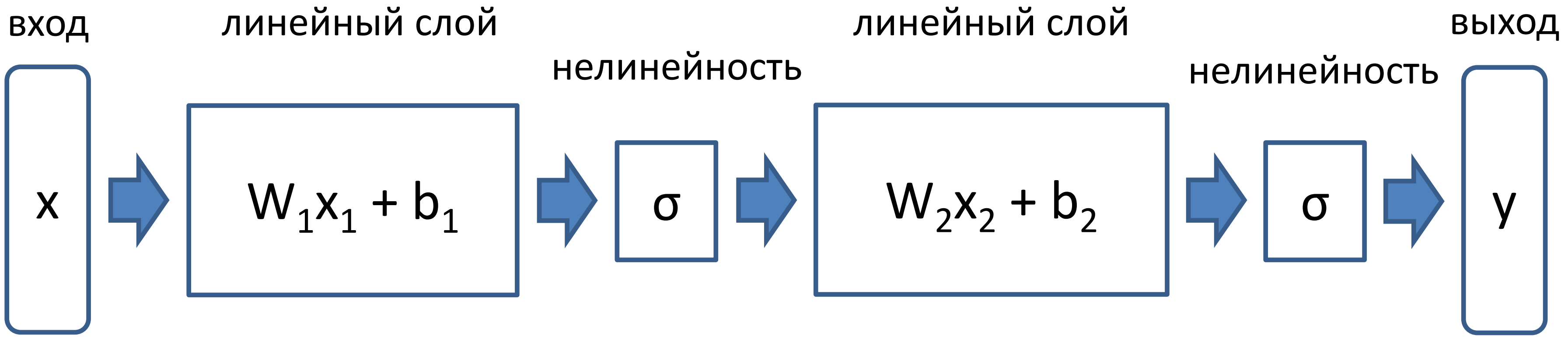
Глубинное обучение
Лекция 1: Введение

Лектор: Антон Осокин

ФКН ВШЭ, 2020

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

Нейросети прямого распространения (feed-forward neural networks)



- Ориентированный граф вычислений
  - Вершины – переменные, нейроны [Rosenblatt, 1958]
  - Рёбра обозначают зависимости
  - Слой = операция, вычисляющая переменные
  - Переменные: входы (данные), выходы, параметры, внутренние
  - Слои: линейные (+ conv), активации (sigmoid, relu) и др.

Задача приближения функции (обучение с учителем)

- Вход: объекты $x_1, \ldots, x_N \in \mathbb{R}^d$, ответы $y_1, \ldots, y_N \in \mathbb{Y}$
  - Классификация: $\mathbb{Y} = \{1, \ldots, K\}$
  - Регрессия: $\mathbb{Y} = \mathbb{R}$
- Семейство функций – нейросети f(x, θ)
  - Параметры θ линейного слоя (Wx + b): W , b (weights, biases)
- Задача – настроить параметры по выборке
- Функция потерь $\ell(f(x,\theta), y)$
  - Регрессия: $\ell(f(x,\theta), y) = (f(x,\theta) - y)^2$
  - Классификация: $\ell(f(x,\theta), y) = -\log\left(\frac{\exp f_y(x,\theta)}{\sum_{s=1}^{K} \exp f_s(x,\theta)}\right)$

Задача приближения функции (обучение с учителем)

- Вход: объекты $x_1, \ldots, x_N \in \mathbb{R}^d$, ответы $y_1, \ldots, y_N \in \mathbb{Y}$
  - Классификация: $\mathbb{Y} = \{1, \ldots, K\}$
  - Регрессия: $\mathbb{Y} = \mathbb{R}$
- Семейство функций – нейросети f(x, θ)
  - Параметры θ линейного слоя (Wx + b): W , b (weights, biases)
- Задача – настроить параметры по выборке
- Функция потерь $\ell(f(x, \theta), y)$
- Задача обучения: $\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}} \ell(f(x, \theta), y)$
- Регуляризованный эмпирический риск

$$\min_\theta \frac{1}{N} \sum_{i=1}^N \ell(f(x_i, \theta), y_i) + \mathcal{R}(\theta)$$

