

Введение в нейронные сети

Лекция 2. Углубление в НС и библиотеку Keras



План занятия

Особенности обучения НС

- 1 Основные виды функций потерь для различных задач глубокого обучения
- 2 Алгоритмы оптимизации: SGD, momentum GD, RMSProp, Adam
- 3 Проблема переобучения
- 4 Алгоритмы регуляризации (Dropout, Batchnorm, L1, L2-regularization)
- 5 Немного о метриках

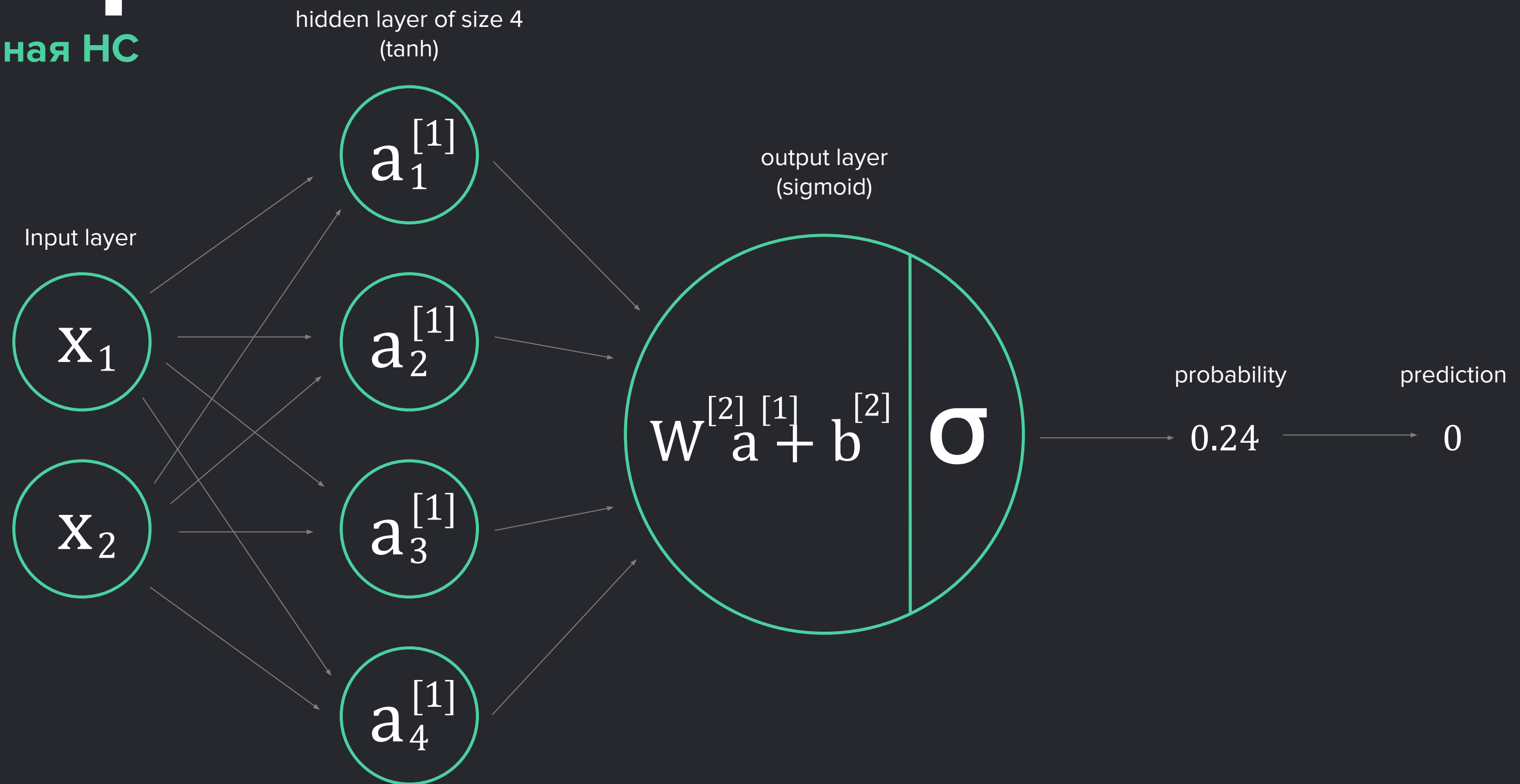
