Лекция 1. Введение

Денис Деркач, Дмитрий Тарасов

LAMBDA • HSE

19 января 2026 года

Параметры курса

Лекции

Денис Деркач

- Магистратура (СПбГУ), 2007, PhD (Париж 11), 2010.
- Заведующий лабораторией методов анализа больших данных (LAMBDA)

Интересы

- Генеративное моделирование физических процессов с помощью машинного обучения.
- Оптимизация и настройка инженерного оборудования с помощью ML.
- Объединение DL и классических моделей.

Идея курса

DL – довольно размытый термин, потому зафиксируем некоторые детали.

К концу курса вы будете:

- Понимать базовые принципы работы глубокого обучения
- Использовать на практике полученные знания.
- Выбирать согласно данным основные архитектуры нейронных сетей - свёрточные, рекуррентные, трансформеры
- Использовать в основных приложения - Computer Vision, NLP
- Применять базовые архитектуры генеративных моделей в стандартных постановках задачи.

Организационные вопросы

- Материалы курса: http://github.com/fintech-dl-hse/course
- Формула оценки: $0.8 \cdot \sum_i O_{\text{hw}_\text{i}} + 0.1 \cdot O_{\text{exam3}} + 0.1 \cdot O_{\text{exam4}}$
- После занятия в боте будет открываться квиз с открытыми вопросами по темам занятия. Вопросы несложные. Прохождение квизов будет допуском к экзаменам отдельно и за 3 и за 4 модуль.
- Telegram чат
- Форма обратной связи после занятий
- Задавайте вопросы!

Литература

- Глубокое обучение. Погружение в мир нейронных сетей. С. Николенко
- А. Кадурин, Е. Архангельская. • Глубокое обучение.
- Я. Гудфеллоу, Й. Бенджио, А. Курвилль. Английская версия: http://www.deeplearningbook.org/.
- Understanding deep learning. S. Prince http://udlbook.github.io/udlbook/
- Dive into Deep Learning. A. Zhang, Z. Lipton, M. Li, A. Smola http://d2l.ai/

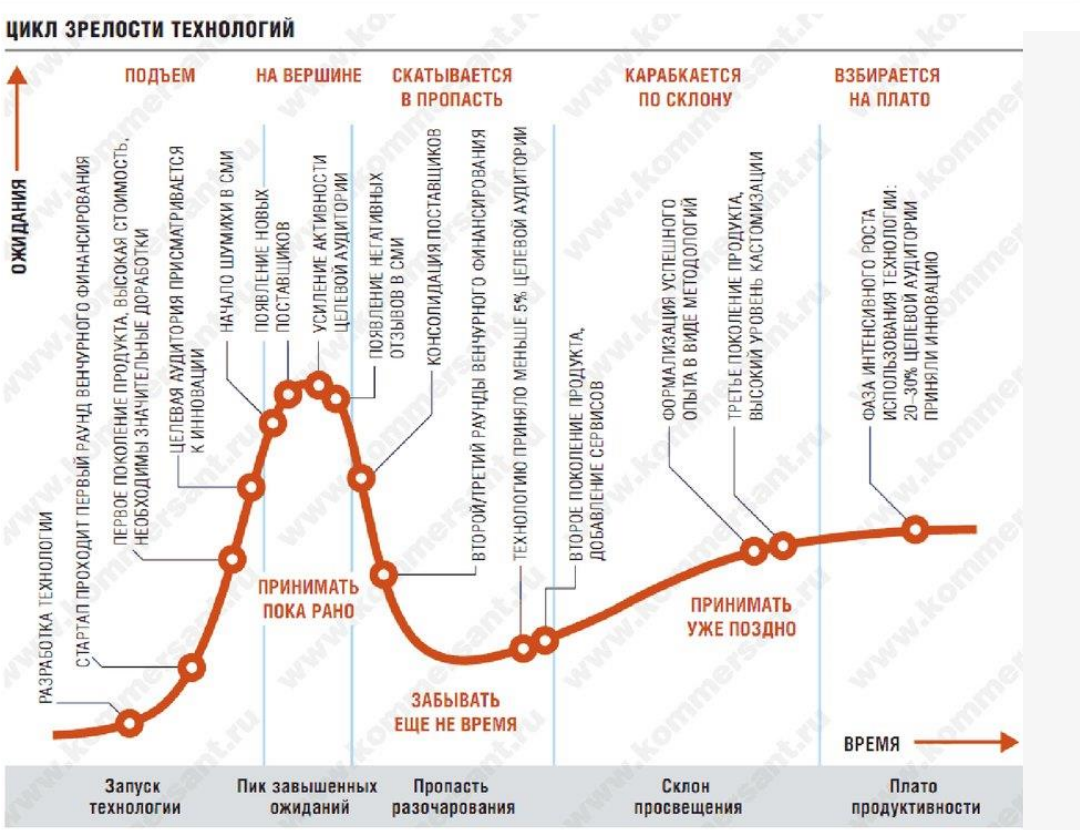
Дополнительные источники

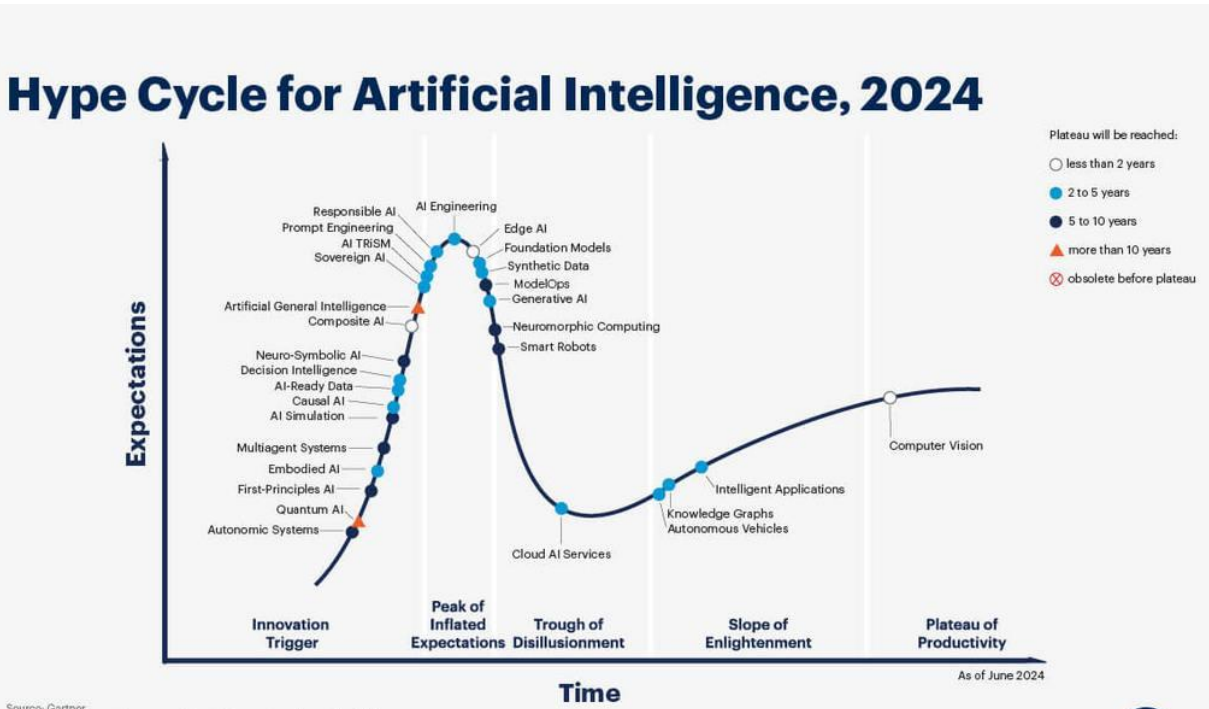
- CS231n - DL for CV (Stanford U.)
- CS224n - NLP with DL (Stanford U.)
- http://github.com/yandexdataschool/Practical_DL
- http://github.com/yandexdataschool/nlp_course
- http://dlcourse.ai/
- Pytorch Tutorials
- http://kaggle.com