Обучение нейросети

- Регуляризованный эмпирический риск

$$\min_\theta \frac{1}{N} \sum_{i=1}^N \ell(f(x_i, \theta), y_i) + \mathcal{R}(\theta)$$

- Задача оптимизации – сложная (не выпуклая)
- Обычно функция дифференцируема
- Стохастическая оптимизация первого порядка
  – Stochastic gradient descent (SGD)

$$\theta_{t+1} \leftarrow \theta_t - \gamma\left(\nabla_\theta\left[\ell(f(x_i, \theta_t), y_i) + \mathcal{R}(\theta_t)\right]\right)$$

  – Вычисление градиента по параметрам – back-propagation (метод обратного распространения ошибки)

[Rumelhart&McClelland, 1986]

Back-propagation

[Rumelhart&McClelland, 1986]

- Вход: x_i, y_i, параметры θ
- Найти градиент по параметрам нейросети
- Основная идея – производная сложной функции

$$\nabla_\theta f(\theta, g(\theta), h(\theta)) = \frac{\partial f}{\partial \theta} + \frac{\partial f}{\partial g}\nabla_\theta g(\theta) + \frac{\partial f}{\partial h}\nabla_\theta h(\theta)$$

- Автоматическое дифференцирование =

  построение полных производных на основе слоев

- Слой $h(x,\theta)$ – абстракция, поддерживающая операции
  - проход вперёд – вычисление $h(x,\theta)$ при известных x и θ
  - проход назад – вычисление $\nabla_x h(x,\theta)$ и $\nabla_\theta h(x,\theta)$

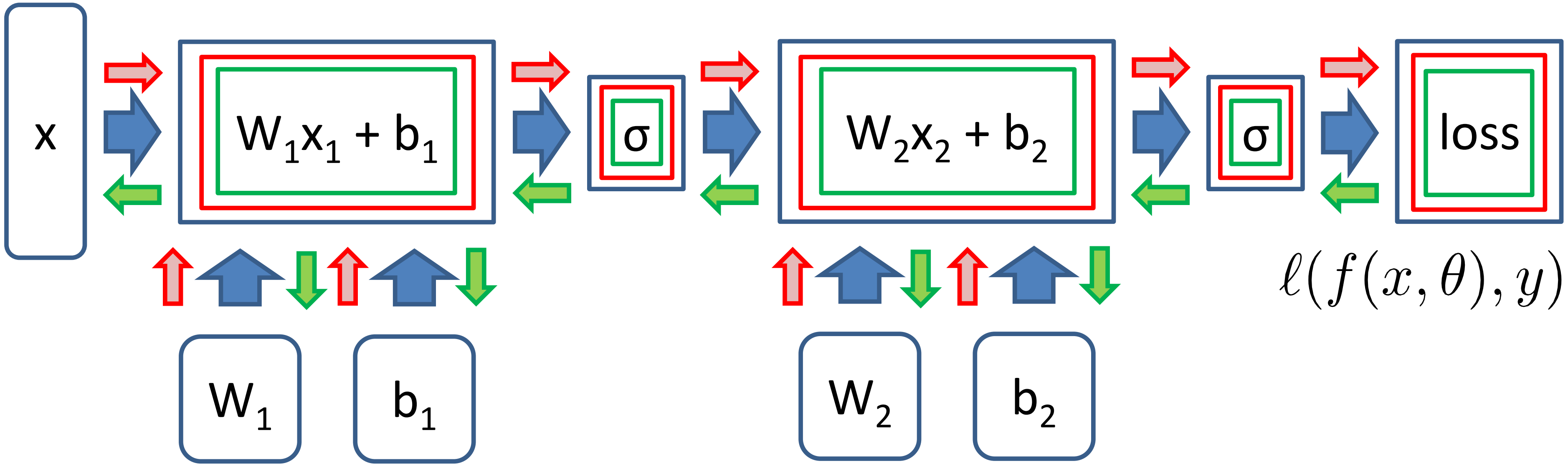
    при известных x, θ, ∇_h

Обычно $\nabla_h = \frac{d}{dh}\ell(f(x,\theta),y)$ – это градиент потерь по узлам *h*

