Глубинное обучение
Лекция 9: Дифференцируемое программирование

Лектор: Антон Осокин

ФКН ВШЭ, 2019

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

Объявления

- Открыт набор в группу байесовский методов
  https://bayesgroup.ru/admission/
  – Байесовские методы (Deep)
  – Deep learning
  – Deep learning for CV
  – Deep learning for NLP
  – Optimization

- Открыт набор на летнюю школу
  http://deepbayes.ru/

План лекции

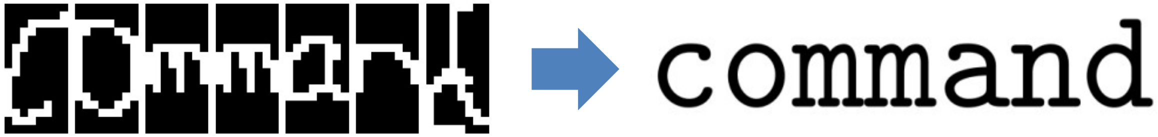
- Мотивация: зачем встраивать алгоритмы в нейросети?
- Алгоритмы в нейросетях
  - Структурный пулинг = комбинаторная оптимизация
  - Итерации алгоритмов = слои нейросетей
  - Дифференцирование по входу алгоритма
- Недифференцируемые модели
  - Что делать?

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
- Deep learning meets computer science!

Задача: распознавание рукописных символы

- Модельная задача – распознавание рукописных символов



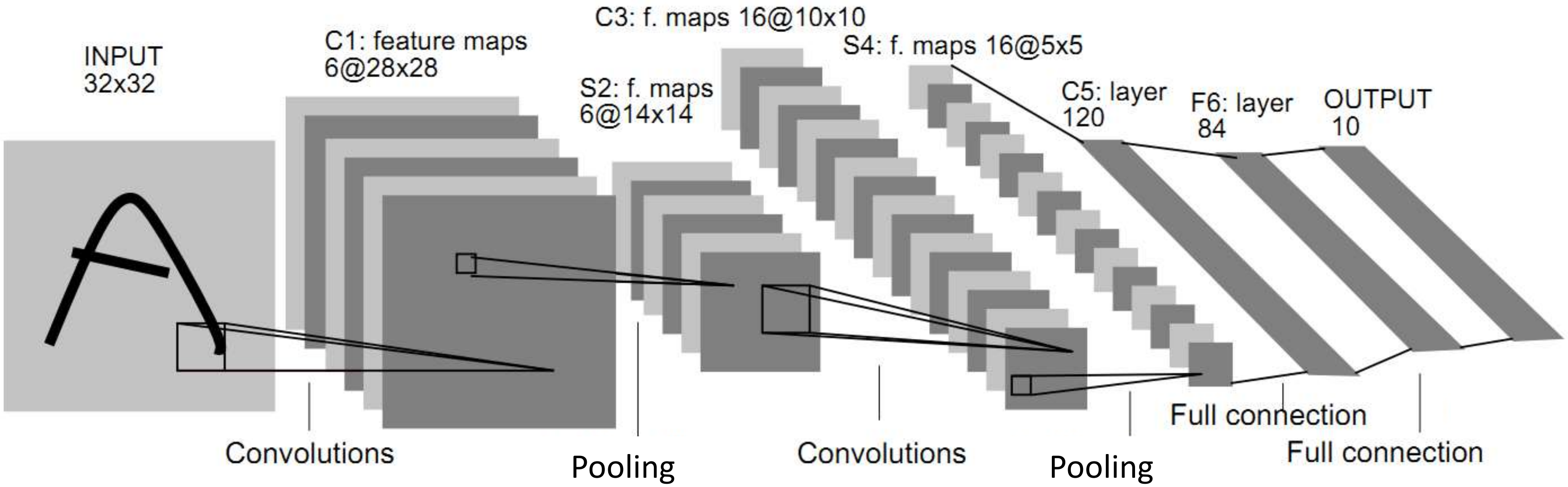
- A MNIST для structured prediction

- Простая задача => очень много методов применимо

- Стандартные алгоритмы:
  динамическое программирование, СЛАУ

Структура нейросети

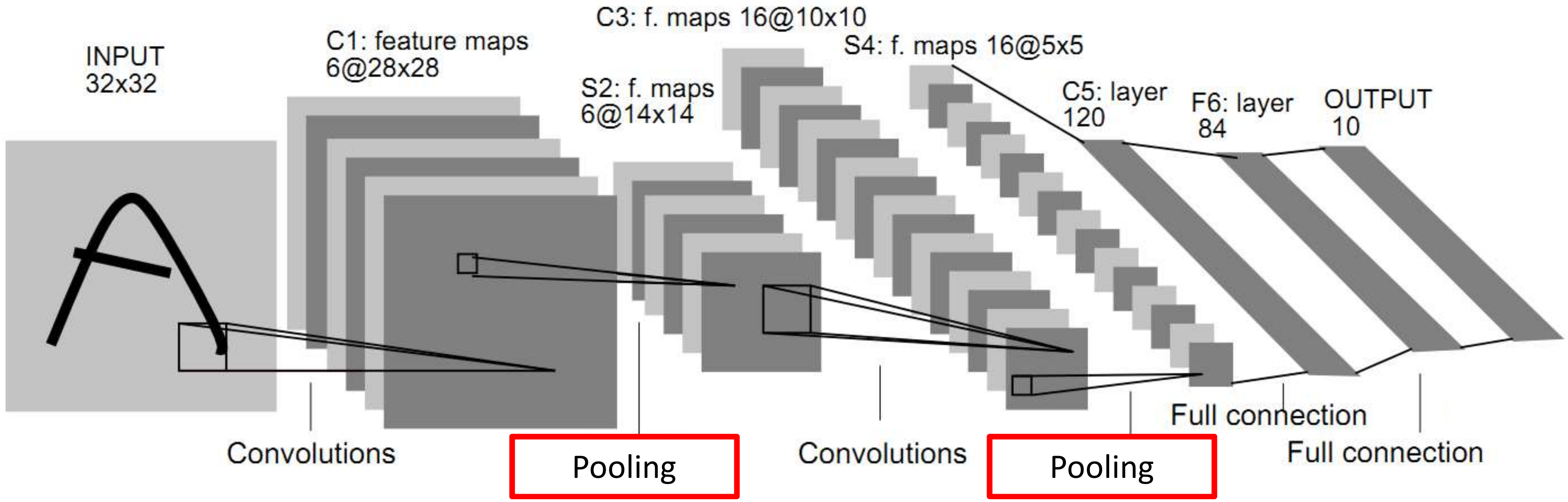
- Нейросеть для распознавания одного символа:



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

Пулинг для выбора активаций

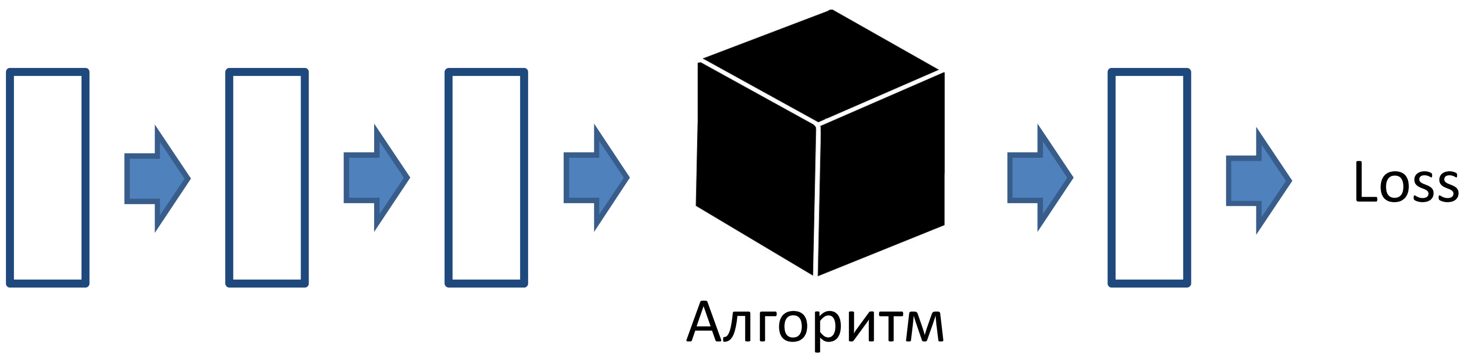
- Нейросеть для распознавания одного символа:



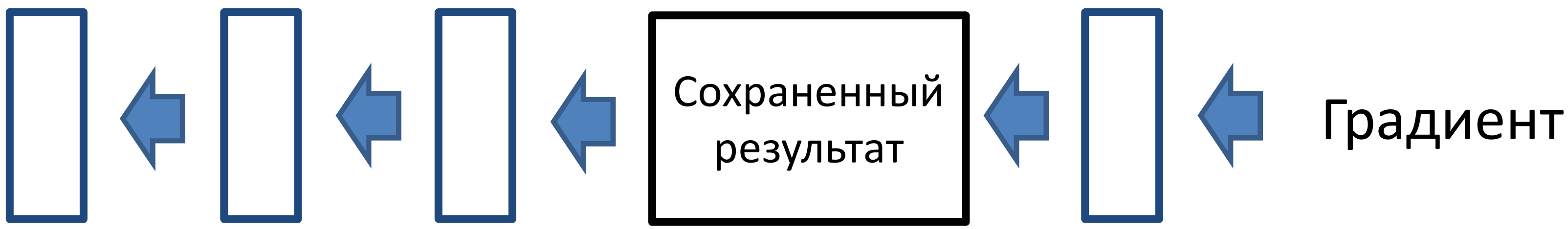
- Пулинг – процедура агрегирования активаций (max, sum)
- Алгоритмы для пулинга: квантиль, сортировка
- И это дифференцируемо?
    - «дифференцируемо» в смысле нейросетей

Структурный пулинг = комбинаторная оптимизация

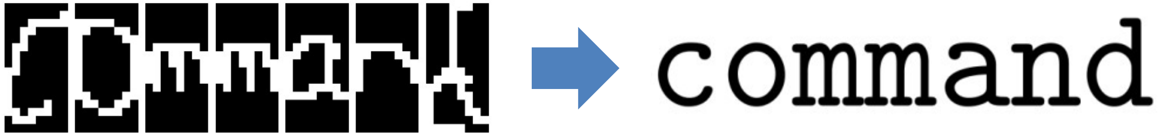
Нейросеть: проход вперёд



Для прохода назад нужен только результат алгоритма



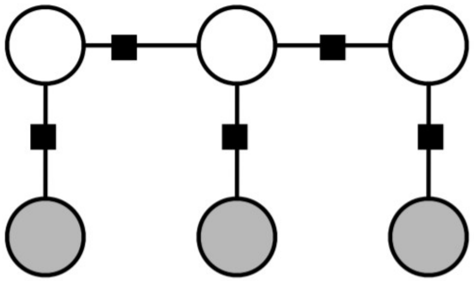
Conditional Random Field (CRF)