Глубокое* обучение**

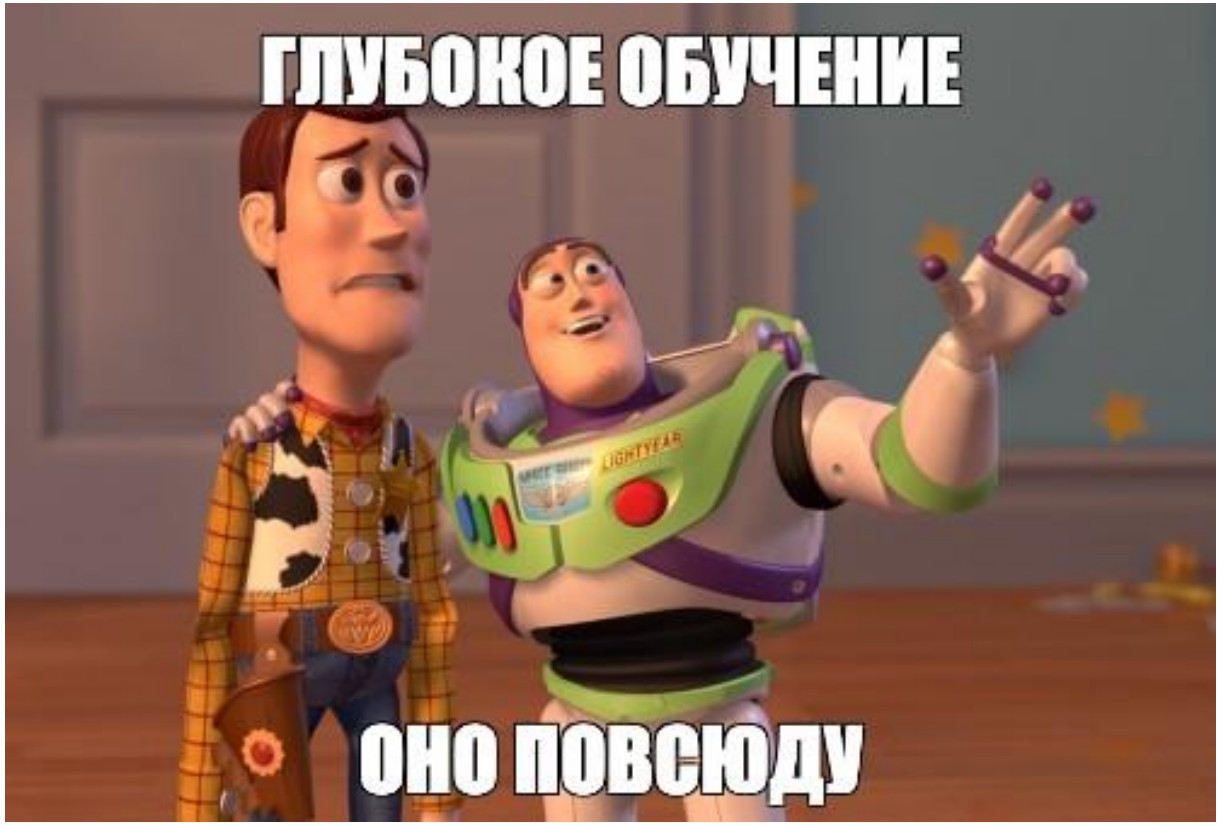
*глубинное
** deep learning, DL

Хайп DL


ЦИКЛ ЗРЕЛОСТИ ТЕХНОЛОГИЙ
ОЖИДАНИЯ
ПОДЪЕМ
НА ВЕРШИНЕ
СКАТЫВАЕТСЯ В ПРОПАСТЬ
КАРАБКАЕТСЯ ПО СКЛОНУ
ВЗБИРАЕТСЯ НА ПЛАТО
РАЗРАБОТКА ТЕХНОЛОГИИ
СТАРТАП ПРОХОДИТ ПЕРВЫЙ РАУНД ВЕНЧУРНОГО ФИНАНСИРОВАНИЯ
ПЕРВОЕ ПОКОЛЕНИЕ ПРОДУКТА. ВЫСОКАЯ СТОИМОСТЬ, НЕОБХОДИМЫ ЗНАЧИТЕЛЬНЫЕ ДОРАБОТКИ
ЦЕЛЕВАЯ АУДИТОРИЯ ПРИСМАТРИВАЕТСЯ К ИННОВАЦИИ
НАЧАЛО ШУМИХИ В СМИ
ПОЯВЛЕНИЕ НОВЫХ ПОСТАВЩИКОВ
УСИЛЕНИЕ АКТИВНОСТИ ЦЕЛЕВОЙ АУДИТОРИИ
ПОЯВЛЕНИЕ НЕГАТИВНЫХ ОТЗЫВОВ В СМИ
КОНСОЛИДАЦИЯ ПОСТАВЩИКОВ
ВТОРОЙ/ТРЕТИЙ РАУНДЫ ВЕНЧУРНОГО ФИНАНСИРОВАНИЯ
ТЕХНОЛОГИЮ ПРИНЯЛО МЕНЬШЕ 5% ЦЕЛЕВОЙ АУДИТОРИИ
ВТОРОЕ ПОКОЛЕНИЕ ПРОДУКТА, ДОБАВЛЕНИЕ СЕРВИСОВ
ФОРМАЛИЗАЦИЯ УСПЕШНОГО ОПЫТА В ВИДЕ МЕТОДОЛОГИЙ
ТРЕТЬЕ ПОКОЛЕНИЕ ПРОДУКТА, ВЫСОКИЙ УРОВЕНЬ КАСТОМИЗАЦИИ
ФАЗА ИНТЕНСИВНОГО РОСТА ИСПОЛЬЗОВАНИЯ ТЕХНОЛОГИИ: 20–30% ЦЕЛЕВОЙ АУДИТОРИИ ПРИНЯЛИ ИННОВАЦИЮ
ПРИНИМАТЬ ПОКА РАНО
ЗАБЫВАТЬ ЕЩЕ НЕ ВРЕМЯ
ПРИНИМАТЬ УЖЕ ПОЗДНО
ВРЕМЯ
Запуск технологии
Пик завышенных ожиданий
Пропасть разочарования
Склон просвещения
Плато продуктивности


Hype Cycle for Artificial Intelligence, 2024
Expectations
Time
Innovation Trigger
Peak of Inflated Expectations
Trough of Disillusionment
Slope of Enlightenment
Plateau of Productivity
Responsible AI
Prompt Engineering
AI TRiSM
Sovereign AI
AI Engineering
Edge AI
Foundation Models
Synthetic Data
ModelOps
Generative AI
Artificial General Intelligence
Composite AI
Neuromorphic Computing
Smart Robots
Neuro-Symbolic AI
Decision Intelligence
AI-Ready Data
Causal AI
AI Simulation
Multiagent Systems
Embodied AI
First-Principles AI
Quantum AI
Autonomic Systems
Cloud AI Services
Computer Vision
Intelligent Applications
Knowledge Graphs
Autonomous Vehicles
Plateau will be reached:
less than 2 years
2 to 5 years
5 to 10 years
more than 10 years
obsolete before plateau
As of June 2024
Source: Gartner
Commercial reuse requires approval from Gartner and must comply with the Gartner Content Compliance Policy on gartner.com.
© 2024 Gartner, Inc. and/or its affiliates. All rights reserved. GTS_3282450
Gartner

Deep Learning



Преимущество

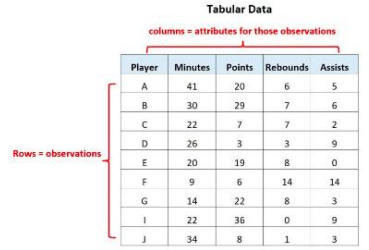
- Гибкость моделирования.
- Следование данным.
- Архитектурные особенности для типа данных (а не для задачи).

Deep Learning

Преимущество

- Гибкость моделирования.
- Следование данным.
- Архитектурные особенности для типа данных (а не для задачи).

Табличные данные - (пока что?) царство градиентного бустинга



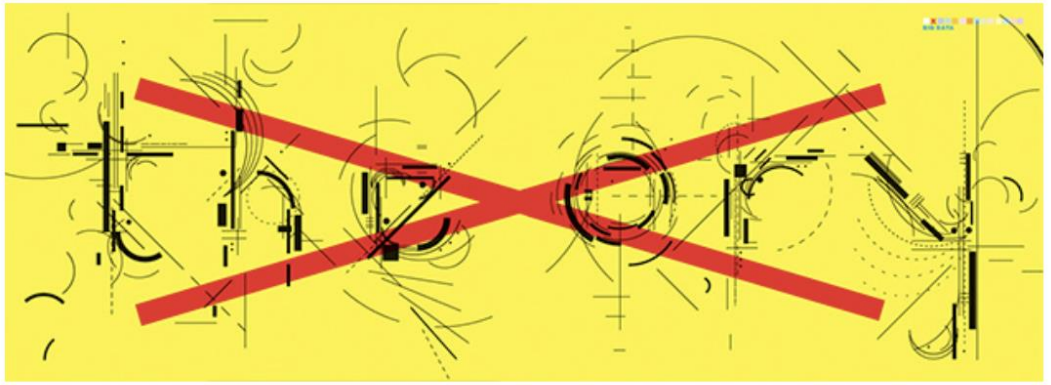
Player	Minutes	Points	Rebounds	Assists
A	41	20	6	5
B	30	29	7	6
C	22	7	7	2
D	26	3	3	9
E	20	19	8	0
F	9	6	14	14
G	14	22	8	3
I	22	36	0	9
J	34	8	1	3

Нейросети хороши в автоматическом выделении признаков, а в табличных данных они уже выделены

Deep Learning

CHRIS ANDERSON SCIENCE 06.23.08 12:00 PM

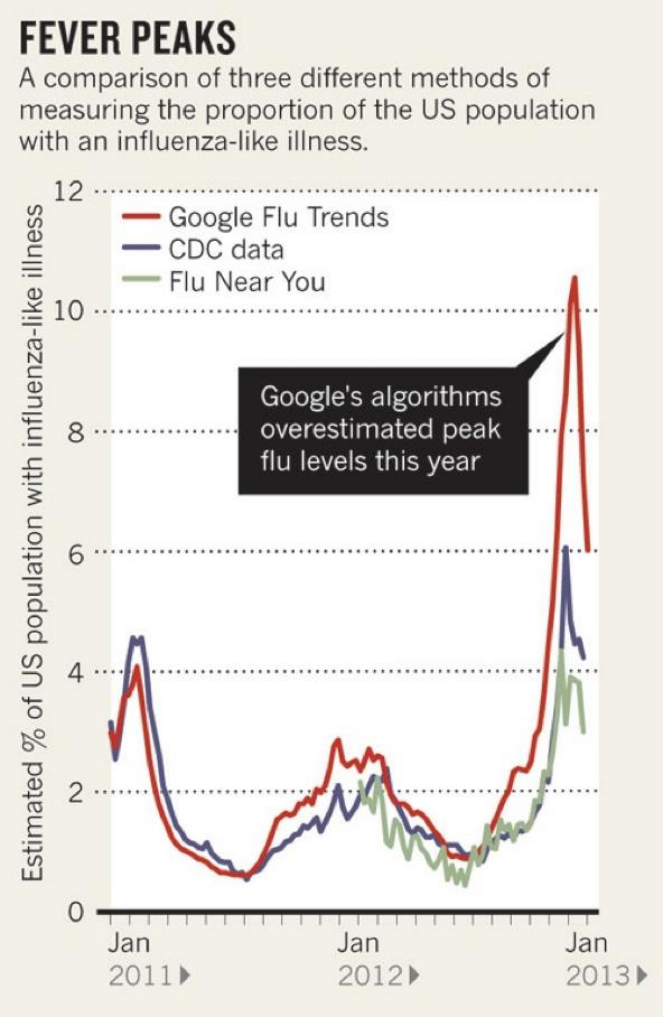
THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE



* Illustration: Marian Bantjes * **"All models are wrong**, but some are useful."