Back-propagation

[Rumelhart&McClelland, 1986]

- Вход: x_i, y_i, параметры W_1, b_1, W_2, b_2
- Найти градиент по параметрам нейросети



$\ell(f(x, \theta), y)$

1. Проход вперёд (вычисление слоёв и функции потерь)
2. Проход назад (вычисление градиентов)

Дифференцирование одного слоя

- Слой $h(x, \theta)$ – абстракция, поддерживающая операции
  - проход вперёд – вычисление $h(x, \theta)$ при известных x и θ
  - проход назад – вычисление $\nabla_x h(x, \theta)$ и $\nabla_\theta h(x, \theta)$ при известных x, θ, ∇_h
- Функция потерь $\ell(f, y)$
  - Выходного градиента нет (можно считать константой 1)
  - Частные производные по выходам нейросети $\frac{\partial \ell}{\partial f}$

Дифференцирование одного слоя

- Слой $h(x, \theta)$ – абстракция, поддерживающая операции
  - проход вперёд – вычисление $h(x, \theta)$ при известных x и θ
  - проход назад – вычисление $\nabla_x h(x, \theta)$ и $\nabla_\theta h(x, \theta)$ при известных x, θ, ∇_h
- Функция потерь $\ell(f, y)$
  - Выходного градиента нет (можно считать константой 1)
  - Частные производные по выходам нейросети $\frac{\partial \ell}{\partial f}$
- Функции активации $\sigma(x) = \frac{1}{1+\exp(-x)}$, $\mathrm{ReLu}(x) = \begin{cases} 0, x \leq 0, \\ x, x > 0 \end{cases}$
  - Поэлементное дифференцирование

$$\frac{\partial \sigma(x)}{\partial x} = \sigma(x)(1 - \sigma(x)) \quad \frac{\partial \mathrm{ReLu}(x)}{\partial x} = \begin{cases} 0, x \leq 0, \\ 1, x > 0 \end{cases}$$

  - Полная производная – поэлементное произведение

$$\nabla_x = \frac{\partial h(x)}{\partial x} \odot \nabla_h$$

Дифференцирование линейного слоя

- Слой $h(x,\theta)$ – абстракция, поддерживающая операции
  - проход вперёд – вычисление $h(x,\theta)$ при известных x и θ
  - проход назад – вычисление $\nabla_x h(x,\theta)$ и $\nabla_\theta h(x,\theta)$ при известных x, θ, ∇_h
- Линейный слой $h(x,W) = Wx$
  - Известны x и ∇_h
  - Найти ∇_x

$$\begin{pmatrix} h_1 \\ h_2 \end{pmatrix} = \begin{pmatrix} w_{11} w_{12} w_{13} \\ w_{21} w_{22} w_{23} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x3 \end{pmatrix} = \begin{pmatrix} w_{11}x_1 + w_{12}x_2 + w_{13}x_3 \\ w_{21}x_1 + w_{22}x_2 + w_{23}x_3 \end{pmatrix}$$

$$\nabla_{x_1} = \nabla_{h_1} \frac{\partial h_1}{\partial x_1} + \nabla_{h_2} \frac{\partial h_2}{\partial x_1} = (w_{11} w_{21}) \begin{pmatrix} \nabla_{h_1} \\ \nabla_{h_2} \end{pmatrix} \Rightarrow \nabla_x = W^T \nabla_h$$

$$\nabla_{w_{21}} = \nabla_{h_1} \frac{\partial h_1}{\partial w_{21}} + \nabla_{h_2} \frac{\partial h_2}{\partial w_{21}} = \nabla_{h_2} x_1 \Rightarrow \nabla_W = \nabla_h x^T$$

Тестирование дифференцирования

Ошибки в вычислении градиентов встречаются часто!
Очень сложно находить!
Система может работать, но на 5-10% хуже

Unit test: конечные разности (finite differences)

$$\nabla_\theta h(x,\theta) \approx \frac{h(x,\theta+\varepsilon)-h(x,\theta-\varepsilon)}{2\varepsilon}$$

Как выбрать ε?