Семинар про Keras

- 1 Keras functional API
- 2 Сохранение модели на диск
- 3 Keras Callbacks



Ресар

Однослойная НС

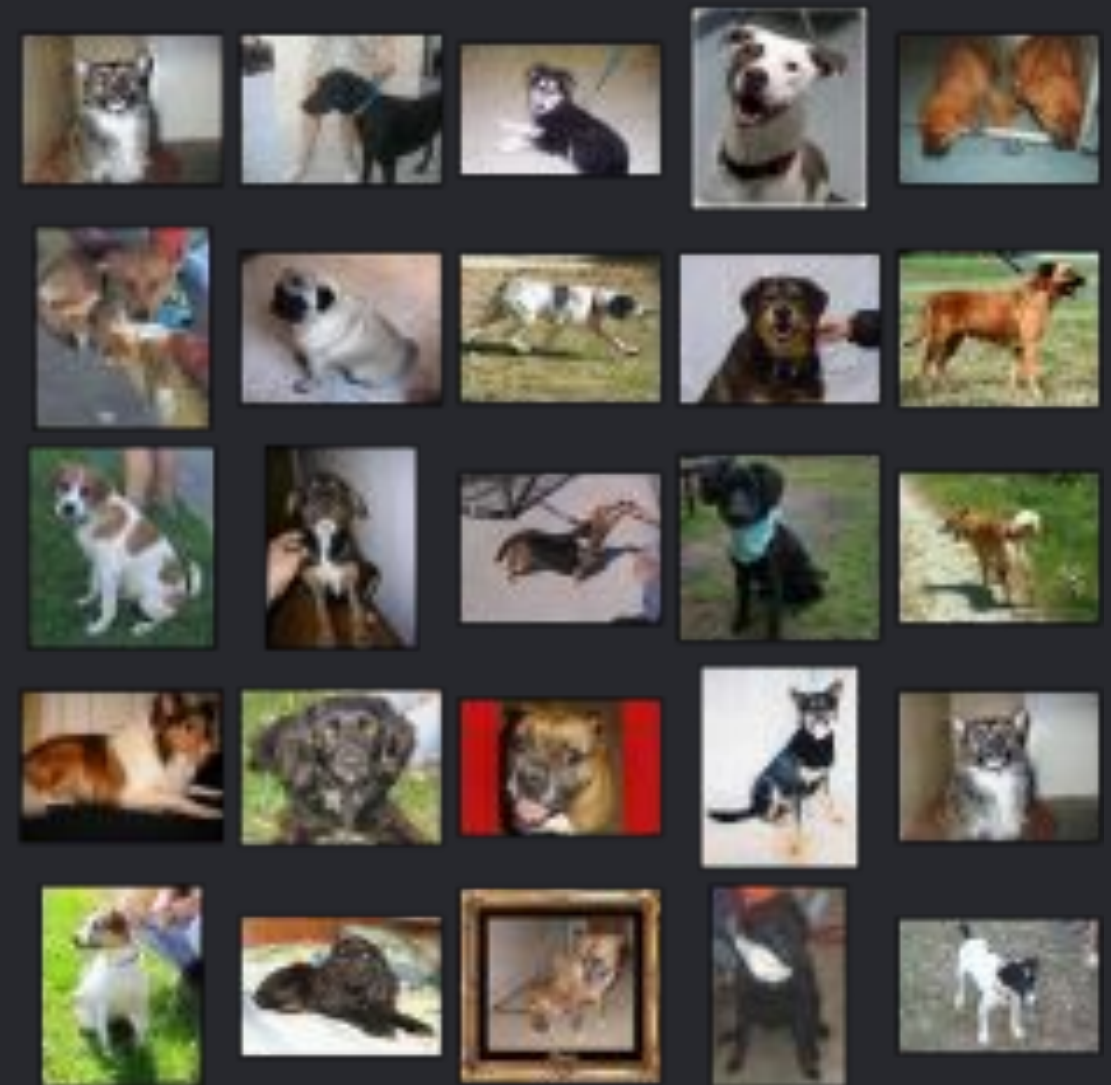


Loss functions

Classification Regression



Dogs vs. Cats dataset Regression.



Loss functions

Classification

Binary cross-entropy:

$$\text{BCE}(y, \hat{y}) = -(y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

Categorical cross-entropy

KL-divergence:

$$\mathcal{L}(y, \hat{y}) = y \cdot \log\left(\frac{y}{\hat{y}}\right)$$



Loss functions

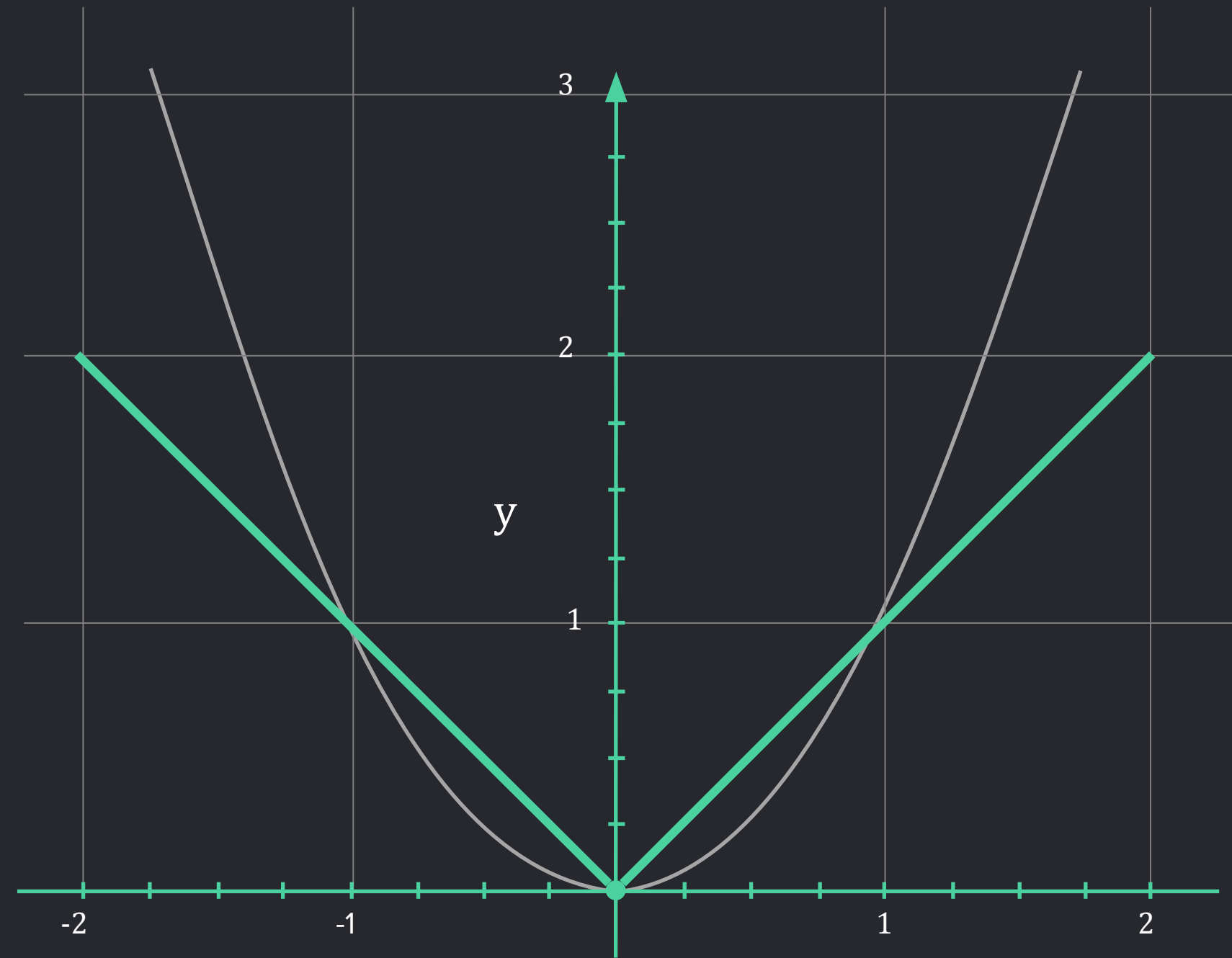
Regression

MSE

- MAE
- Huber loss

Huber loss — своего рода
«компромисс» между MSE и MAE

MSE vs MAE



Summary

- Выбор loss function определяется постановкой задачи и не зависит (почти всегда) от архитектуры НС
- Градиент в НС считается от loss function по параметрам сети — по весам и смещению
- Loss function не всегда можно хорошо интерпретировать, поэтому необходимо контролировать метрики

