

# Глубинное обучение

## Лекция 9: Дифференцируемое программирование

Лектор: Антон Осокин

ФКН ВШЭ, 2019



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ

# План лекции

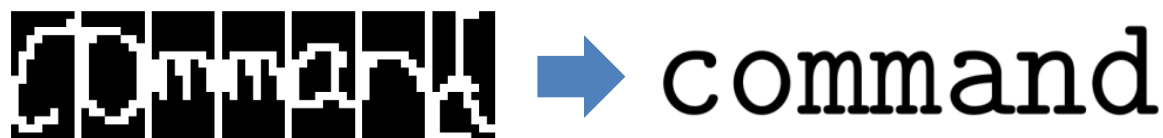
- Мотивация: зачем встраивать алгоритмы в нейросети?
- Алгоритмы в нейросетях
  - Структурный пулинг = комбинаторная оптимизация
  - Итерации алгоритмов = слои нейросетей
  - Дифференцирование по входу алгоритма
    - Неявное дифференцирование

# Алгоритмы в нейросетях, зачем?

- У нейросетей очень сложные структуры
- Структуры сложно создавать для новых задач
- Часто нейросеть не может «выучить всё», надо помогать
- Комбинировать существующие решения и нейросети
- Существуют очень мощные алгоритмы для сложных задач

# Задача: распознавание рукописных СИМВОЛЫ

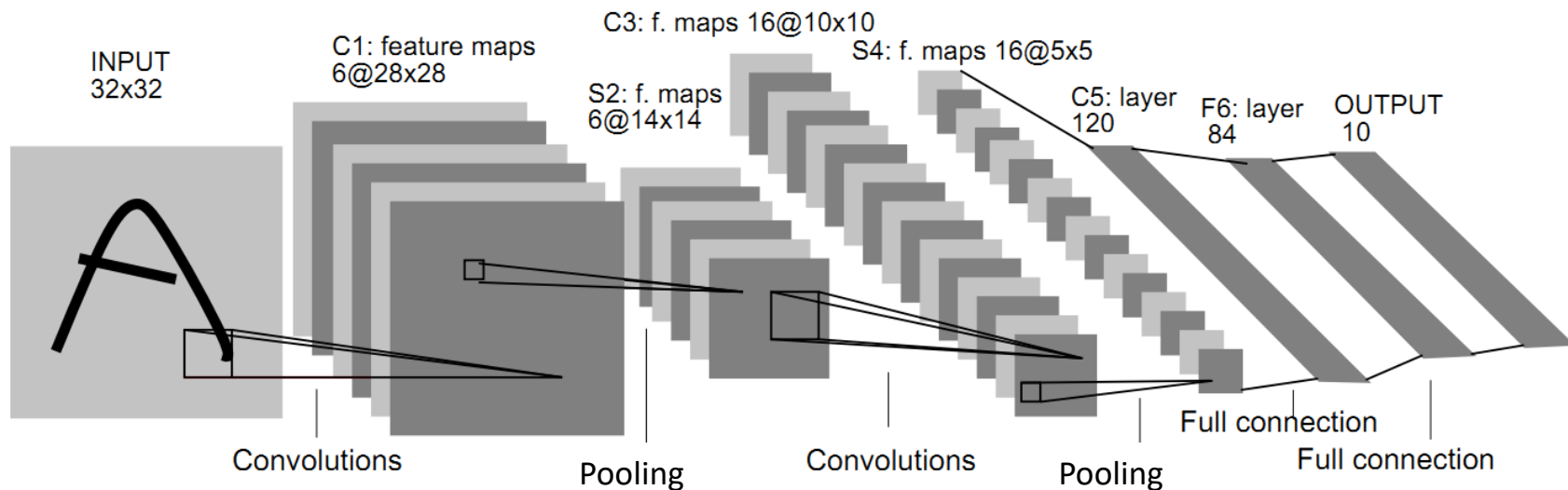
- Модельная задача – распознавание рукописных символов

 → command

- A MNIST для structured prediction
- Простая задача => очень много методов применимо
- Стандартные алгоритмы:  
динамическое программирование, СЛАУ