Тестирование дифференцирования

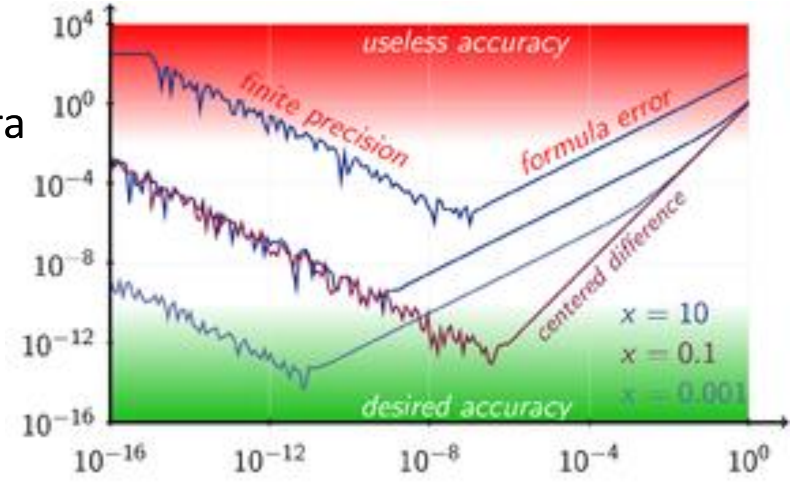
Ошибки в вычислении градиентов встречаются часто!
Очень сложно находить!
Система может работать, но на 5-10% хуже

Unit test: конечные разности (finite differences)

$$\nabla_\theta h(x,\theta) \approx \frac{h(x,\theta+\varepsilon)-h(x,\theta-\varepsilon)}{2\varepsilon}$$

Как выбрать ε?

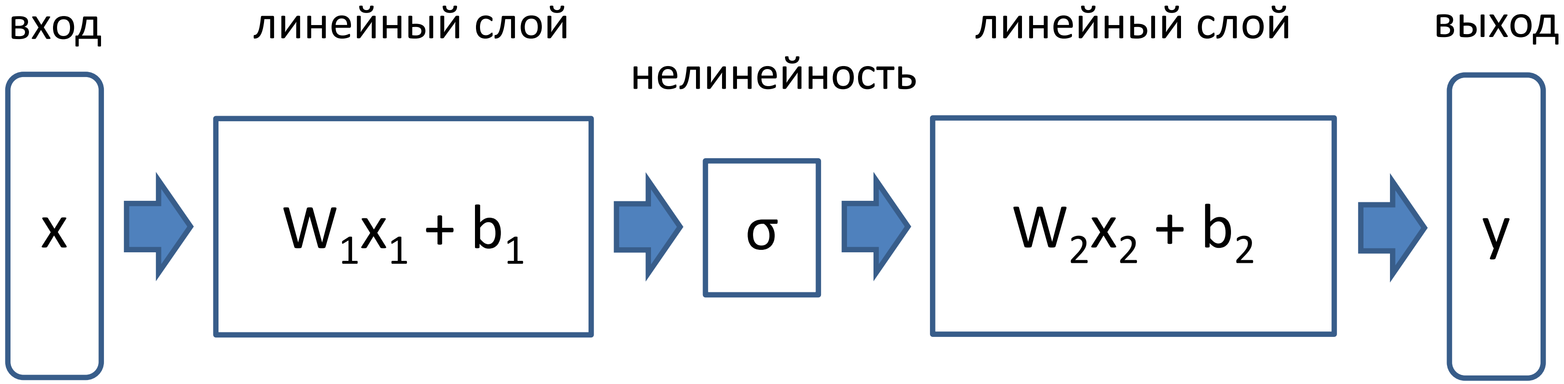
ошибка
градиента



ε

график из википедии

Нейросети прямого распространения = универсальные аппроксиматоры



Теорема об универсальной аппроксимации [Cybenko, 1989]

Любую функцию можно с любой точностью приблизить нейросетью глубины 2 с сигмоидной функцией активации

Задача решена? Строим AI?