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

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

- Здесь ***y*** – метки символов, ***x*** – изображения символов
  θ - 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

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

- Здесь ***y*** – метки символов, ***x*** – изображения символов
  θ - 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:

$$V_1(y_1) = \theta_1(y_1 \mid x_1)$$
$$V_{i+1}(y_{i+1}) = \theta_{i+1}(y_{i+1} \mid x_{i+1}) + \max_{y_i}\Big(\theta_{i,i+1}(y_i, y_{i+1}) + V_i(y_i)\Big)$$

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

Обучение CRF – структурный SVM

- Обучение по размеченной выборке $\left\{\boldsymbol{x}_n, \boldsymbol{y}_n\right\}_{n=1}^{N}$
- Обучение – задача оптимизации $\min\limits_{\boldsymbol{\theta}} \frac{1}{N} \sum\limits_{n=1}^{N} \Psi(\boldsymbol{x}_n, \boldsymbol{y}_n \mid \boldsymbol{\theta})$
- SSVM - обобщение метода опорных векторов

$$\Psi(\boldsymbol{x}_n, \boldsymbol{y}_n \mid \boldsymbol{\theta}) = \max_{\boldsymbol{y}} \Big[F(\boldsymbol{y} \mid \boldsymbol{x}_n, \boldsymbol{\theta}) + \Delta(\boldsymbol{y}, \boldsymbol{y}_n) \Big] - F(\boldsymbol{y}_n \mid \boldsymbol{x}_n, \boldsymbol{\theta})$$

- 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)

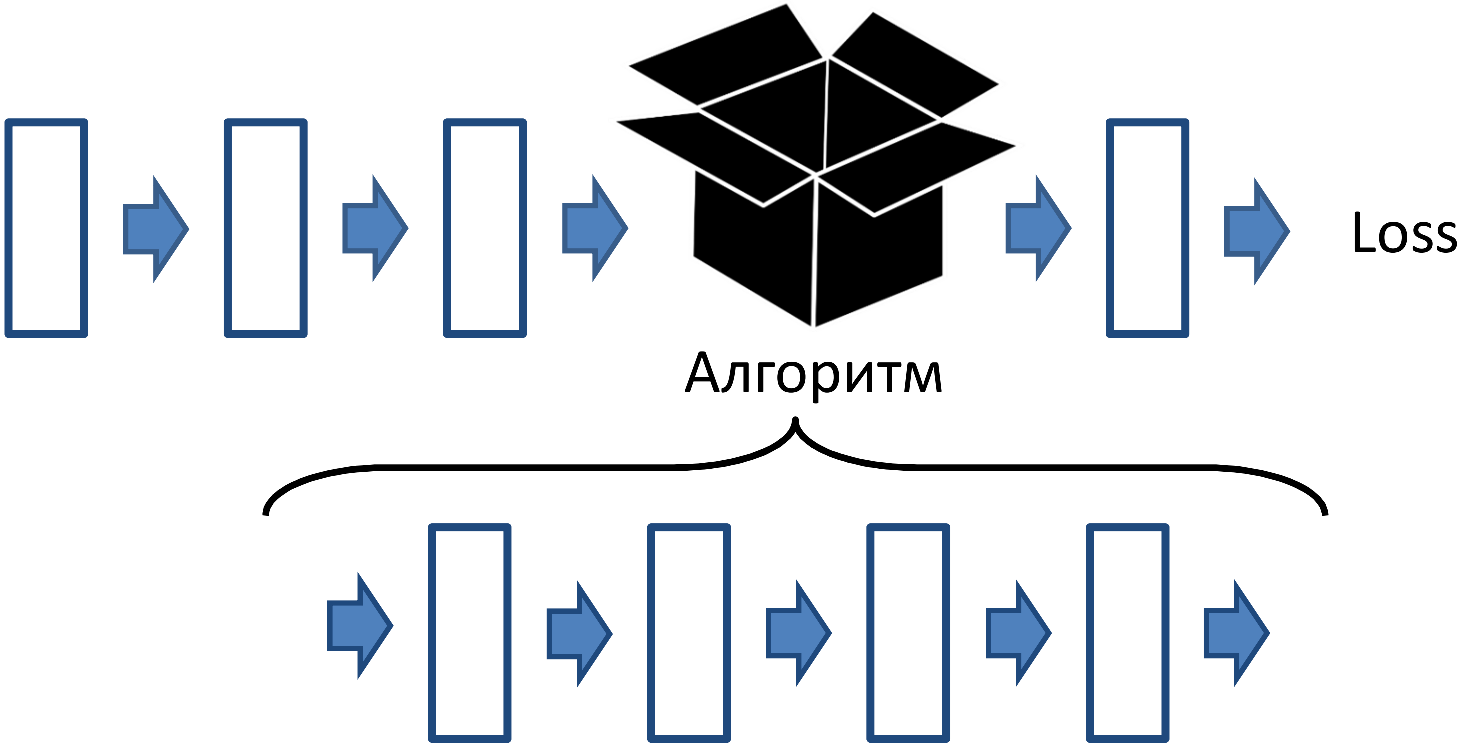
- Детектирование нескольких объектов (Vu et al., ICCV 2015)

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

female/indoor/portrait sky/plant life/tree water/animals/sea animals/dog/indoor indoor/flower/plant life