https://www.wired.com/2008/06/pb-theory/

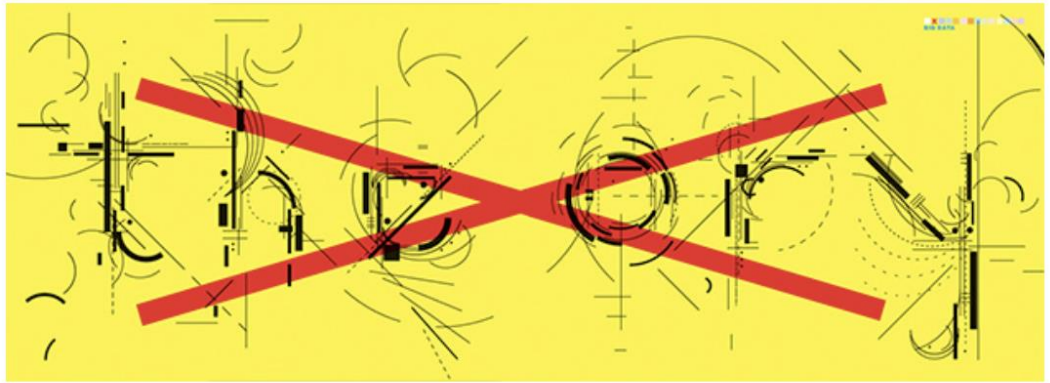
Deep Learning



https://www.nature.com/news/when-google-got-flu-wrong-1.12413

CHRIS ANDERSON SCIENCE 06.23.08 12:00 PM

THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE



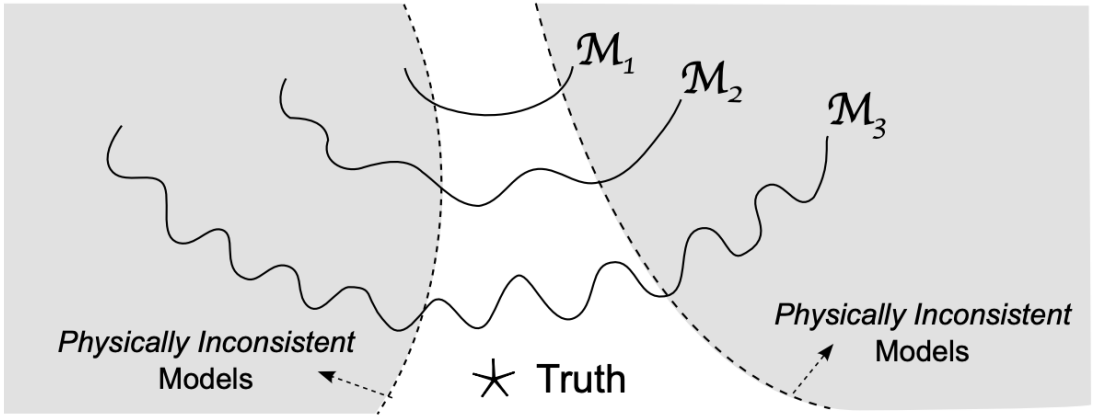
* Illustration: Marian Bantjes * **"All models are wrong**, but some are useful."

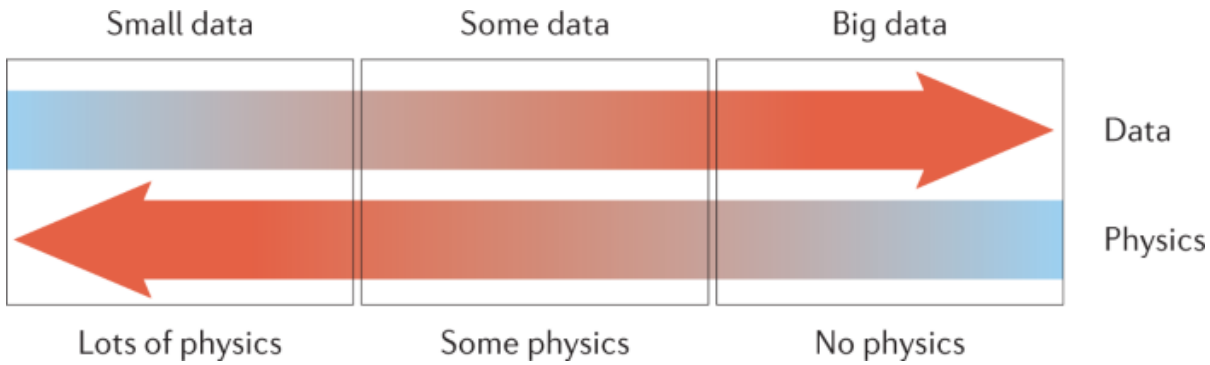
https://www.wired.com/2008/06/pb-theory/

Теория и глубокое обучение

- Робастность
- Экстраполяция.
- Объяснимость.
- Ясность.

Успешное применение теории в местах с маленькими выборками данных (не от хорошей жизни).



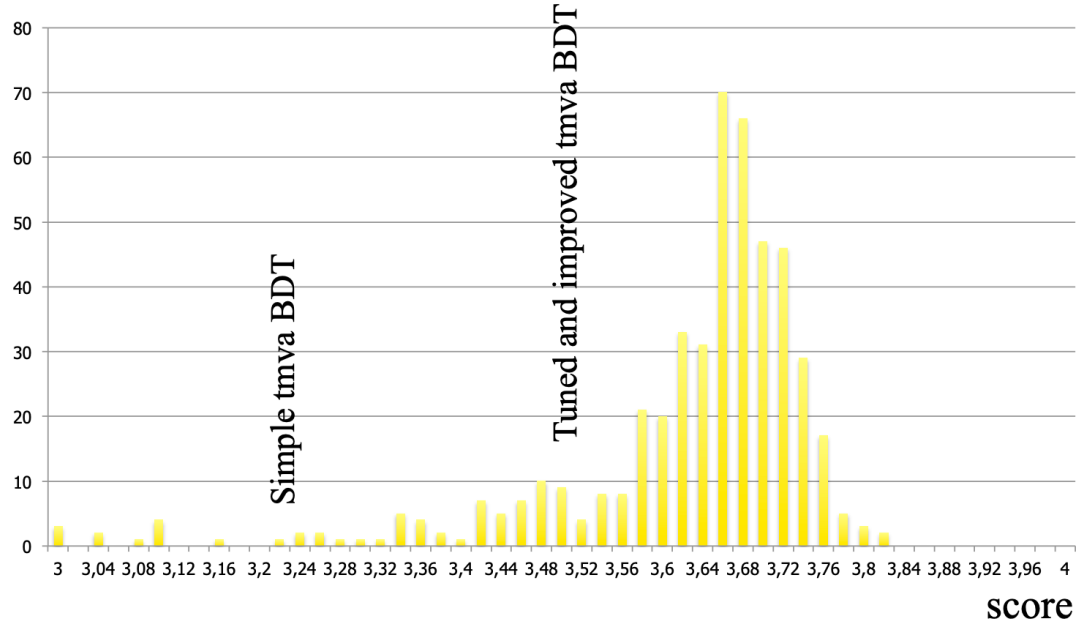


Anuj Karpatne, et al., IEEE Transactions on knowledge and data engineering 29, 10 (2017), 2318–2331

Глубокое обучение и теория

- Использование подхода «черного ящика» в приложениях.
- Быстрая адаптация к новым случаям.
- Анализ многомерных данных.

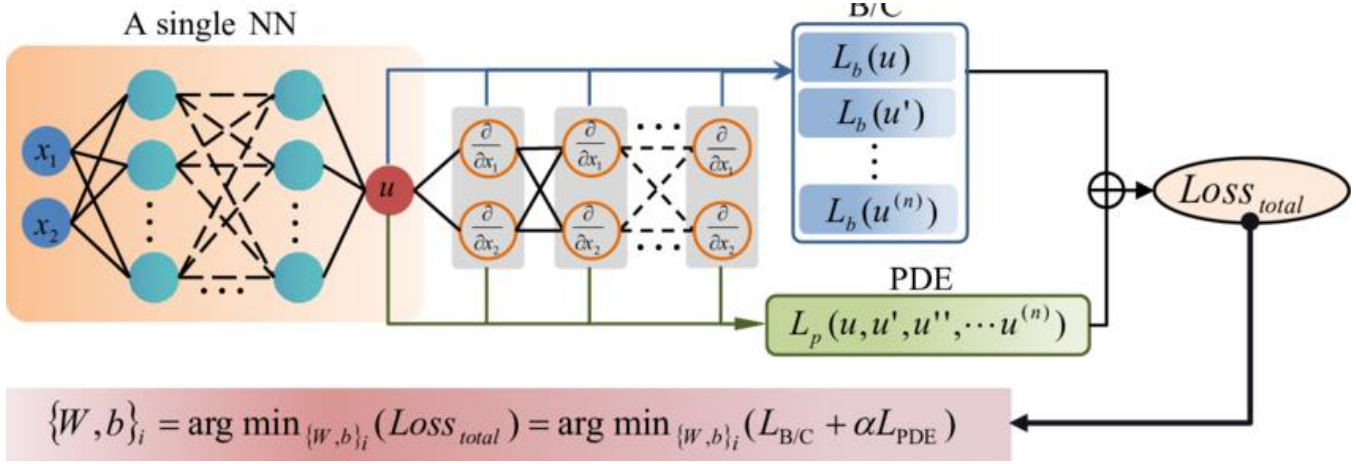
Внедрение методов глубокого обучения позволяет более детально изучить классические методы.



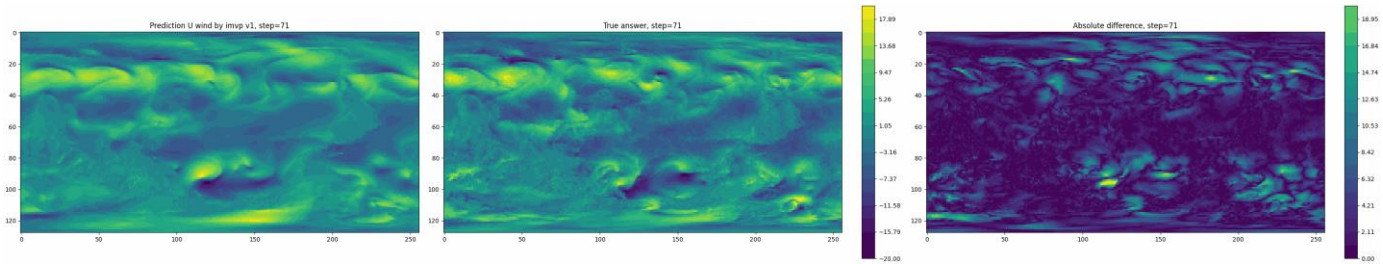
D. Roussau, Connecting the dots 2025
https://indico.physics.lbl.gov/event/149/contributions/219/attachments/211/225/tr150210_davidRousseau_CTD2015_HiggsML.pptx.pdf

Гибридный ИИ

- Выбор смеси теоретической модели и машинного обучения
- Может быть как для скрытых параметров модели, так и для наблюдаемых..



Требует много времени разработчиков. В долгосрочной перспективе - дает хорошие практические решения.



Take-home message

Глубокое обучение проникает во многие сферы человеческой деятельности.

В первую очередь, выигрыш достигается в областях с большим количеством доступных данных и слабыми моделями.

Перспектива – гибридизация глубокого обучения и классических моделей.

