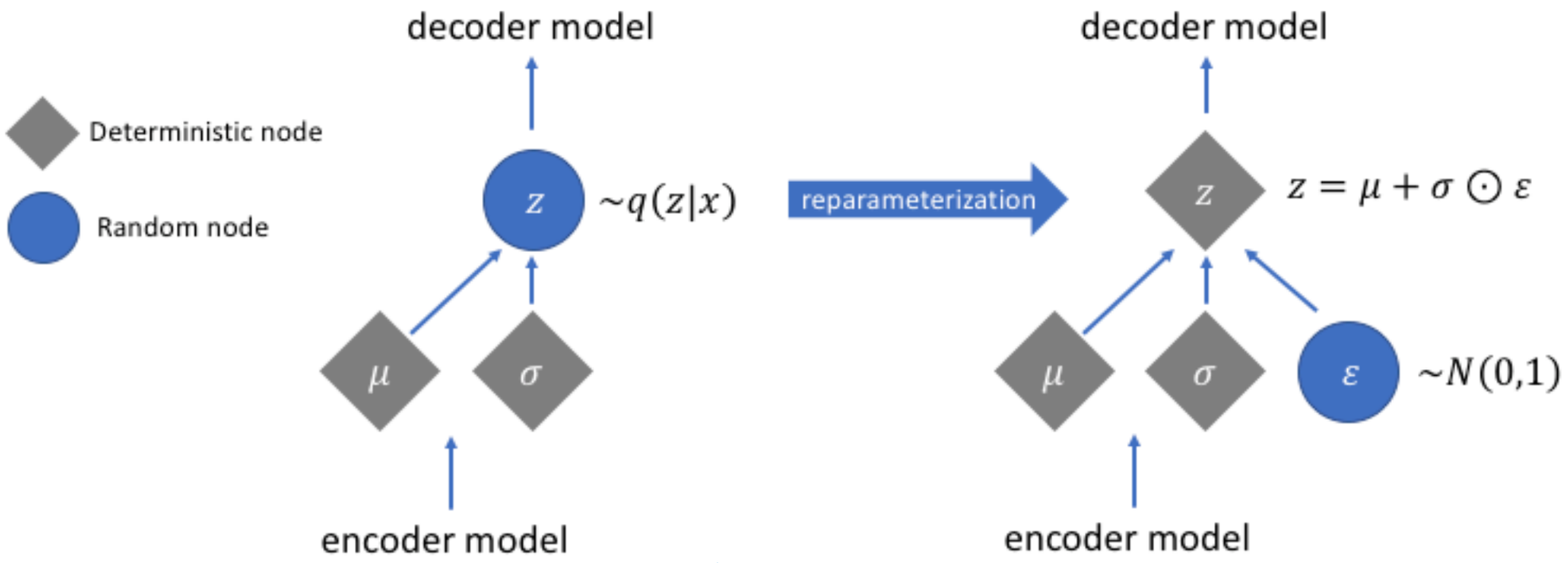


image source

REPARAMETERIZATION TRICK

Будем семплировать N(0, 1) и смещать/нормировать

Тогда это просто как еще одна переменная, которую будем игнорировать во время обратного хода

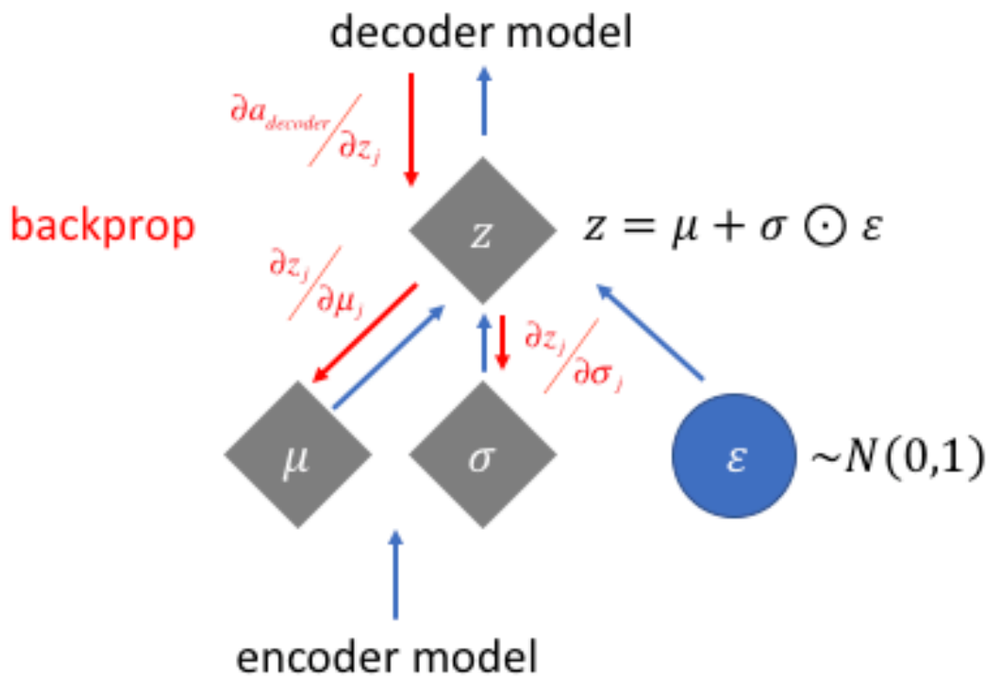


image source

ПРИЛОЖЕНИЯ VAE

- Исправление дефектов изображений
- Дорисовка очков, усов и т.д.
- Создание фейковых изображений/текстов/любых данных