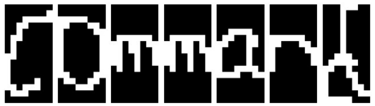
Итерации алгоритма как слои сети

Нейросеть: проход вперёд



Проход назад – обычный back propagation

Обучение CRF – максимальное правдоподобие

 → command

- Обучение по размеченной выборке $\{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^{N}$
- Обучение – задача оптимизации $\min\limits_{\boldsymbol{\theta}} \frac{1}{N} \sum\limits_{n=1}^{N} \Psi(\boldsymbol{x}_n, \boldsymbol{y}_n \mid \boldsymbol{\theta})$
- Метод максимального правдоподобия

$$\Psi(\boldsymbol{x}_n, \boldsymbol{y}_n \mid \boldsymbol{\theta}) = -\log P(\boldsymbol{y} \mid \boldsymbol{x}_n, \boldsymbol{\theta}), \quad P = \frac{1}{Z(\boldsymbol{x}_n, \boldsymbol{\theta})} \exp(F(\boldsymbol{y} \mid \boldsymbol{x}_n, \boldsymbol{\theta}))$$

- Z – нормировочная константа

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

- Сумма экспоненциального числа слагаемых
- Используем Алгоритм!

Обучение CRF – максимальное правдоподобие

- Z – нормировочная константа

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

- Функция *F* обладает структурой:

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

- Алгоритм динамического программирования (sum-product)
  - $V_i(y_i)$ – сумма произведений *Z* для слагаемых ≤ *i*
  - Проход вперёд:

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

  - 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*: $Q(\boldsymbol{y}) = \prod_{i=1}^{T} q_i(y_i)$

$$\mathrm{KL}(Q \,||\, P) = -\sum_{\boldsymbol{y}} Q(\boldsymbol{y}) \log \frac{P(\boldsymbol{y})}{Q(\boldsymbol{y})} \to \min_{Q}$$

- Используем структуру модели

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

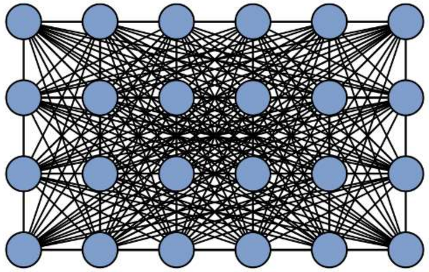
- Формулы пересчёта:

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

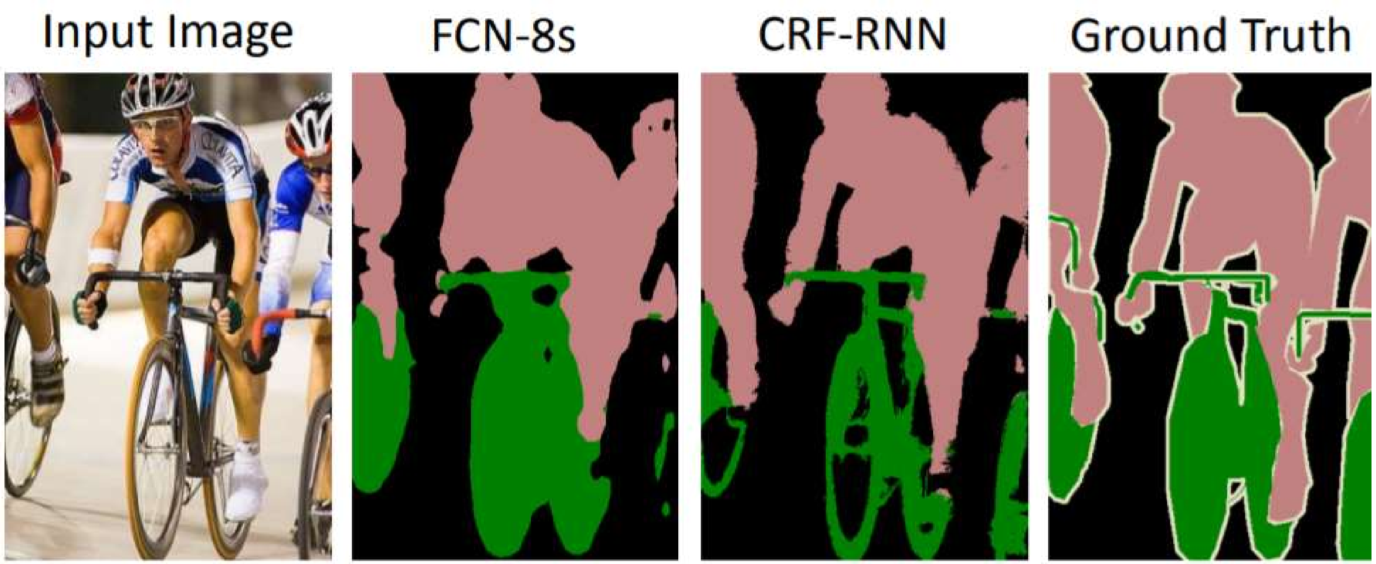
Пример: сегментация изображений

[Zheng et al., ICCV 2015]