История глубокого обучения

Обратное распространение ошибки

1986: Rumelhart, Hinton, Williams - Backpropogation, алгоритм обратного распространения ошибки для обучения нейросетей

Вообще говоря, открывался и переоткрывался несколько раз до этого, в частности Seppo Linnainmaa (1970), Paul Werbos (1982)

- 1980: Fukushima - Неокогнитрон, первая сверточная нейросеть
- 1989: LeCun - CNN для распознавания рукописных цифр (почтовые индексы)
- 1997: Hochreiter, Schmidhuber - LSTM

Глубокое обучение

- 2006: Термин Deep Learning
- 2006: Hinton, Deep Belief Networks - успешное обучение глубоких моделей за счет послойного предобучения без учителя
- 2009: Успех в распознавании речи

Take-home message

Развитие нейронных сетей переживало взлёты и замедления.

Основой развития глубокого обучения стало накопление опыта применения классического машинного обучения и развитие математического аппарата.

К 2010-м годам было накоплено достаточно данных и компьютерных мощностей для получения революционных условий.

Машинное обучение и нейронные сети

Машинное обучение

По сути все, что мы делаем в машинном обучении - это аппроксимация и оптимизация функций:

- есть какая зависимость в реальных данных, очень сложная функция
- мы пытаемся подобрать модель (функцию), которая хорошо аппроксимирует реальную зависимость
- для этого выбираем какой-то класс моделей, обычно параметрический
- подгоняем параметры так, чтобы модель хорошо описывала данные
- то есть оптимизируем какую-то функцию ошибки или функцию качества

Нейронные сети - это очень мощный и гибкий класс моделей.

Обучение с учителем

Обучающий датасет

$$D = \{x_i, y_i\}_{i=1}^{N}$$

Модель - семейство функций

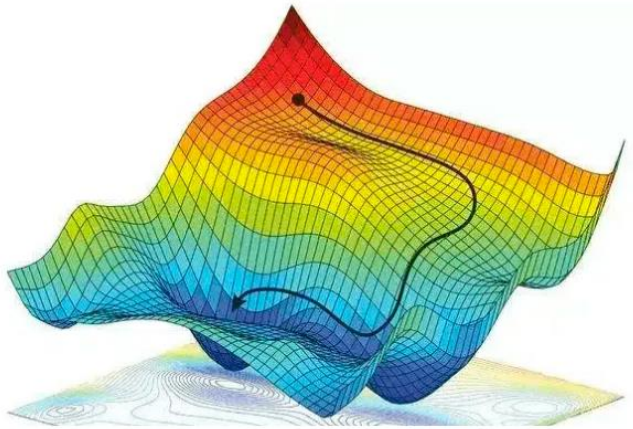
$$\hat{y} = f(x, \theta)$$

Функция потерь (Loss function)

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{y}_i) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i, \theta))$$

Нужно настроить параметры

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, L(\theta)$$



$$\theta' = \theta - \alpha\frac{\partial L}{\partial \theta}$$

Обучение с учителем

Обучающий датасет

$$D = \{x_i, y_i\}_{i=1}^{N}$$

Модель - семейство функций

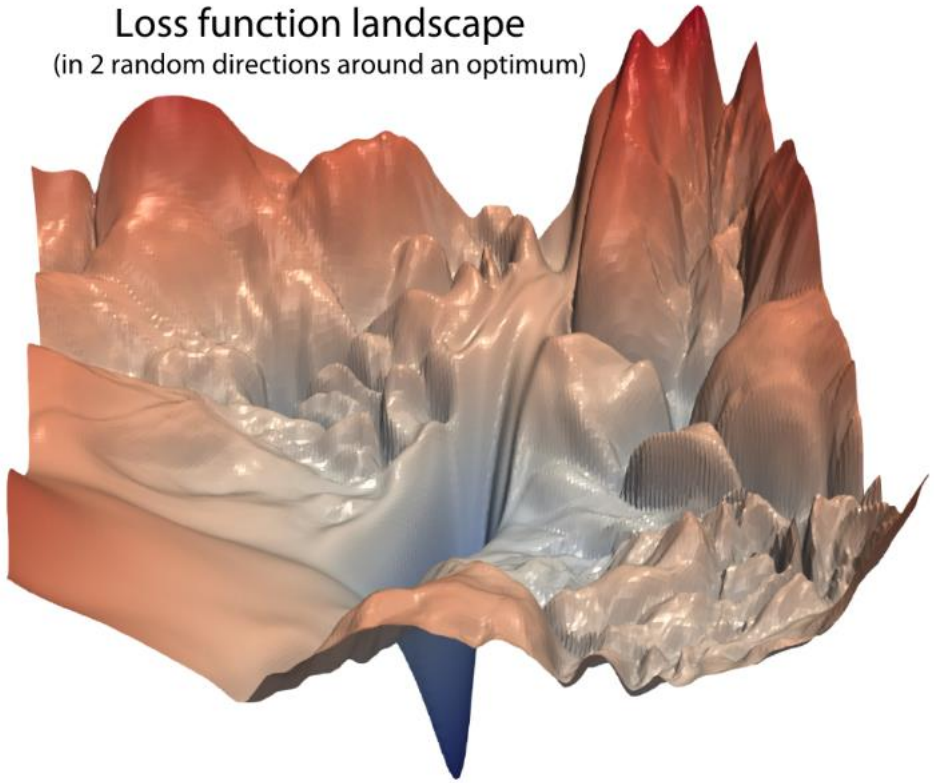
$$\hat{y} = f(x, \theta)$$

Функция потерь (Loss function)

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{y}_i) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i, \theta)),$$

Нужно настроить параметры

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, L(\theta)$$



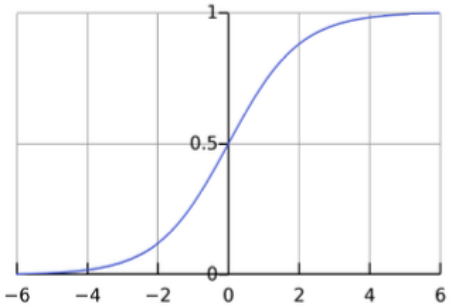
https://papers.nips.cc/paper/7875-visualizing-the-loss-landscape-of-neural-nets

$$\theta' = \theta - \alpha\frac{\partial L}{\partial \theta}$$

Логистическая регрессия

Бинарная классификация: метки $y_i \in \{0, 1\}$

Как сделать так, чтобы предсказания были от 0 до 1 (вероятности)?



$$P(y = 1|x, \theta) = \sigma(\theta^T x) = \frac{1}{1 + \exp(-\theta^T x)}$$

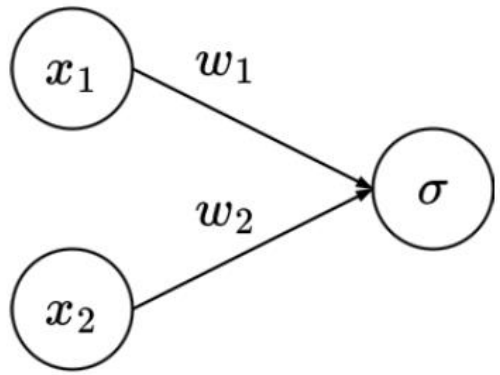
Функция потерь:

$$L(\theta) = -\sum_i (y_i \log P(y = 1|x, \theta) + (1 - y_i) \log(1 - P(y = 1|x, \theta))$$

Получаем линейную разделяющую границу

Логистическая регрессия в графическом представлении

$$y = \sigma(w_0 + w_1 x_1 + w_2 x_2)$$

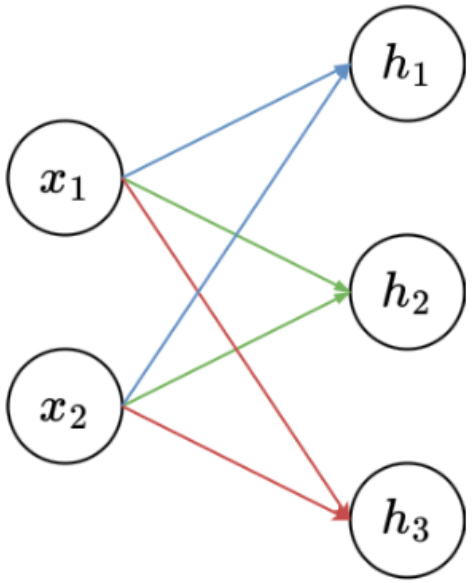


Логистическая регрессия = 1 нейрон

Вход -> линейная операция -> нелинейность -> выход

Несколько регрессий в одной форме

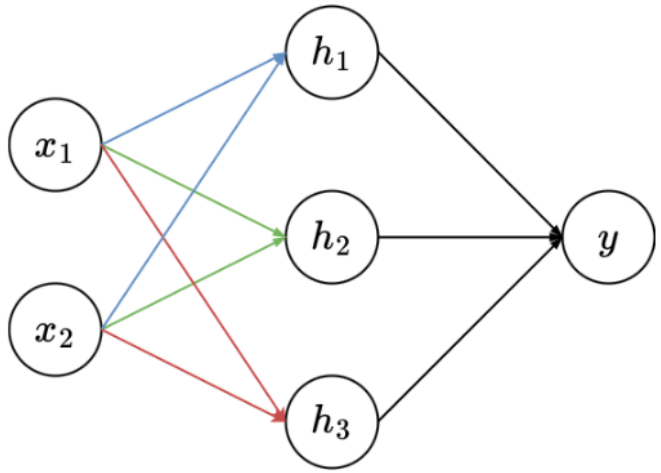
Составим три логистических регрессии с одними входами



$$h_1 = \sigma(b_1 + w_{11}x_1 + w_{12}x_2)$$
$$h_2 = \sigma(b_2 + w_{21}x_1 + w_{22}x_2)$$
$$h_3 = \sigma(b_3 + w_{31}x_1 + w_{32}x_2)$$

Логистическая регрессия в графическом представлении

Объединим выходы трёх логистических регрессий в единый выход



$$h_1 = \sigma(b_1 + w_{11}x_1 + w_{12}x_2)$$
$$h_2 = \sigma(b_2 + w_{21}x_1 + w_{22}x_2)$$
$$h_3 = \sigma(b_3 + w_{31}x_1 + w_{32}x_2)$$