РАССМОТРИМ АЛЬТЕРНАТИВНЫЙ ПОДХОД К ГЕНЕРАЦИИ

- Будем использовать две сети:
  - Генератор создает новые семплы
  - Дискриминатор пытается классифицировать реальные семплы от фиктивных (сгенерированных)
- Сети будут обучаться одновременно, состязаясь друг с другом
- Обучение окончено если дискриминатор ошибается в половине случаев

РАССМОТРИМ АЛЬТЕРНАТИВНЫЙ ПОДХОД К ГЕНЕРАЦИИ

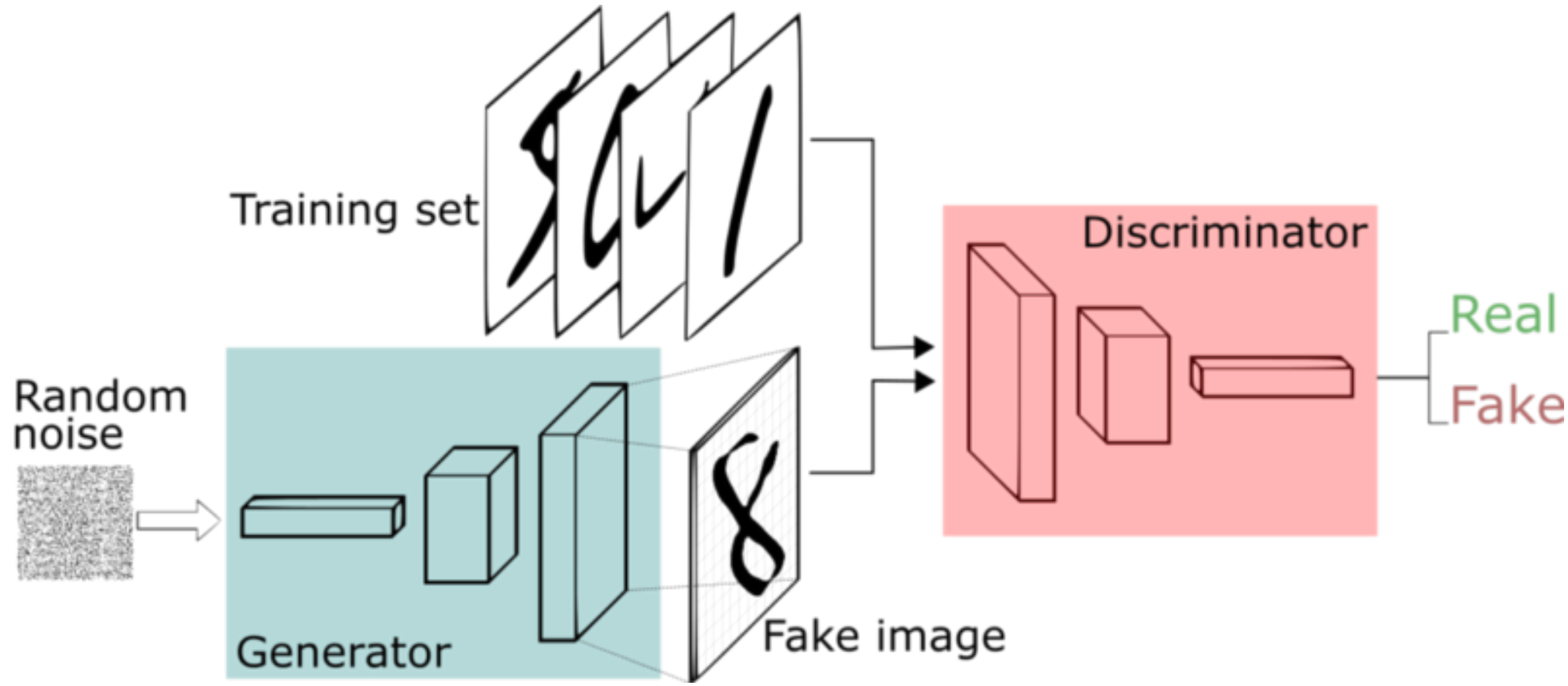


image source

ГЕНЕРАТОР

- Генератор берет на вход случайный вектор, на выходе дает семпл
- Чаще всего GAN применяют к изображениям, поэтому типичный генератор состоит из нескольких сверточных слоев с апсемплингом
- Лоссом будет cross entropy от ошибок дискриминатора
- Можно его "предобучить" как автоэнкодер
- После обучения GAN дискриминатор можно отбросить и просто использовать генератор чтобы генерить семплы

ДИСКРИМИНАТОР

- Типичный дискриминатор будет состоять из сверточных слоев и плотного слоя на конце с одним сигмоидальным выходом
- Он будет получать батч реальных изображений и батч сгенерированных. Лосс дискриминатора будет суммой из cross entropy лосса на реальных и такого же лосса не фейковых изображениях

ЗАМЕЧАНИЯ

- За этим всем стоит своя математика с формулами, но уже разбирать не будем
- Обучать их реально сложно
- Применения - генерация качественных изображений, перенос стиля, смена времен года и т.д.

А ЧТО С РОБОТИКОЙ?

- Как и в любых других областях генеративные модели могут быть использованы для расширения датасета
- Или для выделения фичей, которые идут на вход другим моделям (в частности, RL)

А ЧТО С РОБОТИКОЙ?

- Как и в любых других областях генеративные модели могут быть использованы для расширения датасета
- Или для выделения фичей, которые идут на вход другим моделям (в частности, RL)
- Но есть и более специфичные вещи

REINFORCEMENT LEARNING

- Одна из существенных проблем RL - шаг exploration. Можно долго блуждать в поисках ситуаций, которые влияют на функцию вознаграждения
- В этом могут помочь генеративные модели:

https://arxiv.org/pdf/1803.10122.pdf
https://link.springer.com/chapter/10.1007/978-3-030-47426-3_59