Проблемы нейросетей

- Для аппроксимации может понадобиться слишком много узлов
  - Большая глубина даёт выразительность меньшим числом узлов

Проблемы нейросетей

- Для аппроксимации может понадобиться слишком много узлов
  – Большая глубина даёт выразительность меньшим числом узлов
- Переобучение, слишком много параметров
  – Переиспользование параметров
  – Использование структуры данных (архитектуры)
  – Регуляризация
- Нестабильное обучение, плохие решения
  – Методы оптимизации, архитектуры
- Очень большое время обучения!
  – Эффективные алгоритмы и реализации
  – Hardware: GPU, кластеры, кластеры GPU, TPU, etc.

Где взять данные?

Проблема: для обучения нужно много размеченных данных!

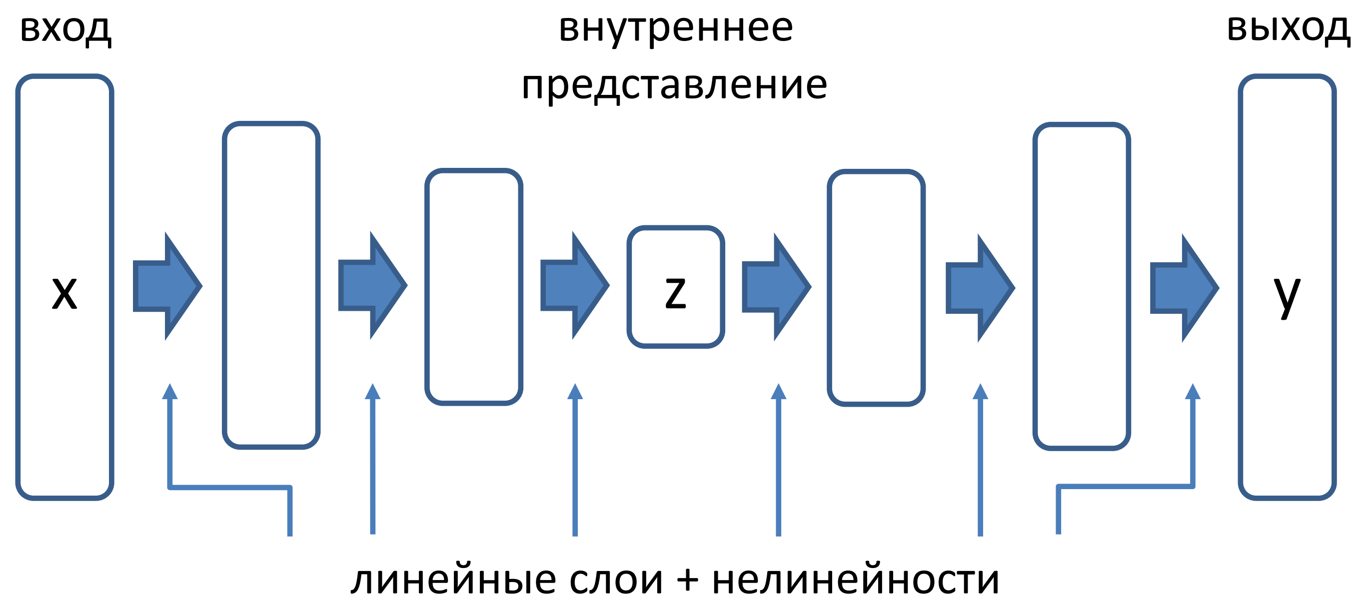
Большинство прорывов связаны с обучением с учителем на больших данных!

Варианты:

1. Размечать ☺ (Amazon MTurk, Яндекс.Толока)
2. Использовать разметку на смежных данных
   - Нейросети позволяют выучивать полезные представления!
   - Domain adaptation
3. Использовать более слабые виды учителя (weak supervision)
   - Reinforcement learning (обучение с подкреплением)
   - Semi-supervised
4. **Self-supervision**: найти разметку в самих данных!

Пример self-supervision: автокодировщик

Модель для восстановление данных через bottleneck



Обучение – восстановление входа $(x\text{-}y)^2$

z – внутреннее представление, которое можно использовать

Заключение

- Нейросети прямого распространения
- Алгоритм back-propagation
- Тестирование производных через конечные разности!
- При использовании нейросетей возникает много проблем, о которых мы в рамках этого курса и поговорим