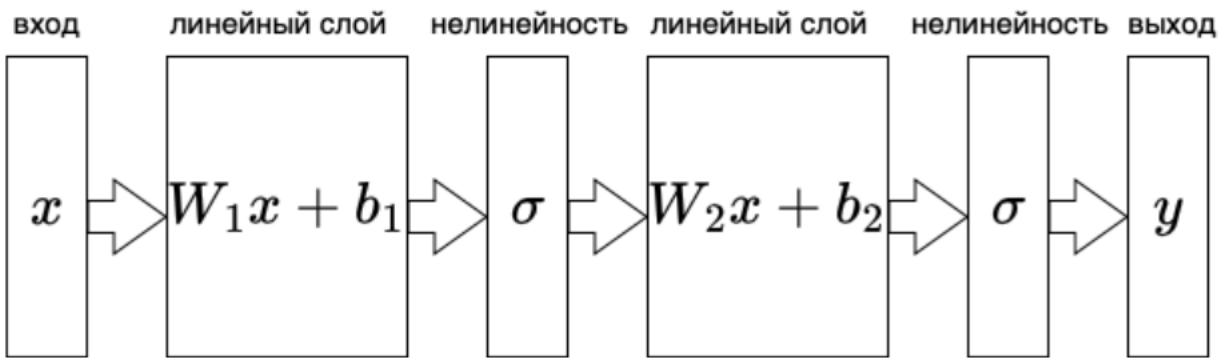
$$y = \sigma(b' + w'_{11}h_1 + w'_{12}h_2 + w'_{13}h_3)$$

Матричное представление

$$h_1 = \sigma(b_1 + w_{11}x_1 + w_{12}x_2)$$
$$h_2 = \sigma(b_2 + w_{21}x_1 + w_{22}x_2)$$
$$h_3 = \sigma(b_3 + w_{31}x_1 + w_{32}x_2)$$

$$h = \sigma(Wx + b)$$
$$y = \sigma(W'h + b')$$

$$y = \sigma(b' + w'_{11}h_1 + w'_{12}h_2 + w'_{13}h_3)$$

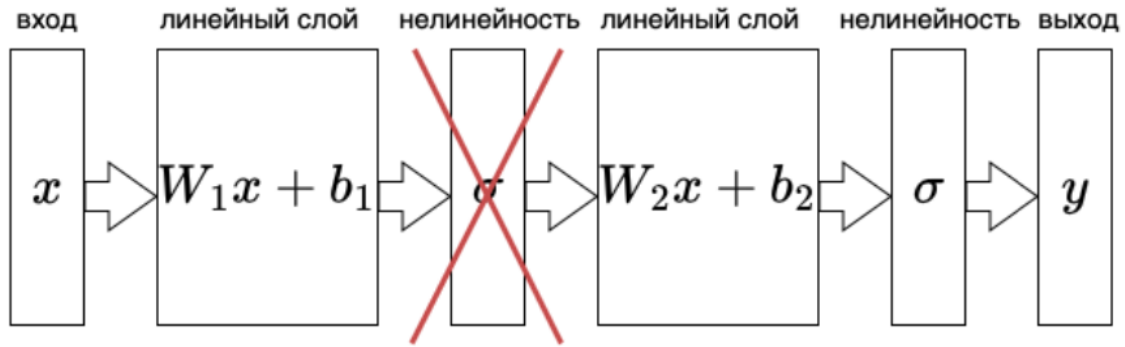


$$y = \sigma(W_2\sigma(W_1x + b_1) + b_2)$$

Логистическая регрессия в графическом представлении

Что если уберем нелинейность?

Суперпозиция линейных функций - снова линейная функция.



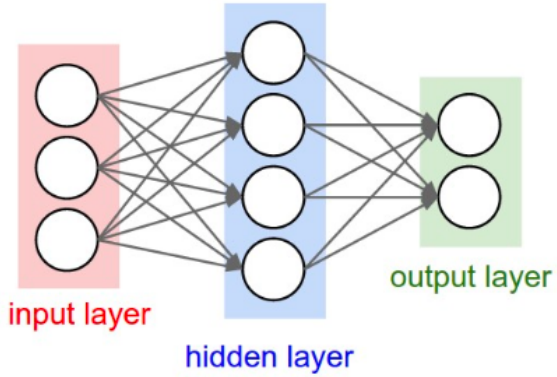
$$y = \sigma(W_2(W_1x+b_1)+b_2) = \sigma(W_2W_1x+W_2b_1+b_2) = \sigma(W'x+b')$$

Но! Не все нелинейности одинаково полезны!

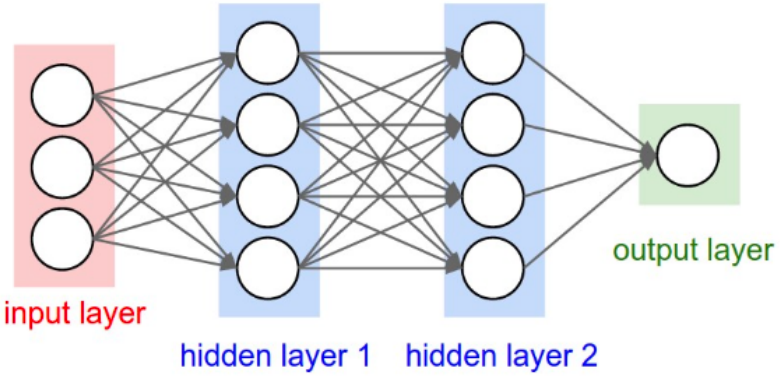
Multilayer Perceptron (MLP)

Многослойный перцептрон

$$y = \sigma(W_2\sigma(W_1x + b_1) + b_2)$$



$$y = \sigma(W_3\sigma(W_2\sigma(W_1x + b_1) + b_2) + b_3)$$



W - веса (weights), b - смещение (bias), σ - функция активации

Организация в слои - можно представить в виде матричных операций и легко параллелизовать

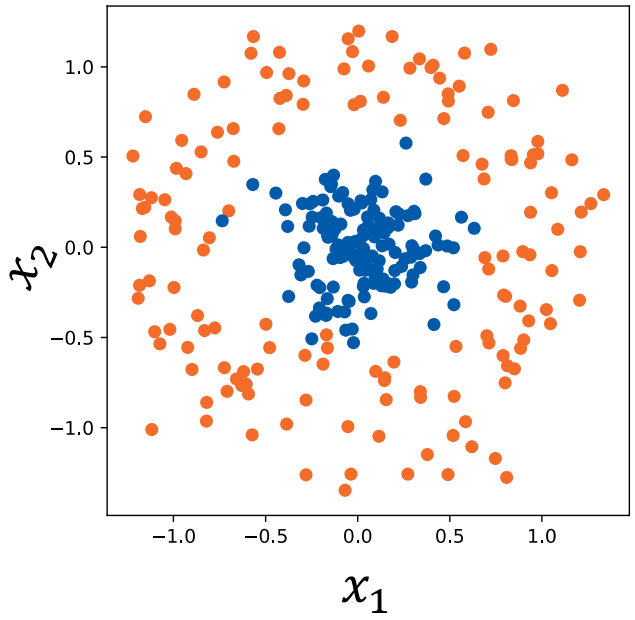
Развитие глубокого обучения

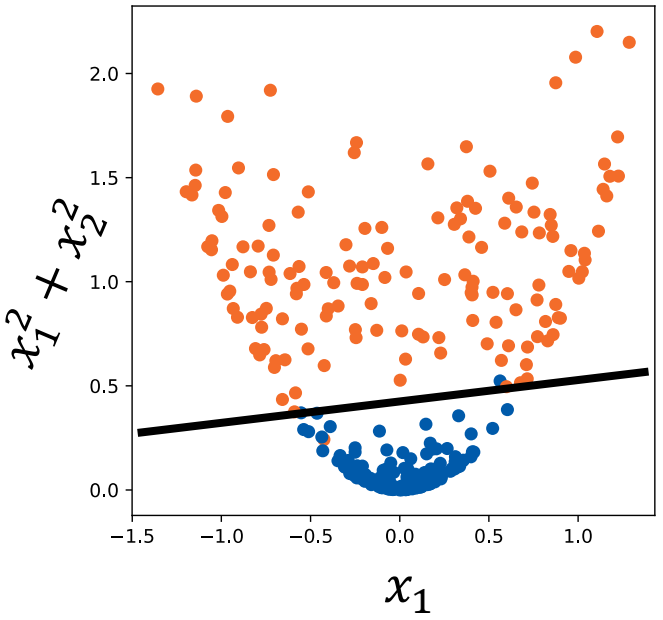
- Вычислительные мощности (GPUs, TPUs)
- Огромное количество данных
- Улучшение алгоритмов и понимания того, как обучать глубокие сети
- Open source инструменты и модели
- Интерес бизнеса, больших корпораций

От линейной модели к нейронной сети

Линейные модели b извлечение признаков

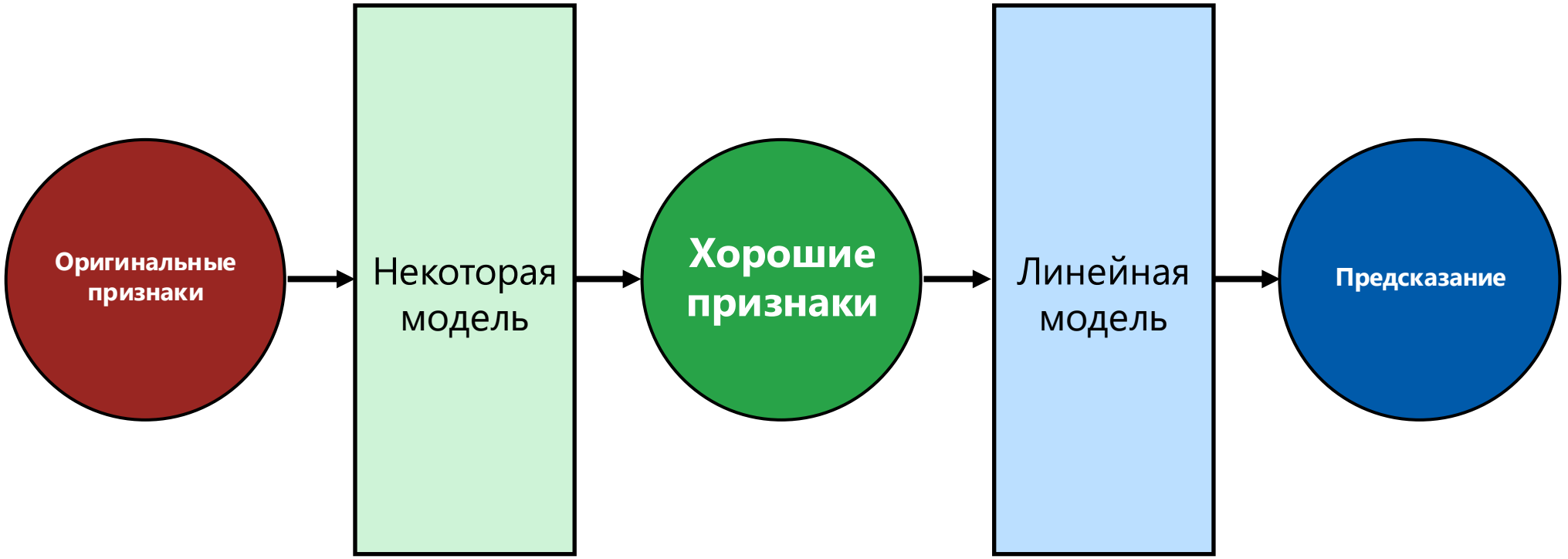
- Для линейных моделей мы **придумываем** новые признаки, чтобы сделать модель более мощной
- Поиск хороших функций (так называемая **feature engineering**) - задача крайне нетривиальная.