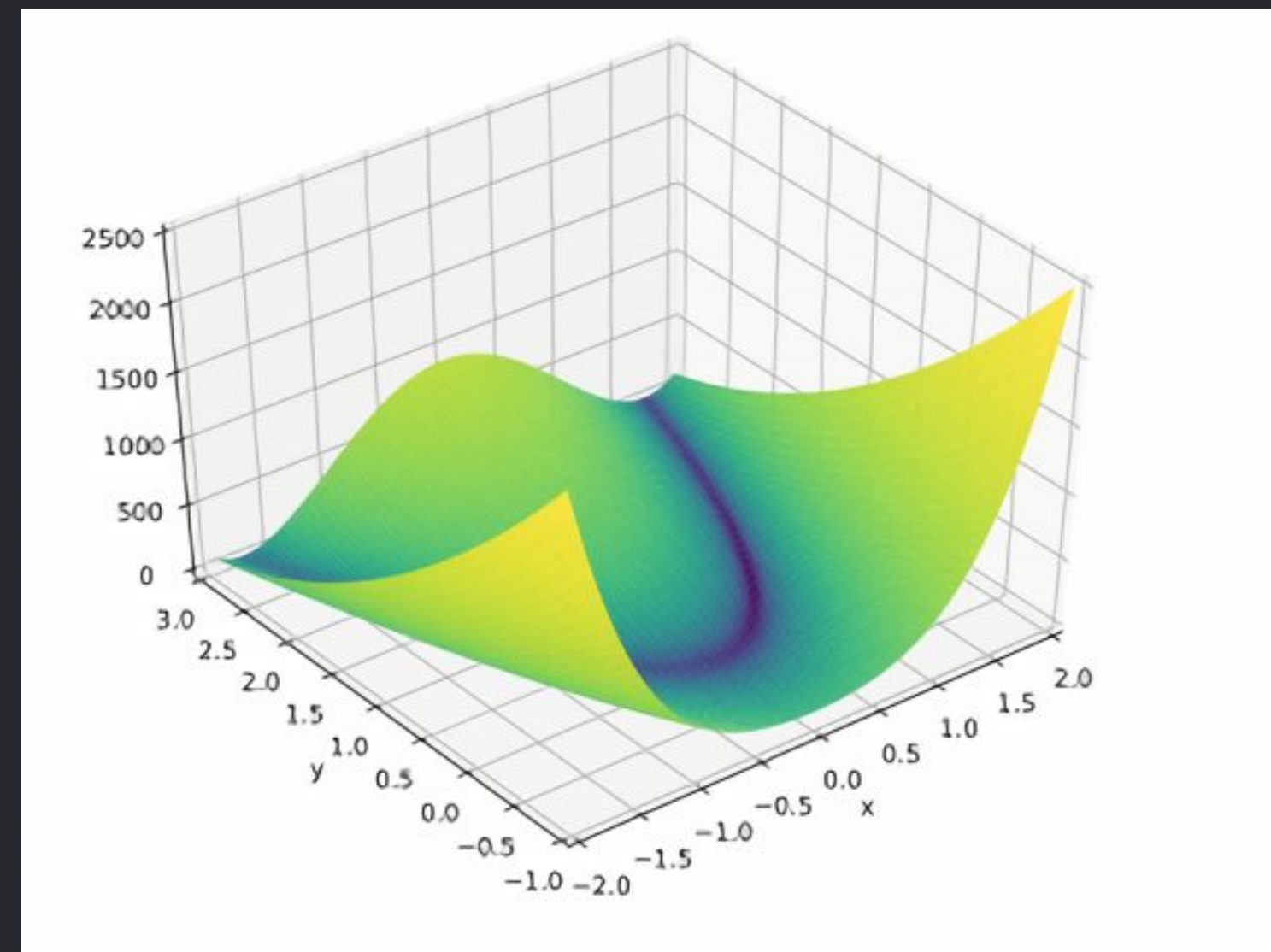


Введение в ОПТИМИЗАЦИЮ в НС

Функция Розенброка — невыпуклая функция с одним глобальным минимумом, которая используется для оценки качества алгоритмов оптимизации.

В НС при оптимизации функции потерь существует две основные проблемы:

- седловые точки
- разные масштабы данных



Mini-batch gradient descent



Часто при обучении НС мы имеем дело с огромными выборками данных (10–100 ГБ), поэтому не представляется возможным загрузить их в память компьютера.

Для решения этой проблемы в современном DL имеют дело с mini-batch gradient descent: градиент подсчитывается на n объектах выборки, после чего обновляются параметры и выбираются следующие n объектов.

Такая подвыборка называется **batch**, а алгоритм такой оптимизации **mini-batch gradient descent**



Скользящее среднее

Скользящее среднее

(exponential moving average) — метод аппроксимации среднего арифметического.

Если задана последовательность O , тогда скользящее среднее v можно определить с помощью формул справа