- Вариационных вывод над полно-связной CRF



- Все операции с использованием свёрток



Пример: подавление шума

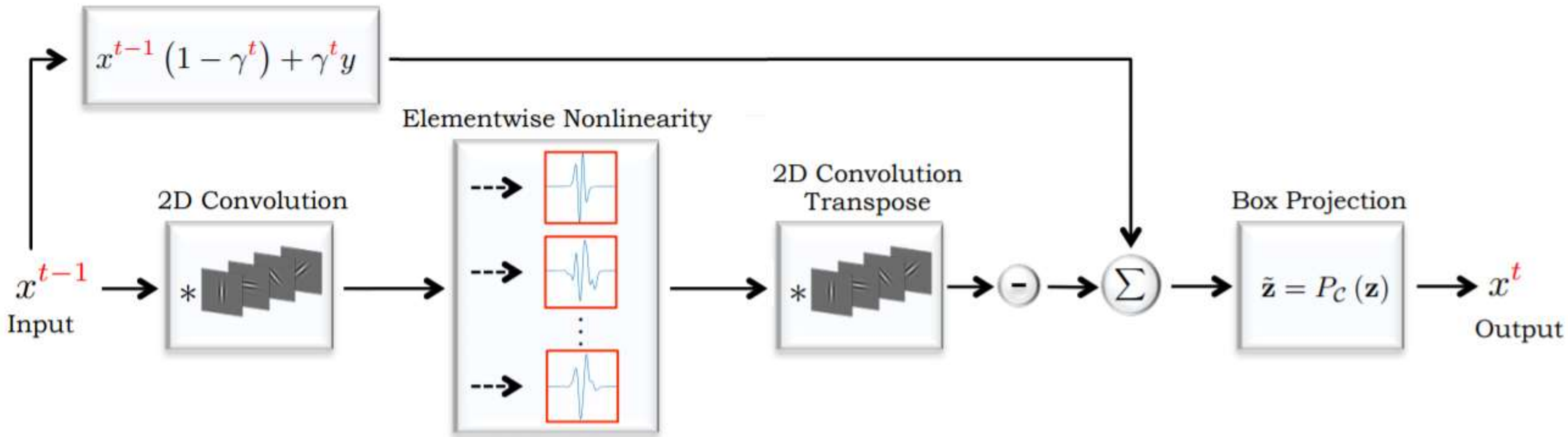
[Lefkimmiatis, CVPR 2017]

- Формулировка задачи (***x*** – ответ, ***y*** – шумное изображение):

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

- Алгоритм решения – Proximal Gradient

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



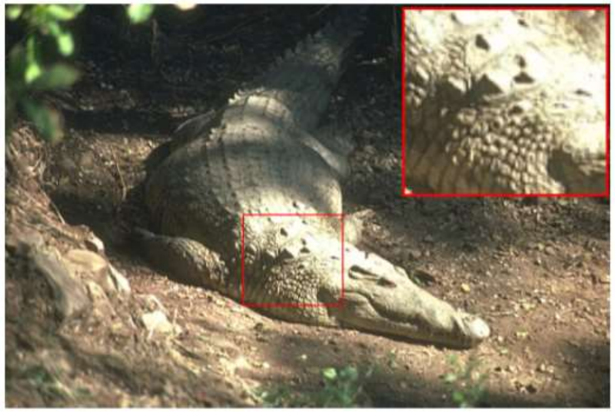
Пример: подавление шума

[Lefkimmiatis, CVPR 2017]

- Формулировка задачи (***x*** – ответ, ***y*** – шумное изображение):

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

- Результаты:



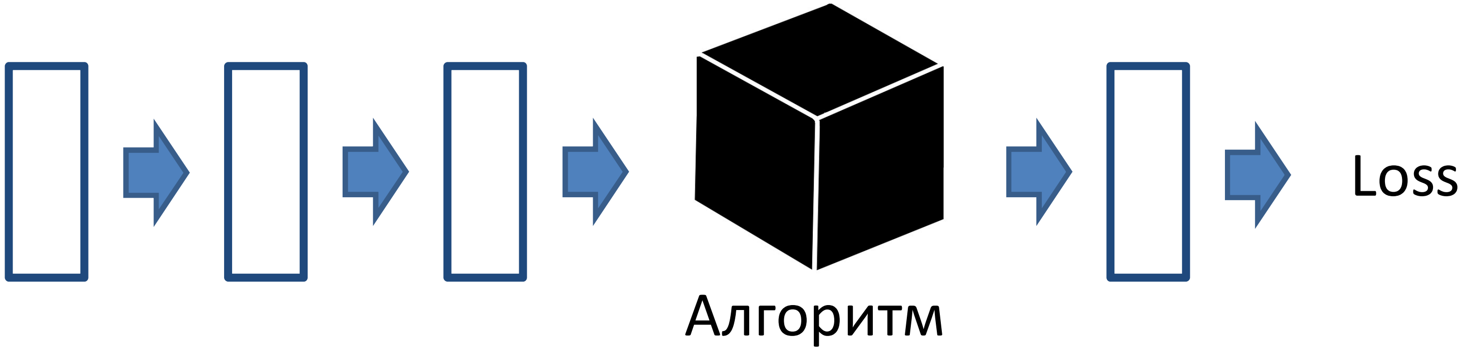
Оригинал

Оригинал + шум

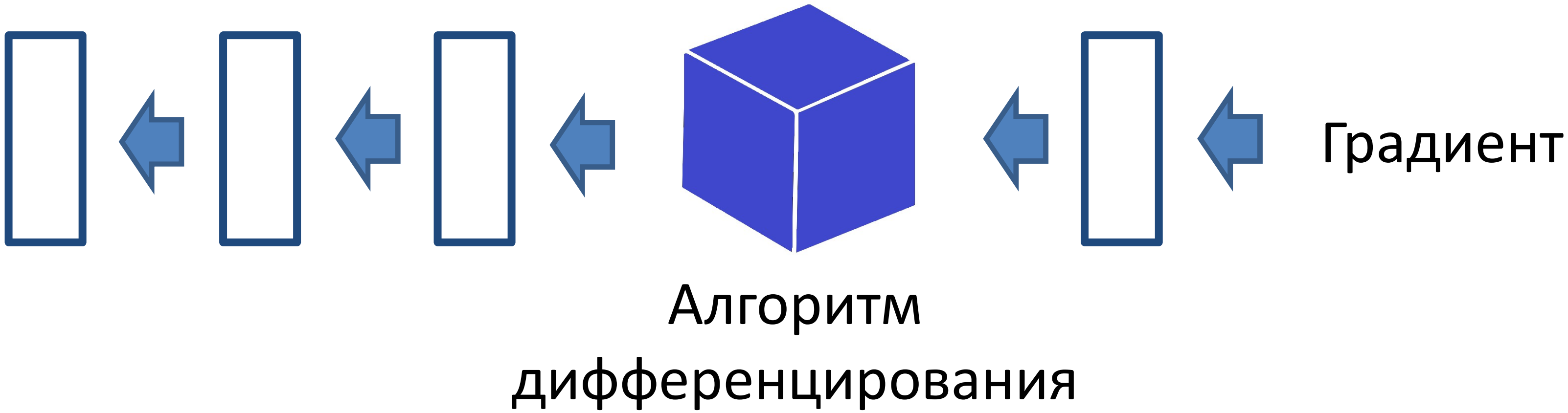
Результат

Дифференцирование по входу

Нейросеть: проход вперёд



Для прохода назад нужен другой алгоритм



Пример: Gausian MRF

[Chandra&Kokkinos, ECCV 2016]

- Непрерывный аналог CRF

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

- ***y*** – переменные, *A*, ***b*** – параметры (выходы нейросети)

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

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

- Дифференцируем линейный слой (СЛАУ)

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

Пример: Gausian MRF

[Chandra&Kokkinos, ECCV 2016]

- Непрерывный аналог CRF

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

- ***y*** – переменные, *A*, ***b*** – параметры (выходы нейросети)
- Алгоритм предсказания – СЛАУ: $\boldsymbol{y} = (A + \lambda I)^{-1}\boldsymbol{b}$
- Задача дифференцирования
  - Вход: параметры A, ***b***, решение ***y***, градиент $\frac{d\Psi}{d\boldsymbol{y}}$
  - Найти: градиенты $\frac{d\Psi}{d\boldsymbol{b}}$ и $\frac{d\Psi}{dA}$
- Производная по *A*: $\frac{d\Psi}{dA} = -\frac{d\Psi}{d\boldsymbol{b}}\boldsymbol{y}^T$

Примеры дифференцирования

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

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

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

- Дискретная! оптимизация [Djolonga&Krause, NIPS 2017]
  - Используется эквивалентность дискретной минимизации субмодулярных функций непрерывным релаксациям

- Решение дифференциальных уравнений [Chen et al., NIPS 2018]
  - Вычисление градиента – решение другого уравнения

Недифференцируемые модели?

- Часто недифференцируемые функции – backpropable (“нейро-дифференцируемые“)
  - Примеры: max, ReLu, медиана
- Совсем-недифференцируемые функции
  - Кусочно-постоянные функции
    - argmax
    - f(x) = 0 if x <0 else 1
  - Сложные индексы
    - Позиция прямоугольник на изображении
  - Ответы внешних систем:
    - Программа, среда, человек

Что делать?

- Игнорировать ☺
  - max, комбинаторный пулинг

- Сглаживать через стохастичность (http://deepbayes.ru/)
  - Стохастические активации

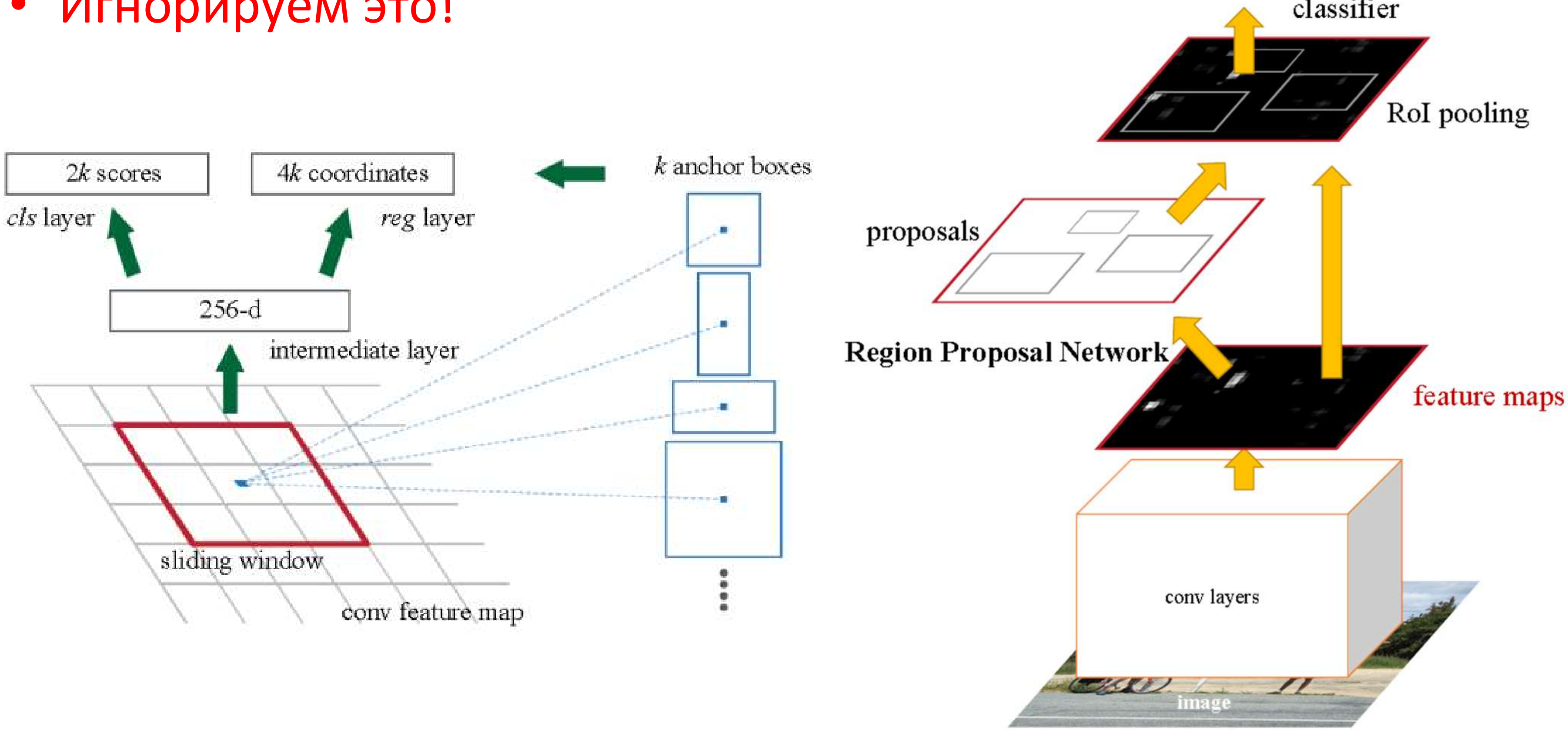
Летняя школа в 2019!