# Структура нейросети

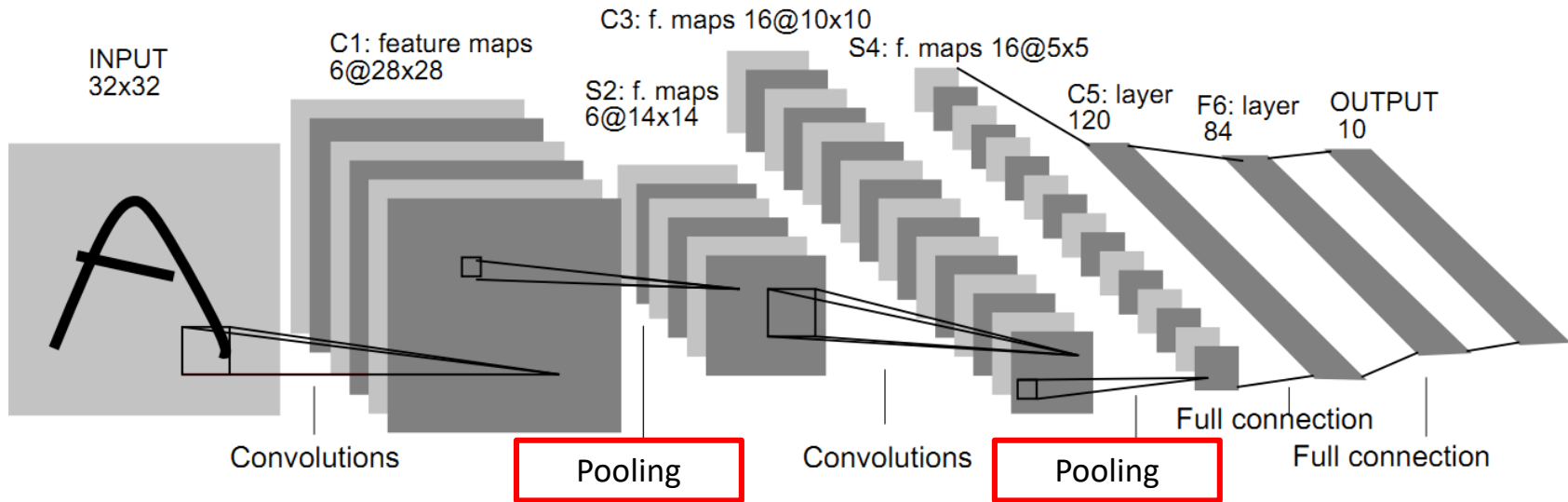
- Нейросеть для распознавания одного символа:



- Линейные операции с параметрами
  - Свёртки для изображений
- Нелинейность (sigmoid, ReLu)
- Пулинг для понижения размерности и инвариантности
- Обучение = стохастическая оптимизация

# Пулинг для выбора активаций

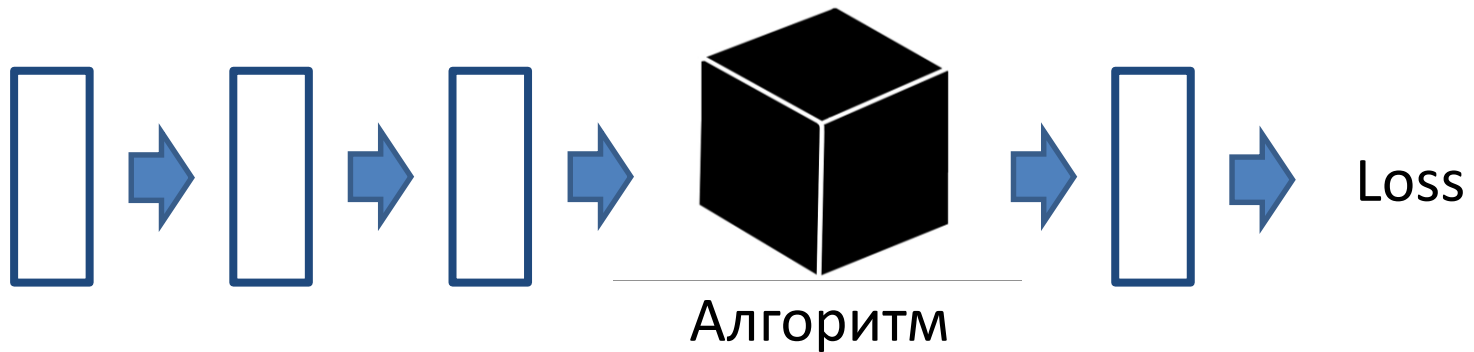
- Нейросеть для распознавания одного символа:



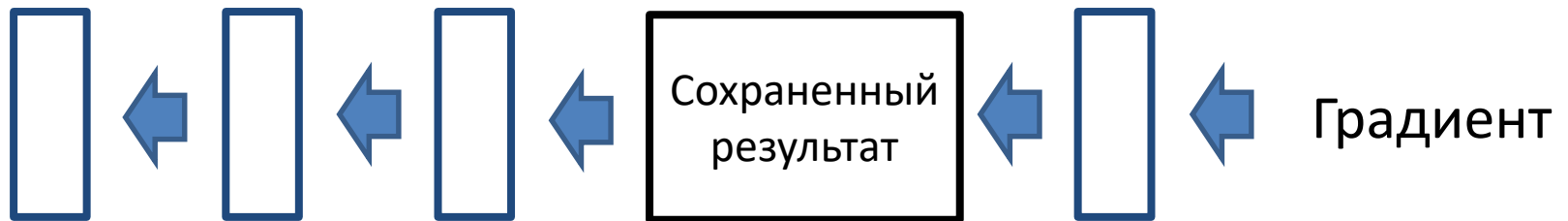
- Пулинг – процедура агрегирования активаций (max, sum)
- Алгоритмы для пулинга: квантиль, сортировка
- И это дифференцируемо?
  - «дифференцируемо» в смысле нейросетей