$$v_0 = 0$$

$$v_1 = 0.9 \cdot v_0 + 0.1 \cdot O_1$$

$$v_2 = 0.9 \cdot v_1 + 0.1 \cdot O_2$$

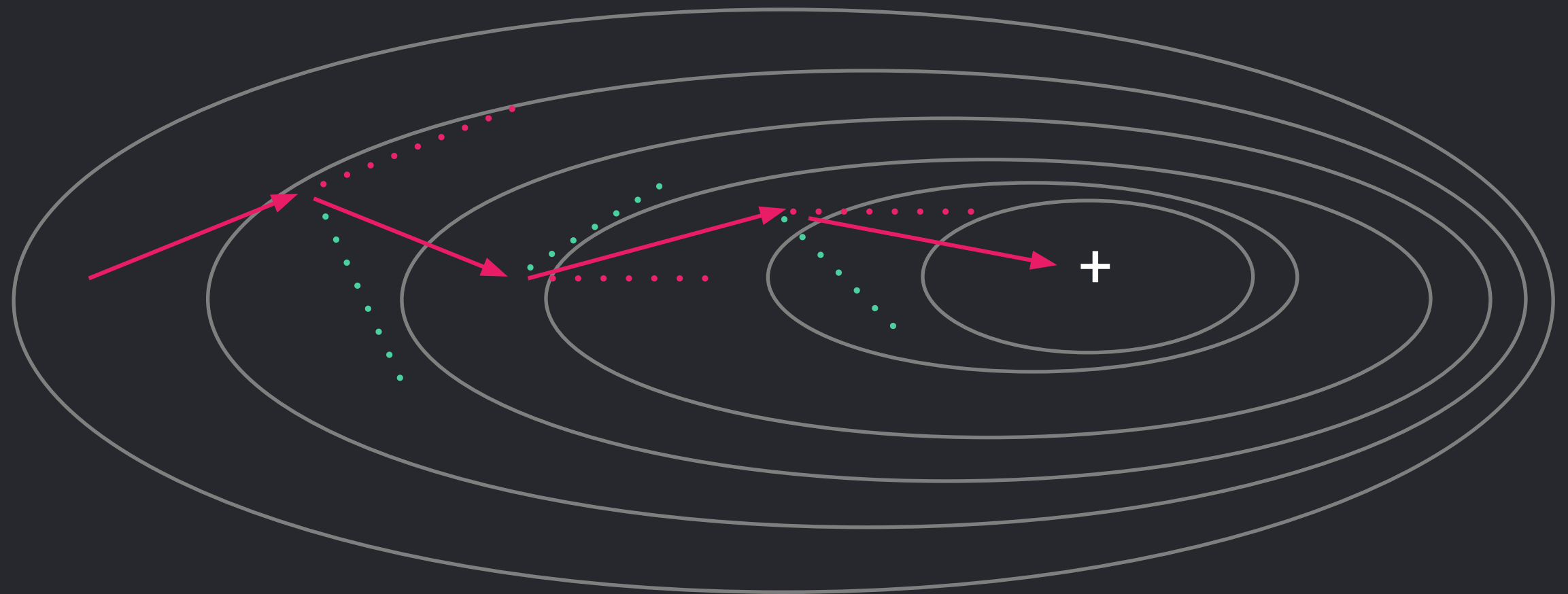
$$v_t = \beta \cdot v_{t-1} + (1 - \beta) \cdot O_t$$



Momentum GD



Идея алгоритма состоит в том, чтобы считать скользящие средние для градиентов и использовать уже эти векторы, а не сами градиенты для обновления параметров



RMSProp

Алгоритм RMSProp эффективно оптимизирует функции, в которых есть серьезное различие дисперсий значений по разным осям.

$$s_{dw} = \beta \cdot s_{dw} + (1 - \beta) \cdot (dw)^2$$

$$s_{db} = \beta \cdot s_{db} + (1 - \beta) \cdot (db)^2$$

$$w = w - \frac{a}{\sqrt{s_{dw}}} \cdot \frac{\partial \mathcal{L}}{\partial w}$$



AdaM

AdaM — комбинация RMSProp и Momentum GD.

Рассмотрим его реализацию:

1

$$v_{dw} = \beta_1 \cdot v_{dw} + (1 - \beta_1) \cdot dw$$

$$v_{db} = \beta_1 \cdot v_{db} + (1 - \beta_1) \cdot db$$

$$S_{dw} = \beta_2 \cdot S_{dw} + (1 - \beta_2) \cdot (dw)^2$$

$$S_{db} = \beta_2 \cdot S_{db} + (1 - \beta_2) \cdot (db)^2$$

2

$$V_{dw}^{correct} = \frac{V_{dw}}{1 - \beta_1^t}$$

$$V_{db}^{correct} = \frac{V_{db}}{1 - \beta_1^t}$$

$$S_{dw} = \frac{S_{dw}}{1 - \beta_2^t}$$

$$S_{db} = \frac{S_{db}}{1 - \beta_2^t}$$

3

Коррекция смещения
Обновление параметров

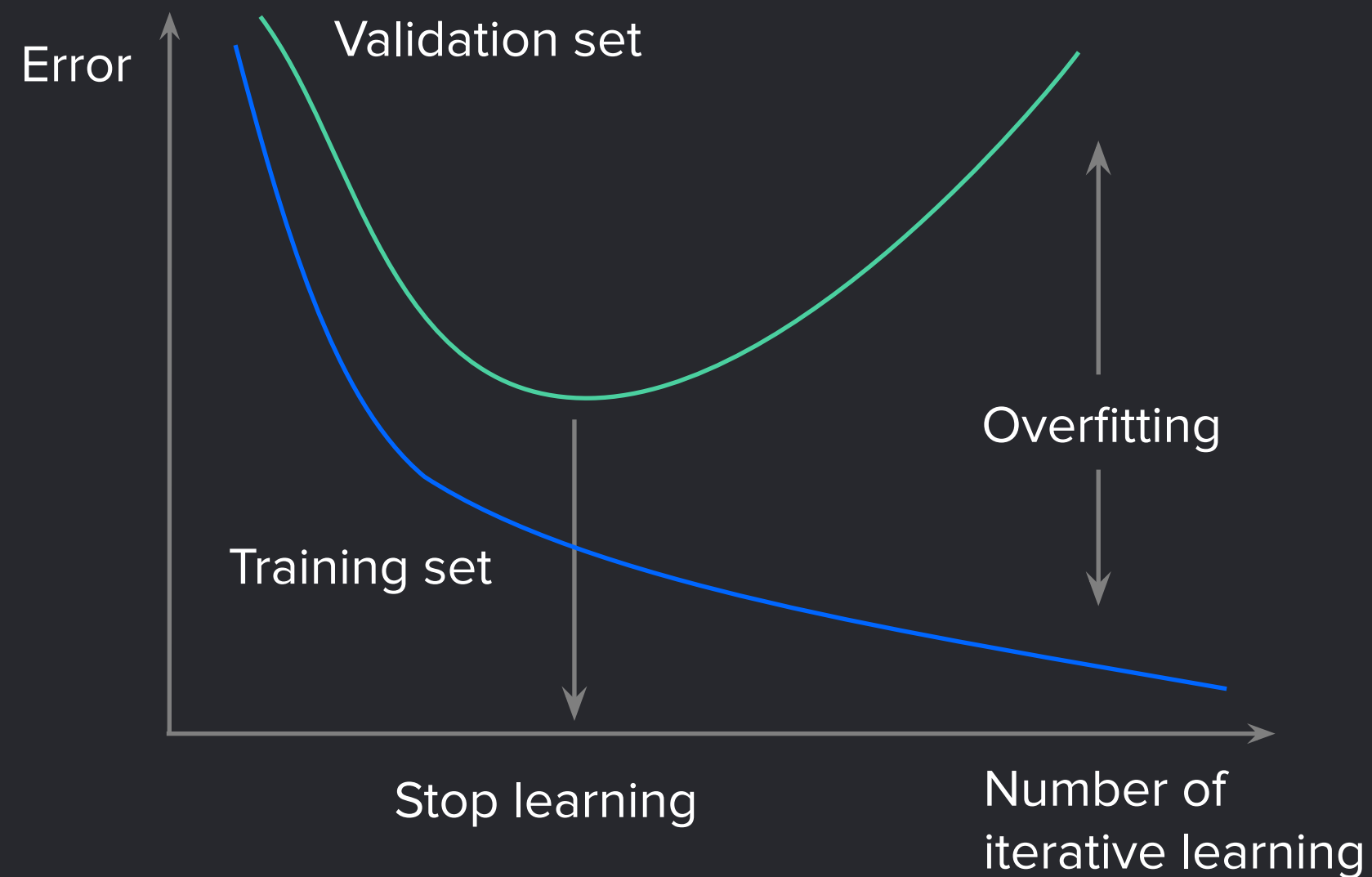
$$w = w - a \frac{V_{dw}^{correct}}{\sqrt{S_{dw}^{correct} + \epsilon}}$$

$$b = b - a \frac{V_{db}^{correct}}{\sqrt{S_{db}^{correct} + \epsilon}}$$



Overfitting

Переобучение или overfitting — это чрезмерное подстраивание под данные обучающей выборки, при котором ухудшается качество работы модели



Регуляризация

Регуляризация — контролируемое ухудшение модели, которое должно предотвращать переобучение

L2 — регуляризация:

$$J(y, \hat{y}) = \sum \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) + \frac{\lambda}{2} \sum_l \|w^{[l]}\|_F^2$$

L1 — регуляризация:

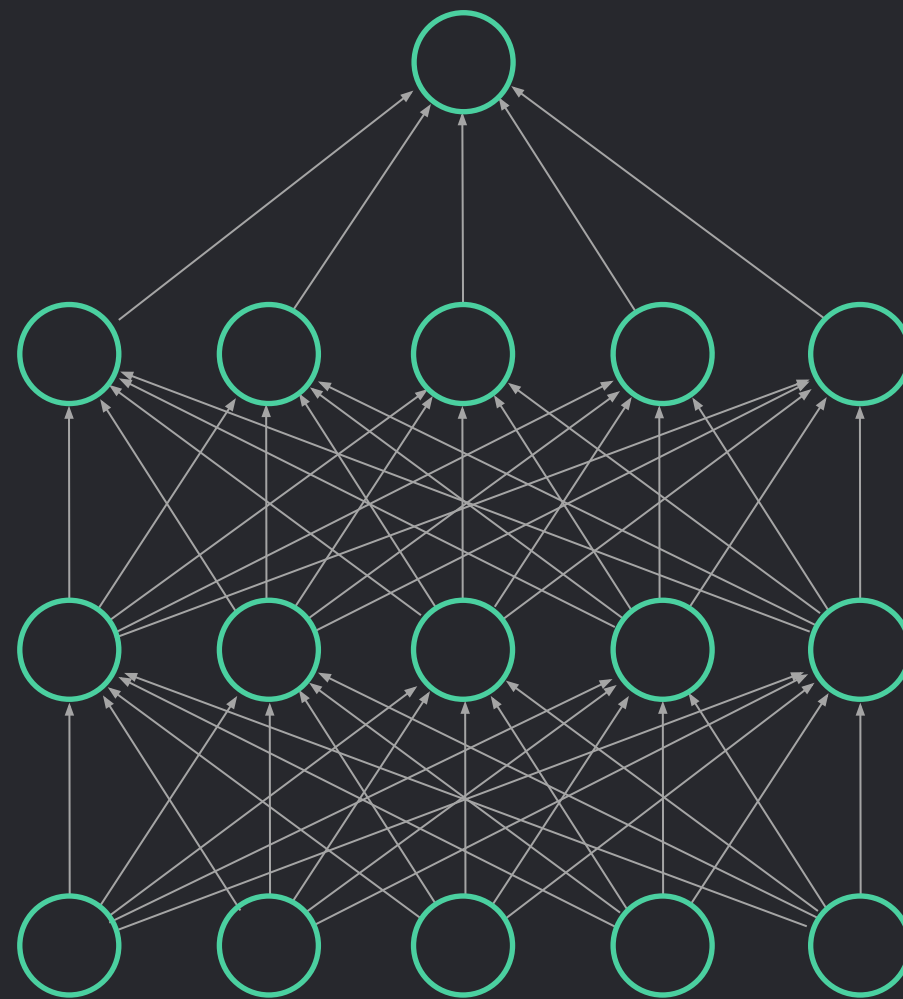
$$J(y, \hat{y}) = \sum \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) + \lambda \sum_l \sum_{i,j} \|w_i^{[l]j}\|_F$$