- Другие способы сглаживания

Детектор Faster R-CNN

[Ren et al., 2015]

- Одна сеть выдаёт гипотезы объектов (proposals)
- Вторая сеть классифицирует гипотезы
- RoI pooling – недифференцируемый по координатам
- Игнорируем это!



Стохастические активации

- Что это такое?
  - θ(*x*) – дифференцируемая функция
  - Распределение *p*(*w* | θ(*x*))
  - Лосс *L*(*w*) – не дифференцируемый?
- Проход вперёд:
  - Вычисление θ(*x*) – параметры распределения
  - Сэмплирование *w* из *p*(*w* | θ(*x*))
  - Вычисление *L*(*w*)
- Функция $\mathbb{E}_{w \sim p(w|\theta(x))} L(w)$ – часто дифференцируемая по θ и *x*
- Градиент получается из log-derivative trick

$$\nabla_\theta = \mathbb{E}_{w \sim p(w|\theta)} \nabla_\theta [\log p(w|\theta(x))] L(w) \qquad \nabla_\theta [\log p(w|\theta)] = \frac{\nabla_\theta [p(w|\theta)]}{p(w|\theta)}$$

- Большая дисперсия!

Обучение: дифференцируемый лосс

- Простой случай:
  - Лосс $L(w)$ – дифференцируемый
  - Распределение $p(w \mid \theta)$ – «хорошее»

- Репараметризация (если возможна) – самое лучшее решение!
  - Разделение случайности и параметров
  - Представим распределение $p(w \mid \theta)$ как $g(\theta, \varepsilon), \ \varepsilon \sim r(\varepsilon)$

    $$z = \mu_\theta(x) + \sigma_\theta(x)\varepsilon, \ \varepsilon \sim r(\varepsilon)$$

    g – детерминированная функция

    ε – шум
  - Тогда градиент легко оценить:

    $$\nabla_\theta = \nabla_\theta \int p(w|\theta)L(w)dw = \int r(\varepsilon)\nabla_\theta L(g(\theta,\varepsilon))d\varepsilon$$

  - Дисперсия градиента сильно уменьшается

Какие распределения можно репараметризовать?

$p(x\|y)$	$r(\epsilon)$	$g(\epsilon, y)$
$\mathcal{N}(x\|\mu, \sigma^2)$	$\mathcal{N}(\epsilon\|0, 1)$	$x = \sigma\epsilon + \mu$
$\mathcal{G}(x\|1, \beta)$	$\mathcal{G}(\epsilon\|1, 1)$	$x = \beta\epsilon$
$\mathcal{E}(x\|\lambda)$	$\mathcal{U}(\epsilon\|0, 1)$	$x = -\frac{\log \epsilon}{\lambda}$
$\mathcal{N}(x\|\mu, \Sigma)$	$\mathcal{N}(\epsilon\|0, I)$	$x = A\epsilon + \mu, \text{where } AA^T = \Sigma$

Slide credit: Dmitry Vetrov

Нельзя репараметризовать дискретные распределения!

- Категориальное распределение $z \sim \mathrm{Discrete}(\alpha_1, \dots, \alpha_L)$
- Надо для argmax!

$$z = (0, 1, 0, \dots, 0)$$

- Релаксация: Gumbel-Softmax [Jang et al., 2017; Maddison et al., 2017]

$$(z_1, \dots, z_L) \sim \mathrm{RelaxedDiscrete}(\alpha_1, \dots, \alpha_L | T)$$

$$z_i = \frac{\exp((\log \alpha_i + G_i)/T)}{\sum_{j=1}^{L} \exp((\log \alpha_j + G_j)/T)}, \; G_k \sim \mathrm{Gumbel}$$

$$G_k = -\log(-\log u_k), \; u_k \sim \mathrm{Uniform}[0, 1]$$

- $\mathrm{RelaxedDiscrete}(\alpha_1, \dots, \alpha_L | T) \xrightarrow{T \to 0} \mathrm{Discrete}(\alpha_1, \dots, \alpha_L)$

- Трюк: Straight-through estimator [Hinton et al., 2015 course] [Bengio et al. (2013)]

$$\nabla_\alpha \approx \nabla_z$$

Градиент 1 в выбранную позицию, остальные 0

Недифференцируемый лосс? Что делать?