# Структурный пулинг = комбинаторная оптимизация

Нейросеть: проход вперёд



Для прохода назад нужен только результат алгоритма



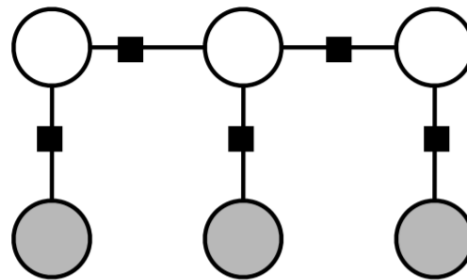
# Conditional Random Field (CRF)



- CRF – способ учесть связи между символами
- Связи задаются функцией меток, связывающей метки

$$F(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}) = \sum_{i=1}^T \theta_i(y_i \mid x_i) + \sum_{i=1}^{T-1} \theta_{i,i+1}(y_i, y_{i+1})$$

- Здесь  $\mathbf{y}$  – метки символов,  $\mathbf{x}$  – изображения символов  
 $\theta$  - потенциалы (унарные и парные)



- Графическая модель  
(фактор-граф)

- Унарные потенциалы вычисляются нейросетью



# CRF: наилучшая конфигурация

- CRF – способ учесть связи между символами
- Связи задаются функцией меток, связывающей метки

$$F(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}) = \sum_{i=1}^T \theta_i(y_i \mid x_i) + \sum_{i=1}^{T-1} \theta_{i,i+1}(y_i, y_{i+1})$$

- Здесь  $\mathbf{y}$  – метки символов,  $\mathbf{x}$  – изображения символов  
 $\theta$  - потенциалы (унарные и парные)
- Чем  $F$  больше, тем конфигурация лучше
- **Динамическое программирование** – сведение к подзадачам

- $V_i(y_i)$  – лучшее значение слагаемых  $F$  для слагаемых  $\leq i$
- Проход вперёд:

$$V_1(y_1) = \theta_1(y_1 \mid x_1)$$

$$V_{i+1}(y_{i+1}) = \theta_{i+1}(y_{i+1} \mid x_{i+1}) + \max_{y_i} \left( \theta_{i,i+1}(y_i, y_{i+1}) + V_i(y_i) \right)$$

- Оптимальное значение:  $F^* = \max_{\mathbf{y}_T} V_T(y_T)$
- Проход назад для восстановления конфигурации

# Обучение CRF – структурный SVM

- Обучение по размеченной выборке  $\{x_n, y_n\}_{n=1}^N$
- Обучение – задача оптимизации  $\min_{\theta} \frac{1}{N} \sum_{n=1}^N \Psi(x_n, y_n \mid \theta)$

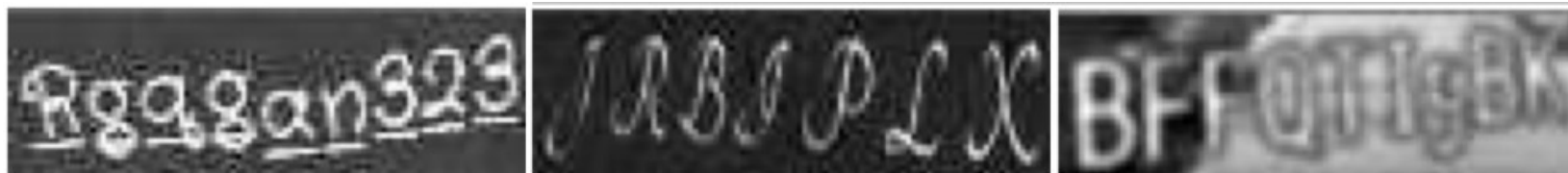
- SSVM - обобщение метода опорных векторов

$$\Psi(x_n, y_n \mid \theta) = \max_y \left[ F(y \mid x_n, \theta) + \Delta(y, y_n) \right] - F(y_n \mid x_n, \theta)$$

- Общая схема метода
  1. Вычисление потенциалов (проход вперёд через нейросеть)
  2. Вычислении функции потерь (дин. программирование)
  3. Градиент по выходу нейросети
  4. Градиент по параметрам нейросети (backprop)
  5. Шаг оптимизации

# Примеры использования

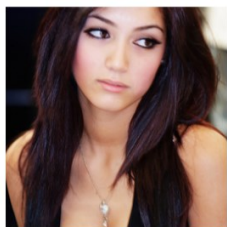
- Свободный текст (Jaderberg et al., ICLR 2015)



- Детектирование нескольких объектов (Vu et al., ICCV 2015)



- Тэггирование изображений (Chen et al., ICML 2015)