Автоматизация конструирования признаков – основная сила DL.

Идея: стэкинг моделей

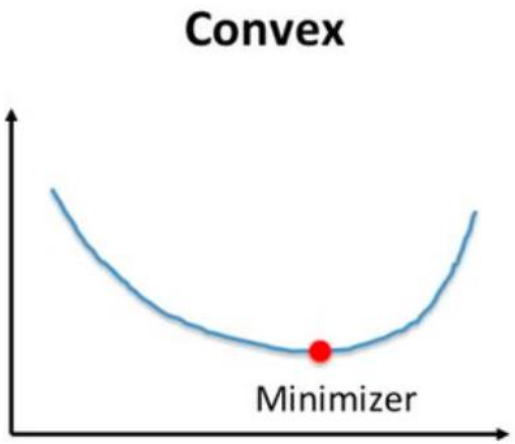


- Добавить ещё одну модель
- Всё обучим одновременно
  - Если модели **дифференцируемые**, можем использовать градиентный спуск

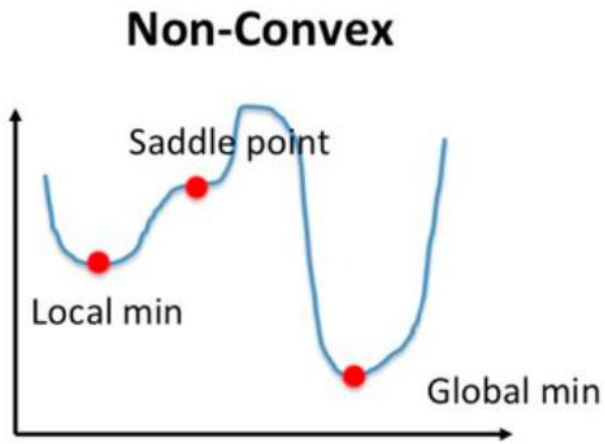
Замечание: такое суммирование моделей, вероятно, делает задачу невыпуклой
⇒ **нет гарантий сходимости**

Convex vs non-convex optimization

Выпуклая оптимизация



Невыпуклая оптимизация

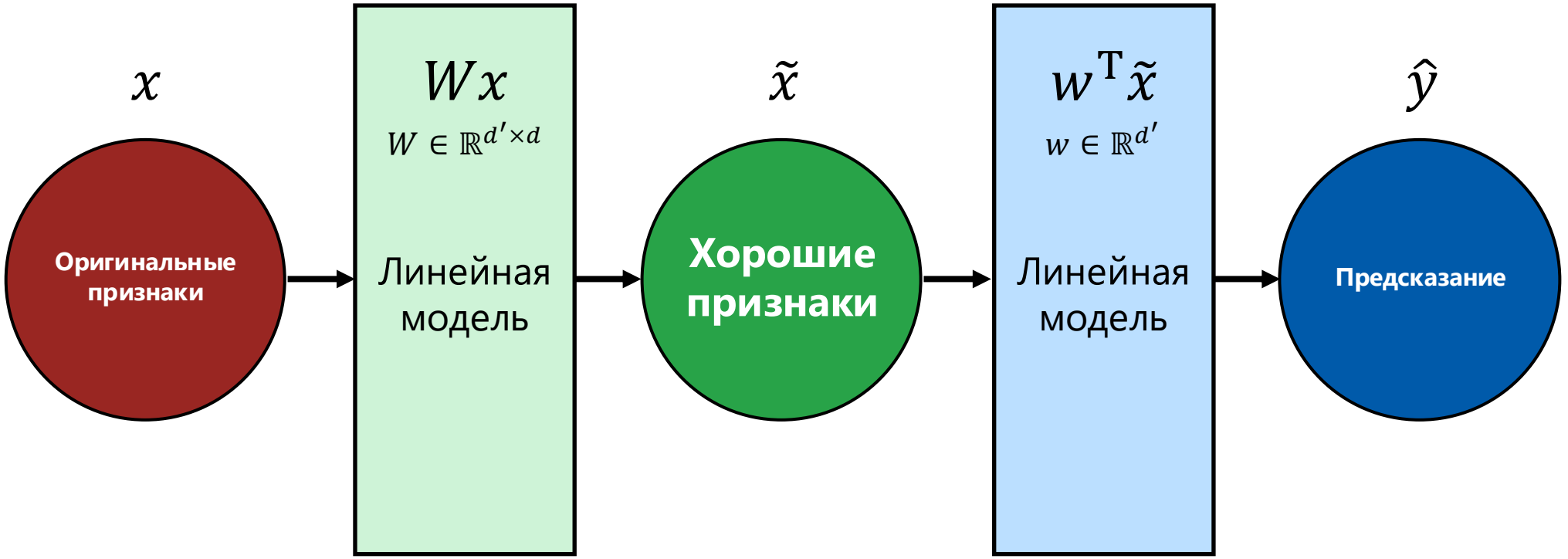


https://ru.pinterest.com/pin/convex-nonconvex-functions--672232681861372201/

Множество минимумов, неявные точки перегиба

Замечание: такое суммирование моделей, вероятно, делает задачу невыпуклой
⇒ нет гарантий сходимости

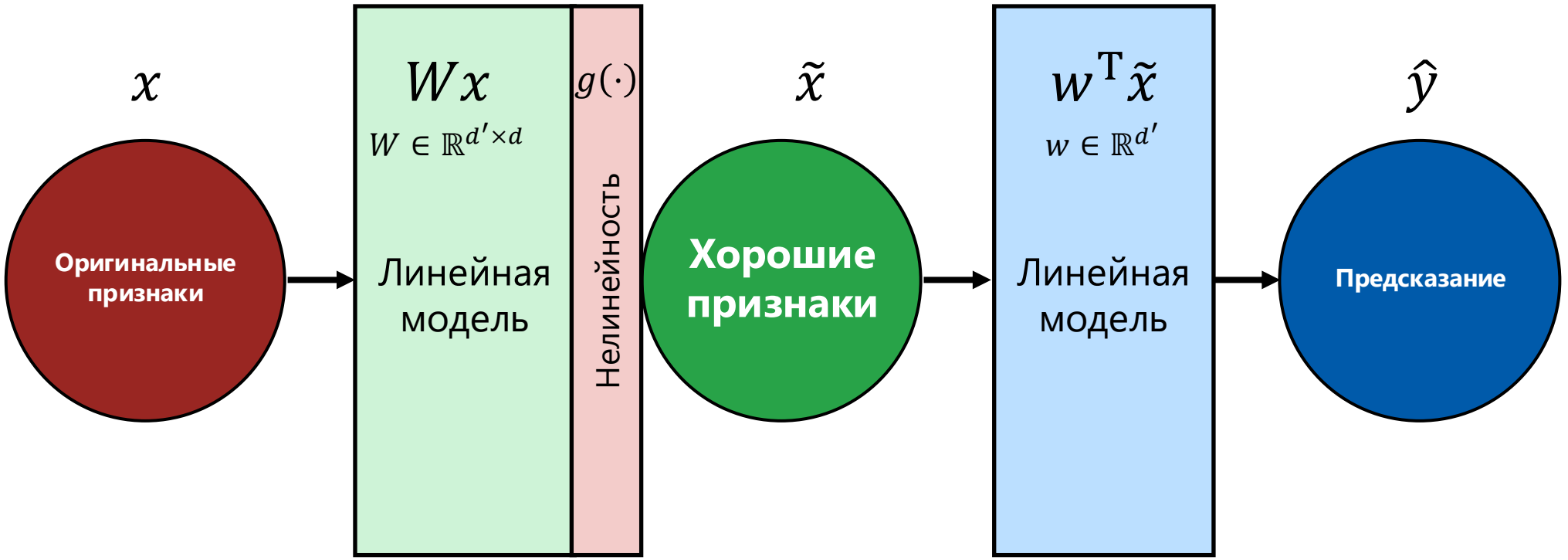
Какие модели идут в стэк?



$$\hat{y} = w^{\mathrm{T}}\tilde{x} = w^{\mathrm{T}}(Wx) = \left(w^{\mathrm{T}}W\right)x = w'^{\mathrm{T}}x$$