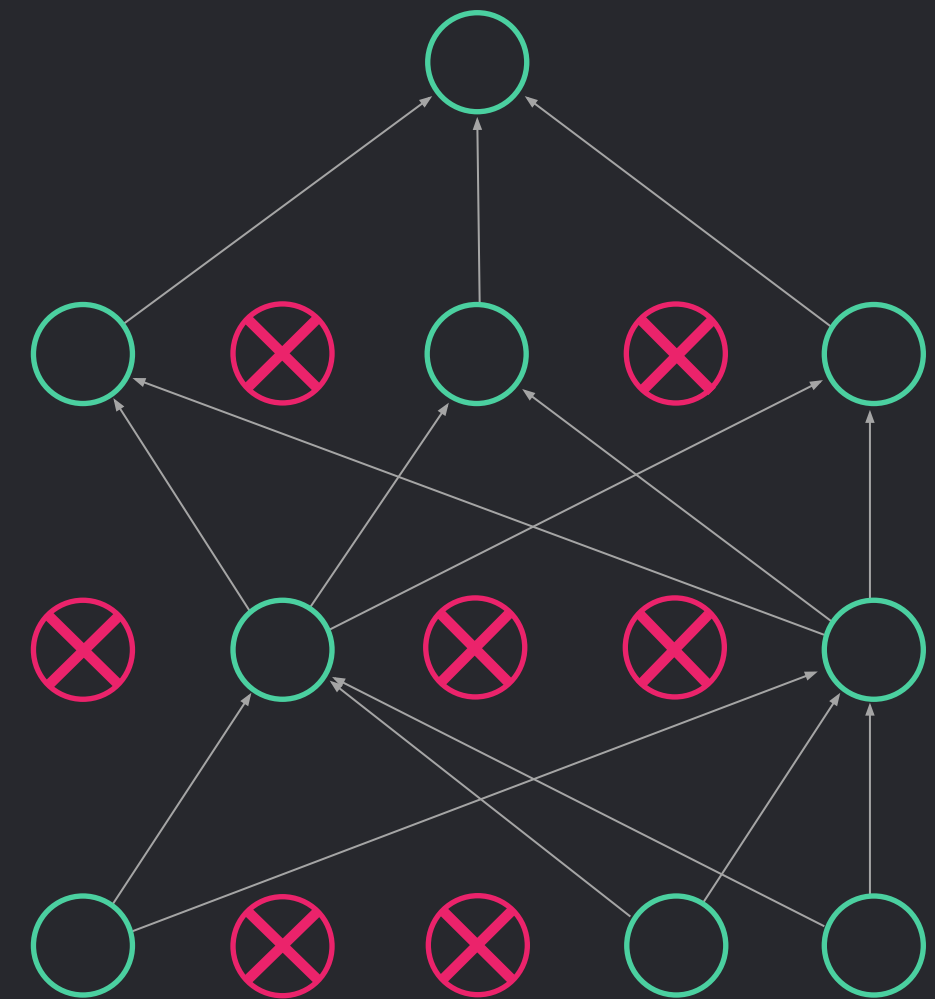
Регуляризация. Dropout

Dropout — техника случайного обнуления заданного количества весов для увеличения стабильности работы нейросети

Dropout применяется только на стадии обучения модели.
Если модель уже обучена, Dropout применять не нужно



(a) Standard Neural Net



(b) After applying dropout



Batch normalization

Batch normalization — это алгоритм, предназначенный для борьбы с затуханием/взрывом градиентов.

Также он ускоряет процесс обучения.

Идея заключается в нормировке данных на каждом слое, чтобы их средние и стандартные отклонения стали равны 0 и 1 соответственно

При таком подходе этот слой содержит как обучаемые параметры, так и те, которые считаются с помощью усреднения по всей выборке

$$\mu^{[l]} = \frac{1}{m} \sum_{i=1}^m z^{[l]}(i)$$

$$\sigma^{[l]} = \frac{1}{m} \sum_{i=1}^m (z^{[l]}(i) - \mu^{[l]})$$

$$z_{norm}^{[l]} = \frac{z^{[l]} - \mu^{[l]}}{\sqrt{\sigma^2 + \varepsilon}}$$

$$\hat{z}^{[l]} = \gamma^{[l]} z_{norm}^{[l]} + \beta^{[l]}$$



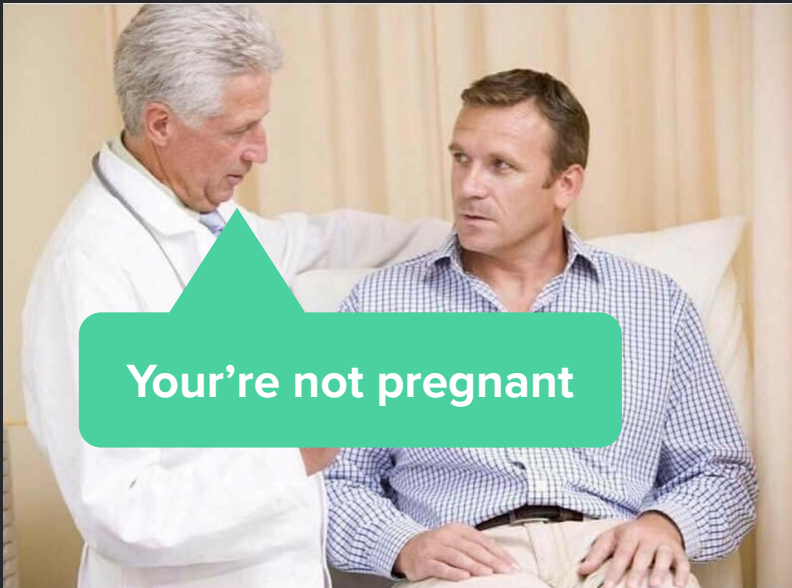


P. S. Немного о метриках

Метрики качества, в отличие от функции потерь, в обучении не используются, рассчитываются исключительно для наглядности и удобства. Но именно они показывают, насколько хорошо алгоритм справляется с поставленной задачей

Важнейшие метрики задачи классификации:

- accuracy
- precision
- recall

		Actual Values	
		1	0
Predicted Values	1	True positive 	False positive 
	0	False negative 	True negative 



Метрики

$$acc = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\mathcal{P} = \frac{TP}{TP + FP}$$

$$\mathcal{R} = \frac{TP}{TP + FN}$$



**Спасибо
за внимание!**