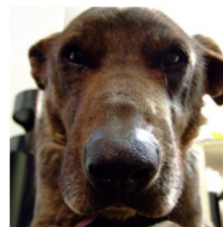
female/indoor/portrait



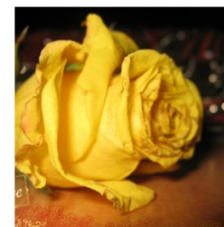
sky/plant life/tree



water/animals/sea



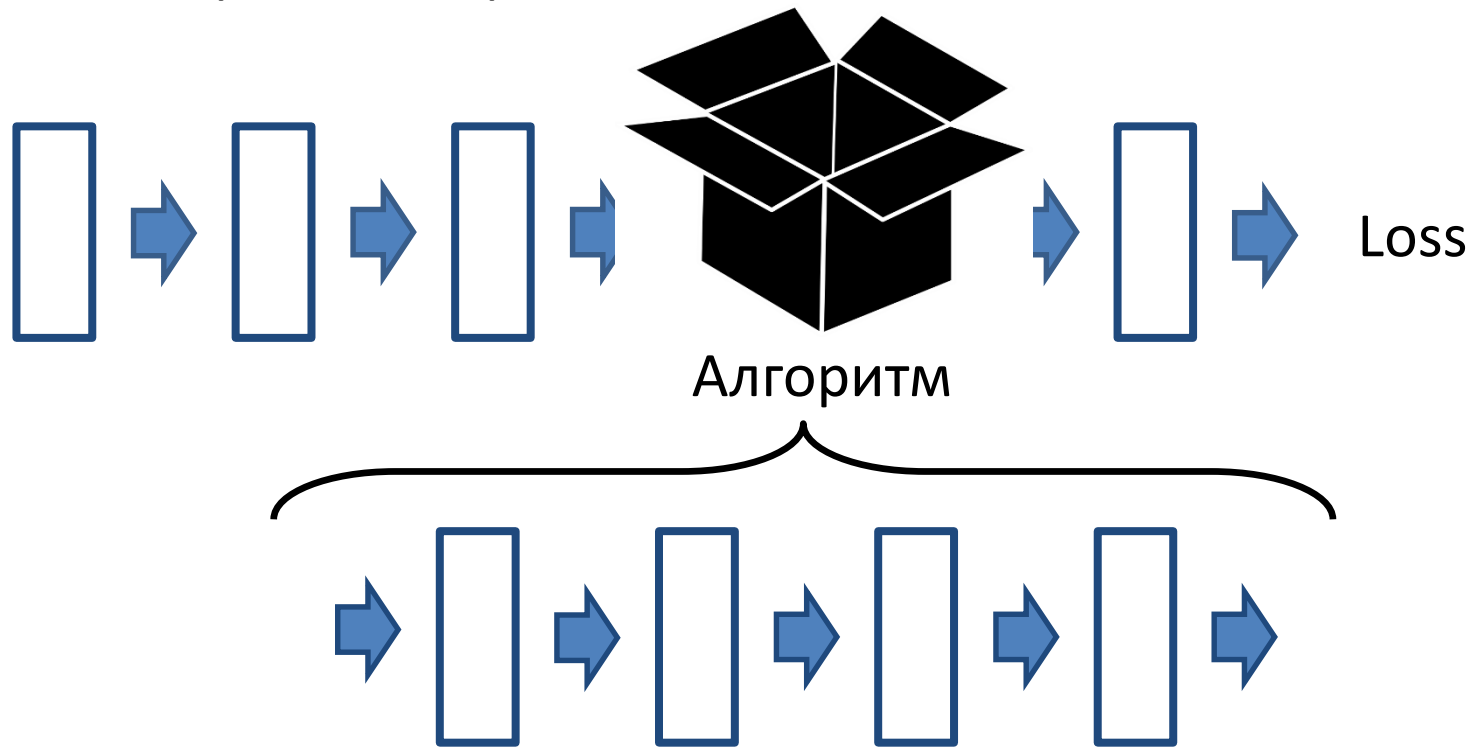
animals/dog/indoor



indoor/flower/plant life

# Итерации алгоритма как слои сети

Нейросеть: проход вперёд



Проход назад – обычный back propagation

# Обучение CRF – максимальное правдоподобие

 → command

- Обучение по размеченной выборке  $\{x_n, y_n\}_{n=1}^N$
- Обучение – задача оптимизации  $\min_{\theta} \frac{1}{N} \sum_{n=1}^N \Psi(x_n, y_n | \theta)$
- Метод максимального правдоподобия  
$$\Psi(x_n, y_n | \theta) = -\log P(y | x_n, \theta), \quad P = \frac{1}{Z(x_n, \theta)} \exp(F(y | x_n, \theta))$$
- $Z$  – нормировочная константа  
$$Z(x_n, \theta) = \sum_y \exp(F(y | x_n, \theta))$$
- Сумма экспоненциального числа слагаемых
- Используем Алгоритм!