- Всё становится линейной моделью

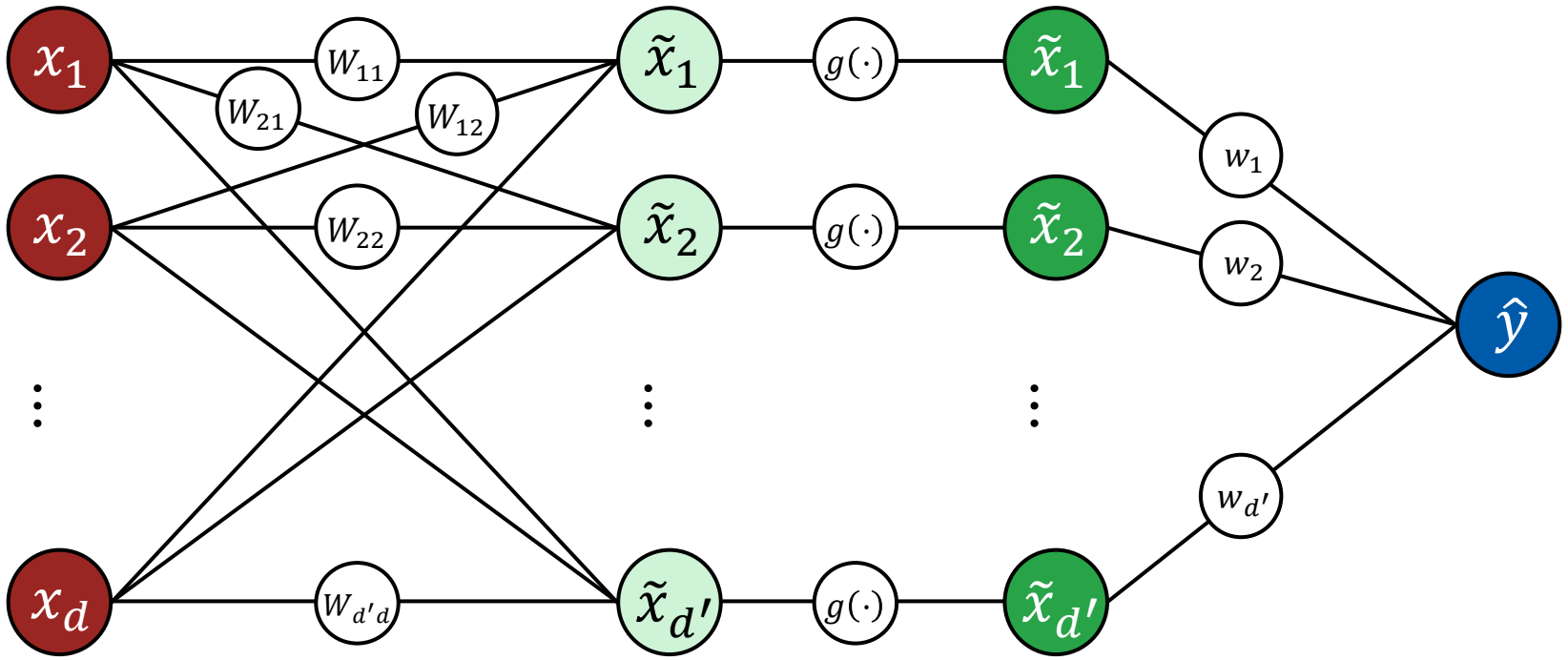
Введём нелинейность



$$\hat{y} = w^{\mathrm{T}}\tilde{x} = w^{\mathrm{T}} g(Wx)$$

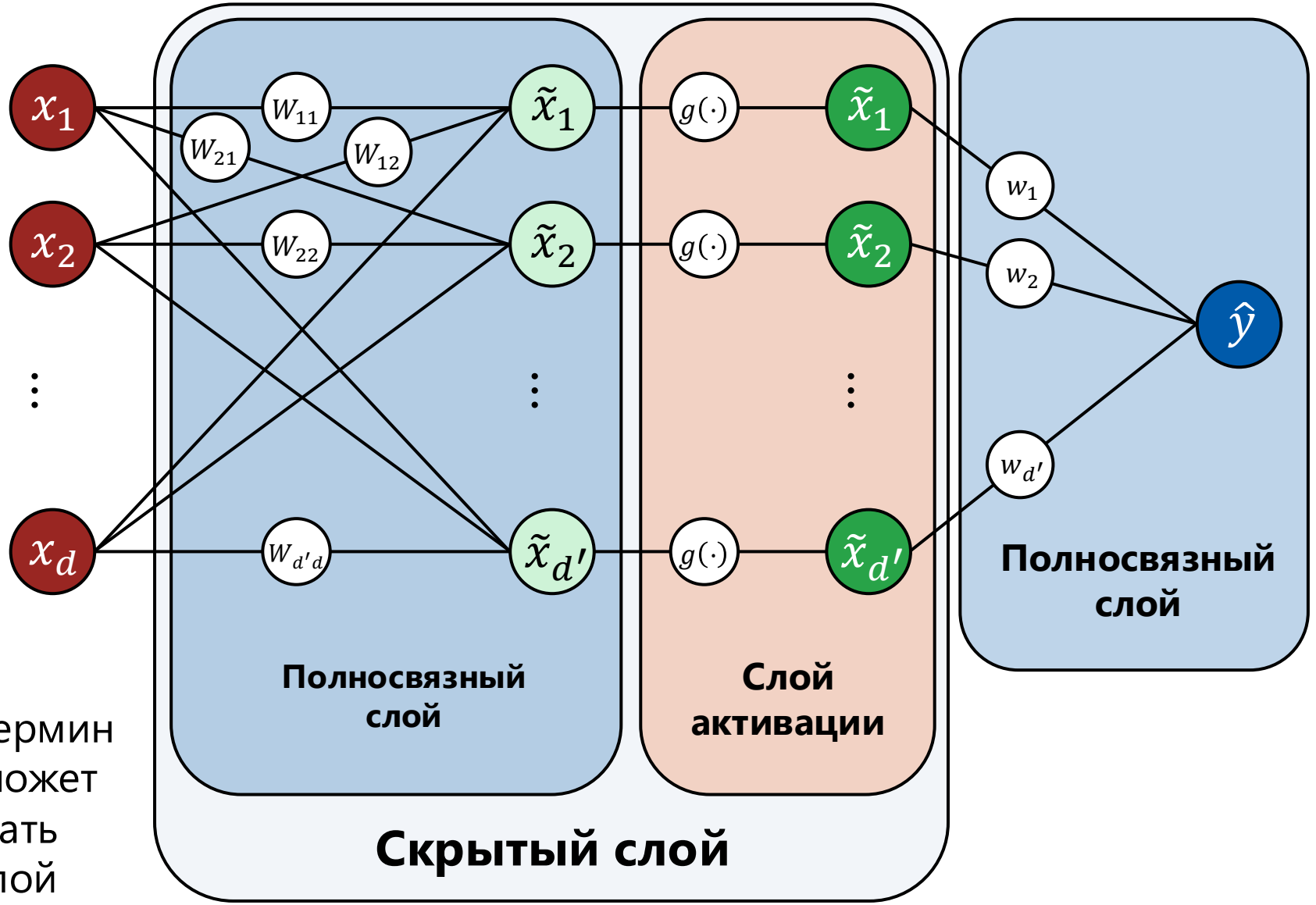
$g(\cdot)$ – некоторая **нелинейная** скалярная функция (поэлементная)

Детализированный вид



$$\hat{y} = w^{\mathrm{T}}\tilde{x} = w^{\mathrm{T}}g(Wx) = \sum_j \left[w_j g\left(\sum_i W_{ji} x_i \right) \right]$$

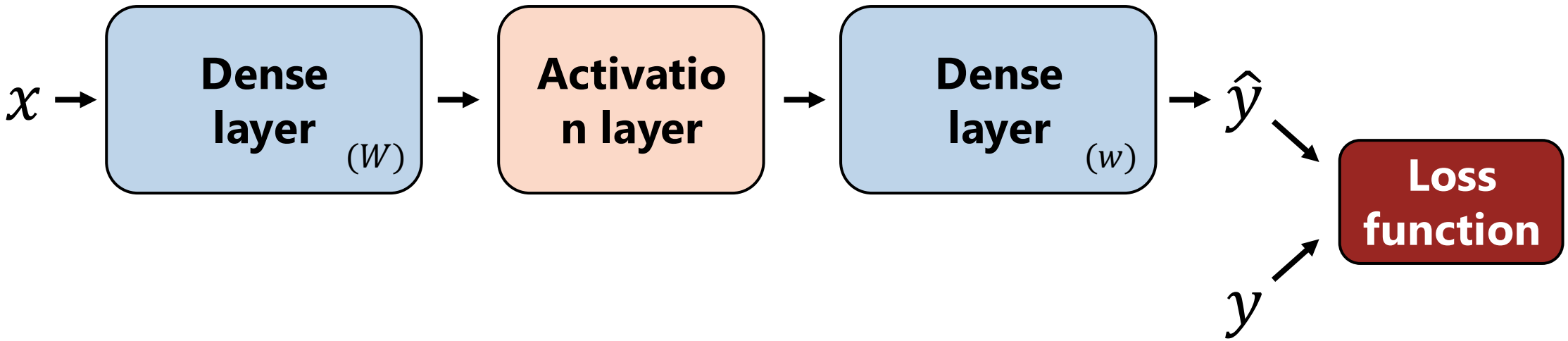
Терминология



Примечание: термин «активация» может также означать выходной слой

Обратное распространение ошибки

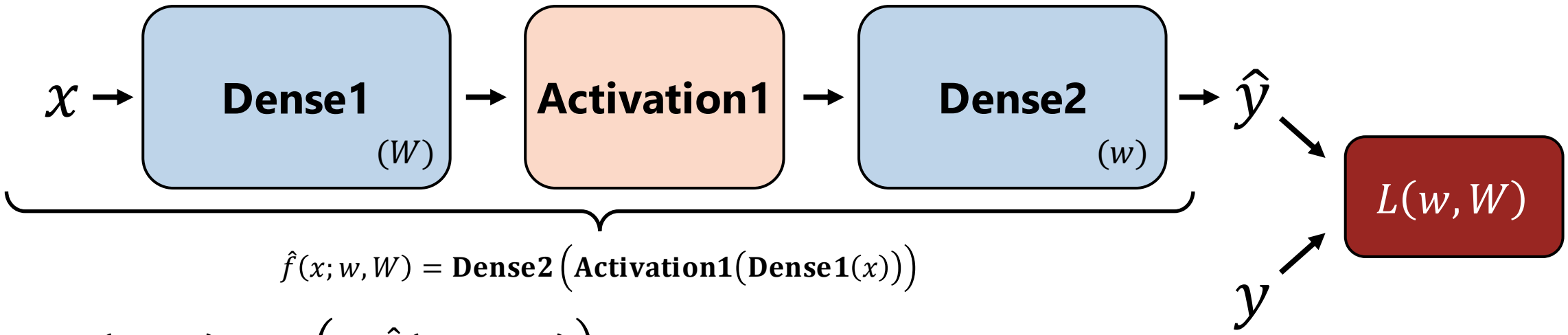
Функция потерь



- Например: квадрат ошибки

$$L = \frac{1}{N} \sum_{i=1\ldots N} \left(y_i - w^{\mathrm{T}} g(W x_i) \right)^2$$

Производные

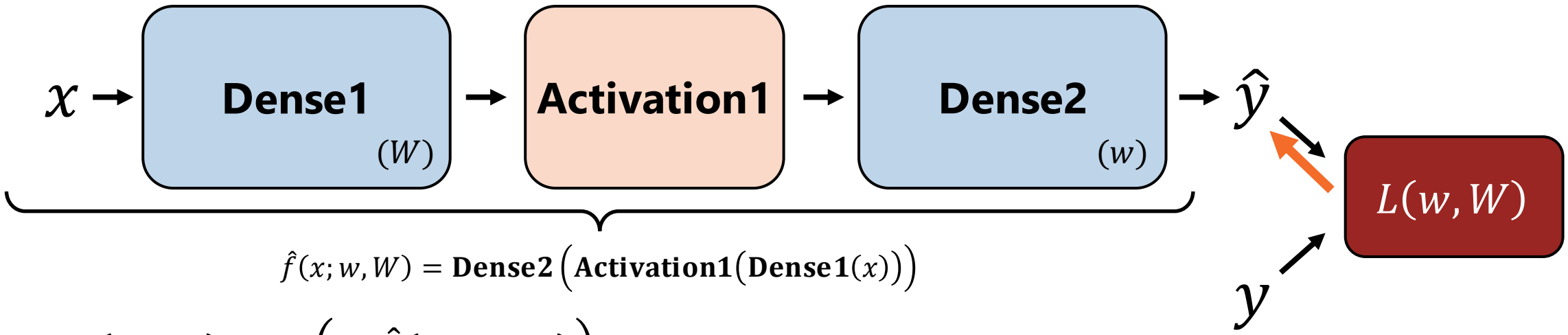


$$\hat{f}(x; w, W) = \mathbf{Dense2}\left(\mathbf{Activation1}\left(\mathbf{Dense1}(x)\right)\right)$$

$$L(w, W) \equiv L\left(y, \hat{f}(x; w, W)\right)$$

$$\frac{\partial L}{\partial W} = \frac{\partial L(y, \hat{f})}{\partial \hat{f}} \cdot \frac{\partial \hat{f}}{\partial W} =$$