# Обучение CRF – максимальное правдоподобие

- $Z$  – нормировочная константа

$$Z(\mathbf{x}_n, \boldsymbol{\theta}) = \sum_{\mathbf{y}} \exp(F(\mathbf{y} \mid \mathbf{x}_n, \boldsymbol{\theta}))$$

- Функция  $F$  обладает структурой:

$$\exp(F(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})) = \prod_{i=1}^T \exp(\theta_i(y_i \mid x_i)) \prod_{i=1}^{T-1} \exp(\theta_{i,i+1}(y_i, y_{i+1}))$$

- **Алгоритм динамического программирования (sum-product)**

- $V_i(y_i)$  – сумма произведений  $Z$  для слагаемых  $\leq i$

- Проход вперёд:

$$V_{i+1}(y_{i+1}) = \exp(\theta_{i+1}(y_{i+1} \mid x_{i+1})) \sum_{y_i} \left( \exp(\theta_{i,i+1}(y_i, y_{i+1})) V_i(y_i) \right)$$

- Ответ:  $Z = \sum_{y_T} V_T(y_T)$

# Обучение CRF – вариационный вывод

- Вар. вывод – один из основных методов обучения байесовских вероятностных моделей

- Приближение  $P$  простым распределением  $Q$ :  $Q(\mathbf{y}) = \prod_{i=1}^T q_i(y_i)$   
$$\text{KL}(Q \parallel P) = - \sum_{\mathbf{y}} Q(\mathbf{y}) \log \frac{P(\mathbf{y})}{Q(\mathbf{y})} \rightarrow \min_Q$$

- Используем структуру модели

$$P(\mathbf{y}) \propto \exp(F(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})) = \prod_{i=1}^T \exp(\theta_i(y_i \mid x_i)) \prod_{i=1}^{T-1} \exp(\theta_{i,i+1}(y_i, y_{i+1}))$$

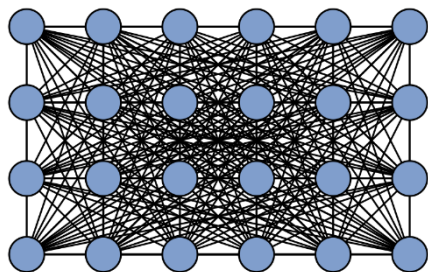
- Формулы пересчёта:

$$q_i(y_i) \propto \exp\left(\theta_i(y_i) + \sum_{y_{i-1}} \theta_{i-1,i}(y_{i-1}, y_i) q_{i-1}(y_{i-1}) + \sum_{y_{i+1}} \theta_{i,i+1}(y_i, y_{i+1}) q_{i+1}(y_{i+1})\right)$$

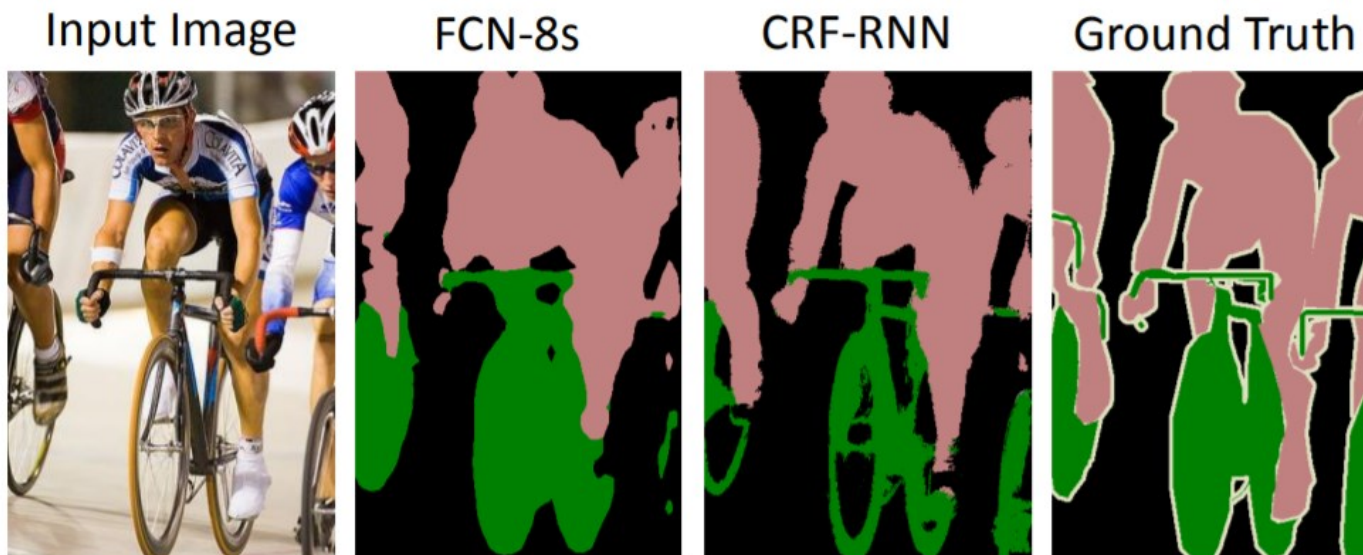
# Пример: сегментация изображений

[Zheng et al., ICCV 2015]

- Вариационных вывод над полно-связной CRF



- Все операции с использованием свёрток





# Пример: подавление шума

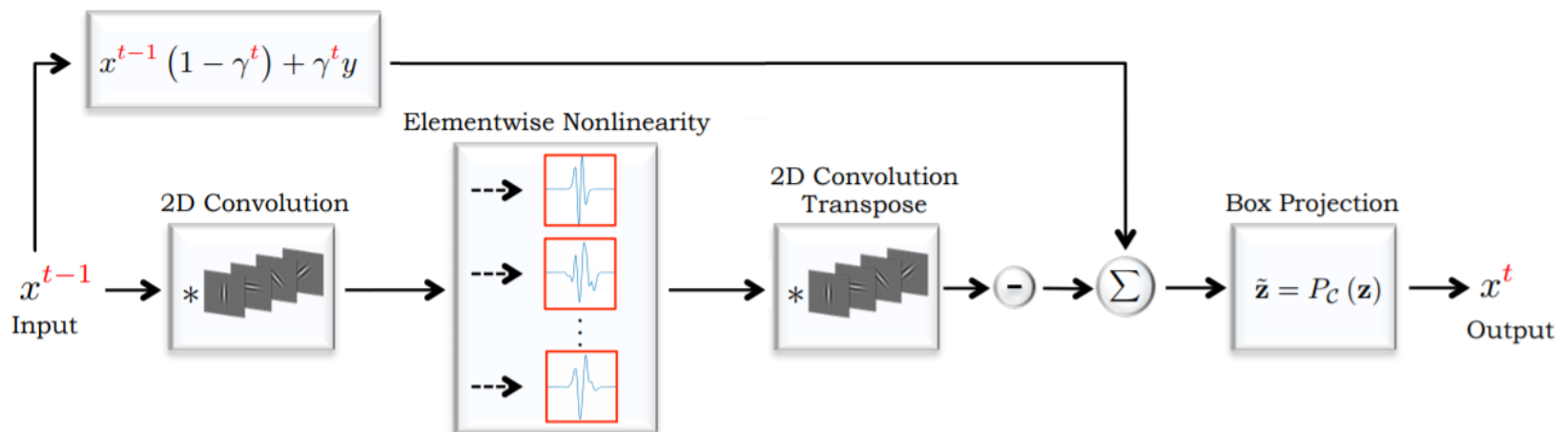
[Lefkimmatis, CVPR 2017]

- Формулировка задачи ( $\mathbf{x}$  – ответ,  $\mathbf{y}$  – шумное изображение):

$$\mathbf{x}^* = \arg \min_{a \leq x_n \leq b} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{r=1}^R \phi(\mathbf{L}_r \mathbf{x})$$

- Алгоритм решения – Proximal Gradient

$$\mathbf{x}^t = P_C \left( \mathbf{x}^{t-1} (1 - \gamma^t) + \gamma^t \mathbf{y} - \alpha^t \sum_{r=1}^R \mathbf{L}_r^T \psi(\mathbf{L}_r \mathbf{x}^{t-1}) \right)$$



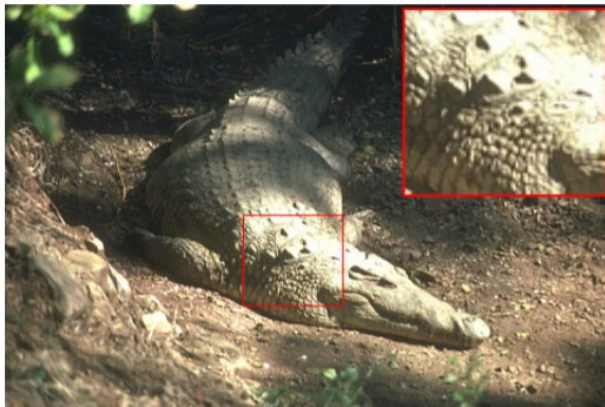
# Пример: подавление шума

[Lefkimmatis, CVPR 2017]

- Формулировка задачи ( $\mathbf{x}$  – ответ,  $\mathbf{y}$  – шумное изображение):

$$\mathbf{x}^* = \arg \min_{a \leq x_n \leq b} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{r=1}^R \phi(\mathbf{L}_r \mathbf{x})$$

- Результаты:



Оригинал



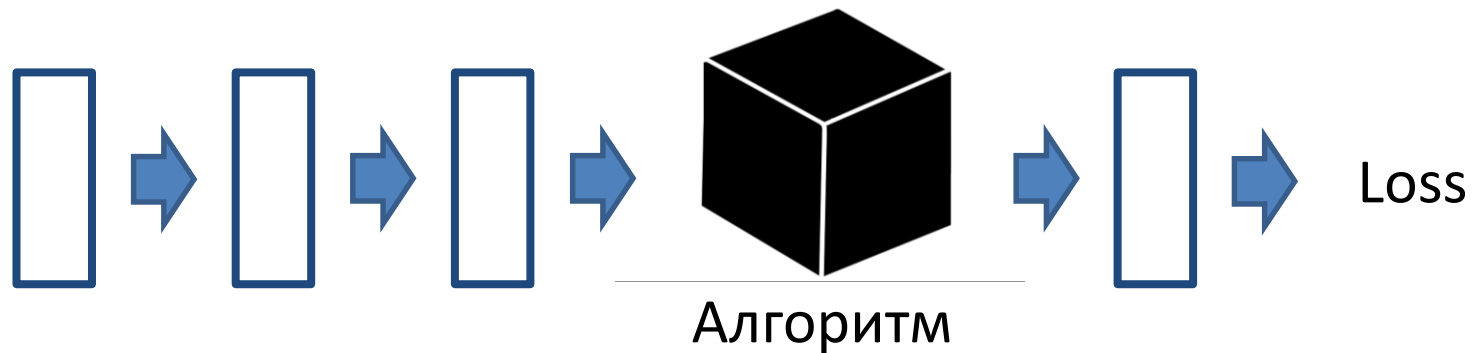
Оригинал + шум



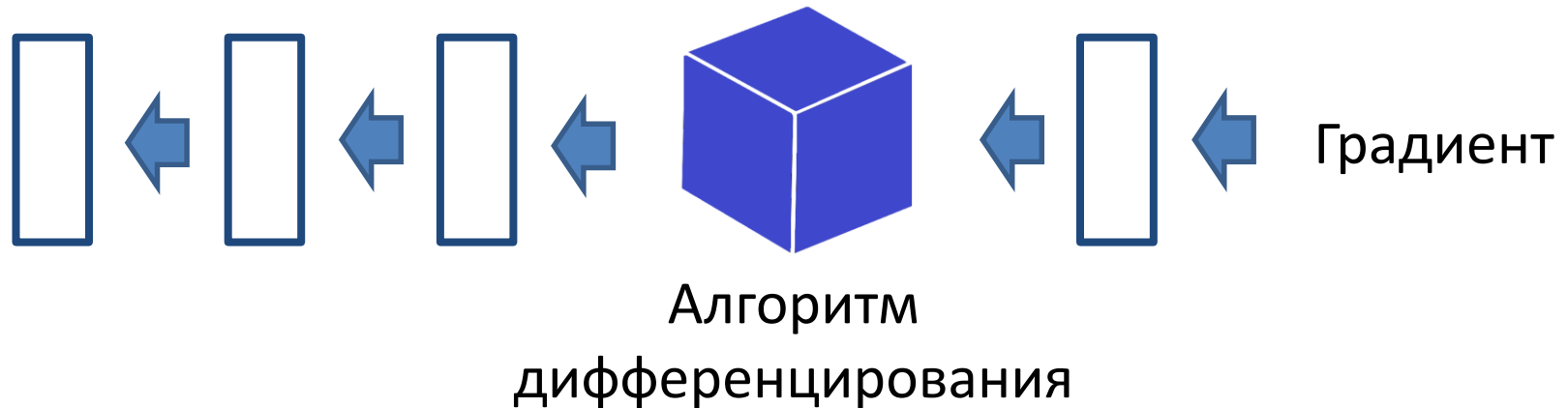
Результат

# Дифференцирование по входу

Нейросеть: проход вперёд



Для прохода назад нужен другой алгоритм



# Пример: Gaussian MRF

[Chandra&Kokkinos, ECCV 2016]

- Непрерывный аналог CRF

$$F(\mathbf{y}) = -\frac{1}{2}\mathbf{y}^T (A + \lambda I)\mathbf{y} + \mathbf{b}^T \mathbf{y} \rightarrow \max_{\mathbf{y}}$$

- $\mathbf{y}$  – переменные,  $A$ ,  $\mathbf{b}$  – параметры (выходы нейросети)

- Алгоритм предсказания – СЛАУ:  $\mathbf{y} = (A + \lambda I)^{-1}\mathbf{b}$

- Задача дифференцирования

- Вход: параметры  $A$ ,  $\mathbf{b}$ , решение  $\mathbf{y}$ , градиент  $\frac{d\Psi}{d\mathbf{y}}$

- Найти: градиенты  $\frac{d\Psi}{d\mathbf{b}}$  и  $\frac{d\Psi}{dA}$

- Дифференцируем линейный слой (СЛАУ)

$$\frac{d\Psi}{d\mathbf{b}} = (A + \lambda I)^{-T} \frac{d\Psi}{d\mathbf{y}}$$

# Пример: Gaussian MRF

[Chandra&Kokkinos, ECCV 2016]