Производные

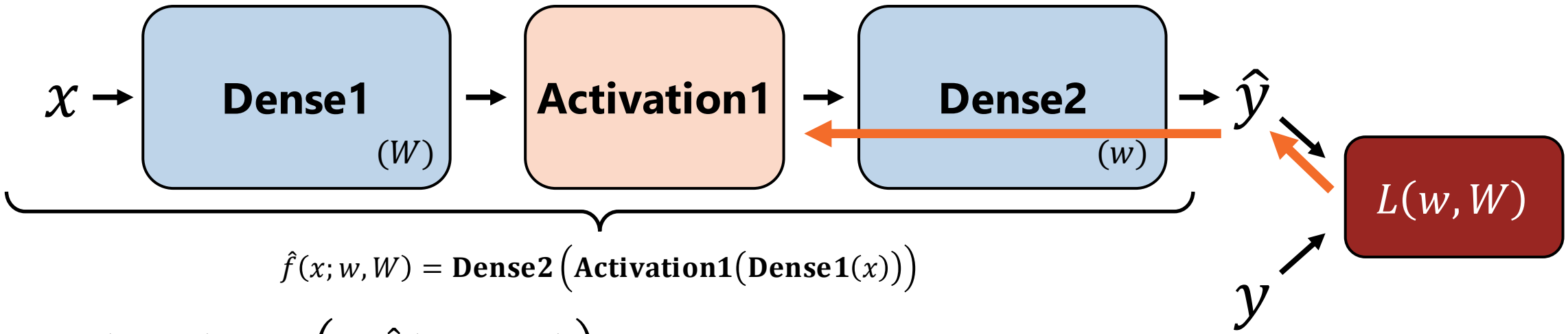


$$\hat{f}(x; w, W) = \mathbf{Dense2}\left(\mathbf{Activation1}\left(\mathbf{Dense1}(x)\right)\right)$$

$$L(w, W) \equiv L\left(y, \hat{f}(x; w, W)\right)$$

$$\frac{\partial L}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot \frac{\partial \hat{f}}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot$$

Снова производные

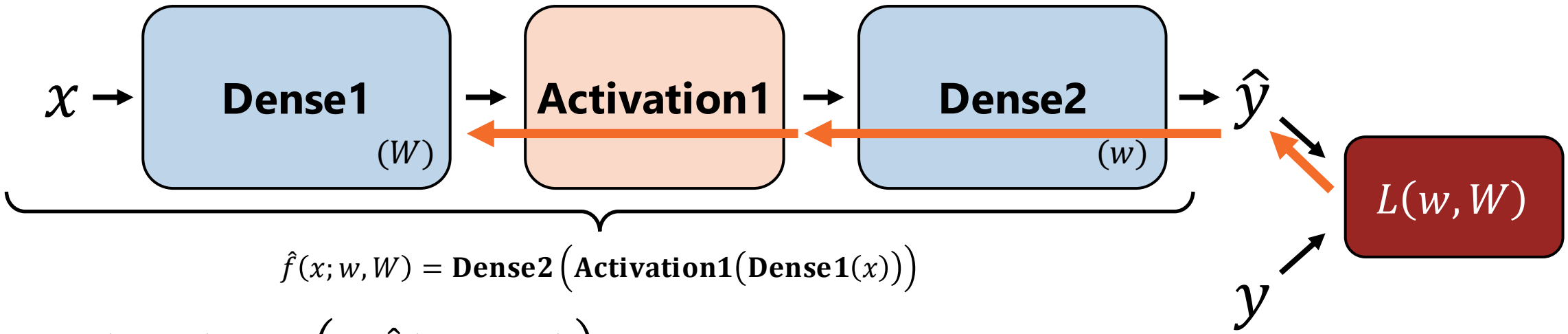


$$\hat{f}(x; w, W) = \mathbf{Dense2}\left(\mathbf{Activation1}\left(\mathbf{Dense1}(x)\right)\right)$$

$$L(w, W) \equiv L\left(y, \hat{f}(x; w, W)\right)$$

$$\frac{\partial L}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot \frac{\partial \hat{f}}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot \frac{\partial \mathbf{Dense2}}{\partial \mathbf{Activation1}} \cdot$$

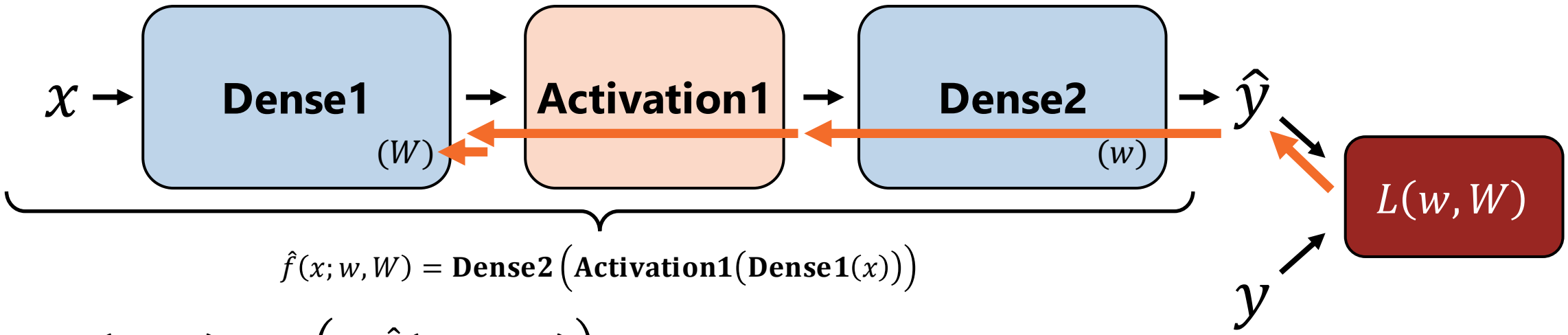
Calculating derivatives



$$L(w, W) \equiv L\left(y, \hat{f}(x; w, W)\right)$$

$$\frac{\partial L}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot \frac{\partial \hat{f}}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot \frac{\partial \mathbf{Dense2}}{\partial \mathbf{Activation1}} \cdot \frac{\partial \mathbf{Activation1}}{\partial \mathbf{Dense1}} \cdot$$

Calculating derivatives

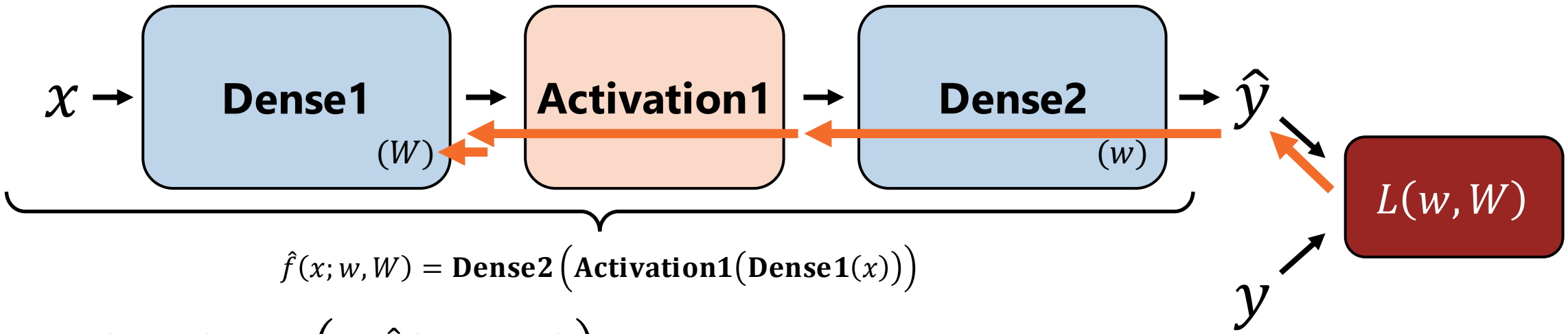


$$L(w, W) \equiv L\left(y, \hat{f}(x; w, W)\right)$$

$$\frac{\partial L}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot \frac{\partial \hat{f}}{\partial W} = \frac{\partial L\left(y, \hat{f}\right)}{\partial \hat{f}} \cdot \frac{\partial \mathbf{Dense2}}{\partial \mathbf{Activation1}} \cdot \frac{\partial \mathbf{Activation1}}{\partial \mathbf{Dense1}} \cdot \frac{\partial \mathbf{Dense1}}{\partial W}$$

- Backpropagation algorithm ≈ applying the chain rule
  - The actual algorithm states how to do it efficiently

Calculating derivatives



$$\hat{f}(x; w, W) = \mathbf{Dense2}\left(\mathbf{Activation1}\left(\mathbf{Dense1}(x)\right)\right)$$

$$L(w, W) \equiv L\left(y, \hat{f}(x; w, W)\right)$$

$$\frac{\partial L}{\partial W} = \frac{\partial L(y, \hat{f})}{\partial \hat{f}} \cdot \frac{\partial \hat{f}}{\partial W} = \underbrace{\frac{\partial L(y, \hat{f})}{\partial \hat{f}}}_{\text{scalar}} \cdot \underbrace{\frac{\partial \mathbf{Dense2}}{\partial \mathbf{Activation1}}}_{d'\text{-vector}} \cdot \underbrace{\frac{\partial \mathbf{Activation1}}{\partial \mathbf{Dense1}}}_{d' \times d'\text{-matrix}} \cdot \underbrace{\frac{\partial \mathbf{Dense1}}{\partial W}}_{d' \times d' \times d\text{-tensor}}$$

$= d' \times d$-matrix

(scalar · unit matrix)

- Обратное распространение ошибки ≈ цепное правило
  - Алгоритм показывает как делать это эффективно