- Непрерывный аналог CRF

$$F(\mathbf{y}) = -\frac{1}{2}\mathbf{y}^T (A + \lambda I)\mathbf{y} + \mathbf{b}^T \mathbf{y} \rightarrow \max_{\mathbf{y}}$$

- $\mathbf{y}$  – переменные,  $A$ ,  $\mathbf{b}$  – параметры (выходы нейросети)

- Алгоритм предсказания – СЛАУ:  $\mathbf{y} = (A + \lambda I)^{-1}\mathbf{b}$

- Задача дифференцирования

- Вход: параметры  $A$ ,  $\mathbf{b}$ , решение  $\mathbf{y}$ , градиент  $\frac{d\Psi}{d\mathbf{y}}$
- Найти: градиенты  $\frac{d\Psi}{d\mathbf{b}}$  и  $\frac{d\Psi}{dA}$

- Производная по  $A$ :  $\frac{d\Psi}{dA} = -\frac{d\Psi}{d\mathbf{b}}\mathbf{y}^T$

# Примеры дифференцирования


- SVD разложение  $X = U\Sigma V^T$  [Ionescu et al., ICCV 2015]

$$\frac{\partial L \circ f}{\partial X} = DV^T + U \left( \frac{\partial L}{\partial \Sigma} - U^T D \right)_{diag} V^T + 2U\Sigma \left( K^T \circ \left( V^T \left( \frac{\partial L}{\partial V} - VD^T U\Sigma \right) \right) \right)_{sym} V^T$$

$$K_{ij} = \begin{cases} \frac{1}{\sigma_i^2 - \sigma_j^2}, & i \neq j \\ 0, & i = j \end{cases} \quad D = \left( \frac{\partial L}{\partial U} \right)_1 \Sigma_n^{-1} - U_2 \left( \frac{\partial L}{\partial U} \right)_2^T U_1 \Sigma_n^{-1}$$

- Дискретная! оптимизация [Djolonga&Krause, NIPS 2017]
  - Используется эквивалентность дискретной минимизации субмодулярных функций непрерывным релаксациям
- Решение дифференц-х уравнений [Chen et al., NeurIPS 2018]
  - Вычисление градиента – решение другого уравнения

# Неявное дифференцирование

- Алгоритм = неявный слой
  - Явный слой  $y := f(x)$
  - Неявный слой  $h(x, y(x)) = 0$  (система уравнений)
  - Алгоритм решает уравнение
- Как дифференцировать?
  1.  $\frac{d}{dx} h(x, y(x)) = 0$
  2.  $\frac{\partial h}{\partial x} + \frac{\partial h}{\partial y} \frac{dy}{dx} = 0$   Автоматическое дифференцирование
  3.  $\frac{dy}{dx} := -\left(\frac{\partial h}{\partial y}\right)^{-1} \frac{\partial h}{\partial x}$ 
    - А. Якобиан обратим? (+ регуляризация)
    - В. Неявные методы для решения

# Примеры неявных слоев

- OptNet: Differentiable Optimization as a Layer [Amos&Kolter, 2017]
- Пример: квадратичное программирование (QP)

$$\hat{\mathbf{y}}(\mathbf{x}) := \arg \min_{\mathbf{y}} \frac{1}{2} \mathbf{y}^T Q(\mathbf{x}) \mathbf{y} + \mathbf{y}^T \mathbf{c}(\mathbf{x})$$
$$\text{s.t. } A(\mathbf{x}) \mathbf{y} = \mathbf{b}(\mathbf{x})$$
$$G(\mathbf{x}) \mathbf{y} \leq \mathbf{s}(\mathbf{x})$$

Пример: подавление шума

$$\mathbf{x}^* = \arg \min_{\mathbf{a} \leq x_n \leq b} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{r=1}^R \phi(\mathbf{L}_r \mathbf{x})$$

- Уравнения: условия ККТ (Каруша-Куна-Таккера)

$$Q\mathbf{y} + \mathbf{c} + A^T \boldsymbol{\nu} + G^T \boldsymbol{\lambda} = 0$$
$$A\mathbf{y} - \mathbf{b} = 0$$
$$\text{diag}(\boldsymbol{\lambda})(G\mathbf{y} - \mathbf{s}) = 0$$

Еще примеры:

- SATNet [Wang et al., 2019]
  - Релаксация дискретной оптимизации
- Deep equilibrium models [Bai et al, 2019]
  - Повторение слоев ResNet/Transformer
- Движок физики в симуляторе  
[de Avila Belbute-Peres et al., 2018]
  - Законы физики = уравнения



# Заключение

- Разные способы встраивать алгоритмы в нейросети
  - Структурный пулинг
  - Итерации алгоритма => слои нейросети
  - Прямое (аналитическое ) дифференцирование по входу
- Расширение библиотеки слоев
- Использование алгоритмов позволяет встраивать знания о задаче в нейросети