- Модель:
  - θ(*x*) – дифференцируемая функция
  - Распределение *p*(*w* | θ(*x*))
  - Лосс *L*(*w*) недифференцируемый
- Обучение с подкреплением ☹
  - Политика *p*(*w* | θ(*x*))
  - Награда-reward: – *L*(*w*)
- Log-derivative trick лежит в основе REINFORCE, policy gradients

$$\nabla_\theta = \mathbb{E}_{w \sim p(w|\theta)} \nabla_\theta [\log p(w|\theta(x))] L(w)$$

  - Борьба с дисперсией! (всеми средствами)

Что делать с дисперсией?

- Модель:
  - θ(*x*) – дифференцируемая функция
  - Распределение *p*(*w* | θ(*x*))
  - Лосс *L*(*w*) недифференцируемый
- Градиент: $\nabla_\theta \approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta[\log p(w_i|\theta(x))]L(w_i)$
- Идея: baseline $\nabla_\theta \approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta[\log p(w_i|\theta(x))]\Big(L(w_i) - b\Big)$
  - Почему?

$$\int p(w|\theta)\nabla_\theta[\log p(w|\theta(x))]b\, dw = \int \nabla_\theta p(w|\theta) b\, dw = b\nabla_\theta \int p(w|\theta)\, dw = 0$$

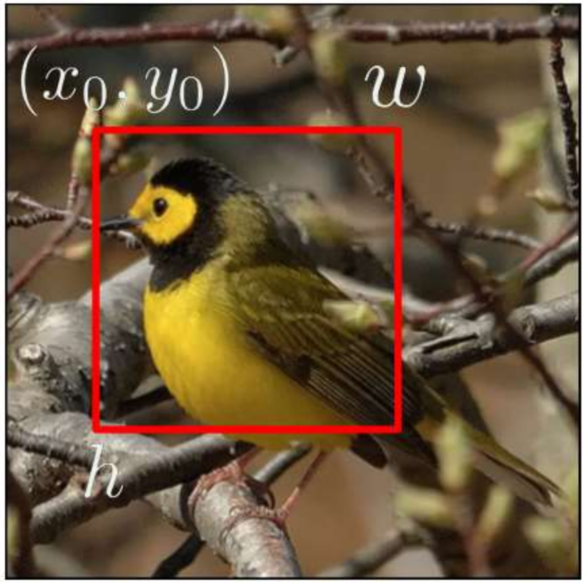
- Идея: дифференцируемый бейзлайн, зависящий от *w*
  - Компенсировать смещение репараметризацией бейзлайна

Другие способы сглаживания: Spatial Transformer

[Jaderberg et al., 2015]

- Параметрическая модель фрагмента (x_0, y_0, w, h)
- Сеть по картинке выдает параметры
- Нужно вырезать патч
- Можно обучать REINFORNCE (но работает не очень) [Mnih et al. , 2014]

- **Spatial Transformer** – лучше
  - **Идея**: билинейная интерполяция дифференцируема
  - «Локальное внимание»
  - Расширение области определения



Slide credit: Michael Figurnov

Дифференцируемая интерполяция

[Jaderberg et al., 2015]

- Билинейная интерполяция вычисляет цвет в нецелой точке (x, y)
- U – картинка, v – значение в (x, y)
- Основное наблюдение:

$$v = \sum_{i=1}^{H} \sum_{j=1}^{W} U_{ij} \max(0, 1 - |x - i|) \max(0, 1 - |y - j|)$$

- Градиенты

$$\frac{dv}{dU_{ij}} = \max(0, 1 - |x - i|) \max(0, 1 - |y - j|)$$

$$\frac{dv}{dx} = \sum_{i=1}^{H} \sum_{j=1}^{W} U_{ij} \max(0, 1 - |y - j|) \begin{cases} 0, & \text{if } |x - i| \geq 1 \\ 1, & \text{if } x < i \\ -1, & \text{if } x \geq i \end{cases}$$

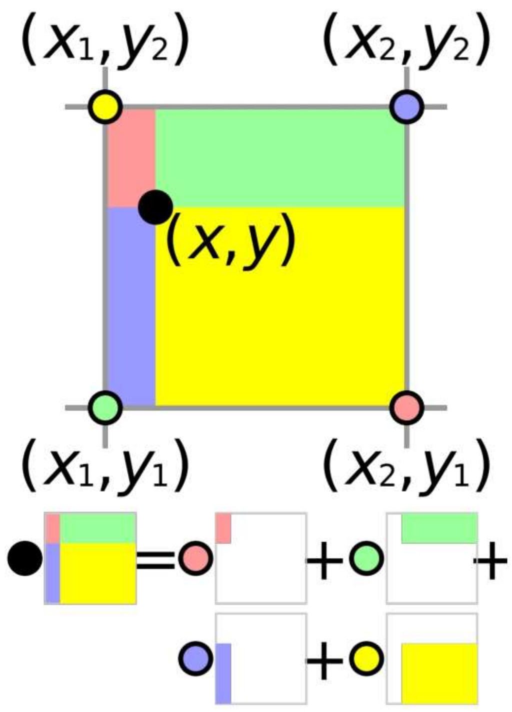


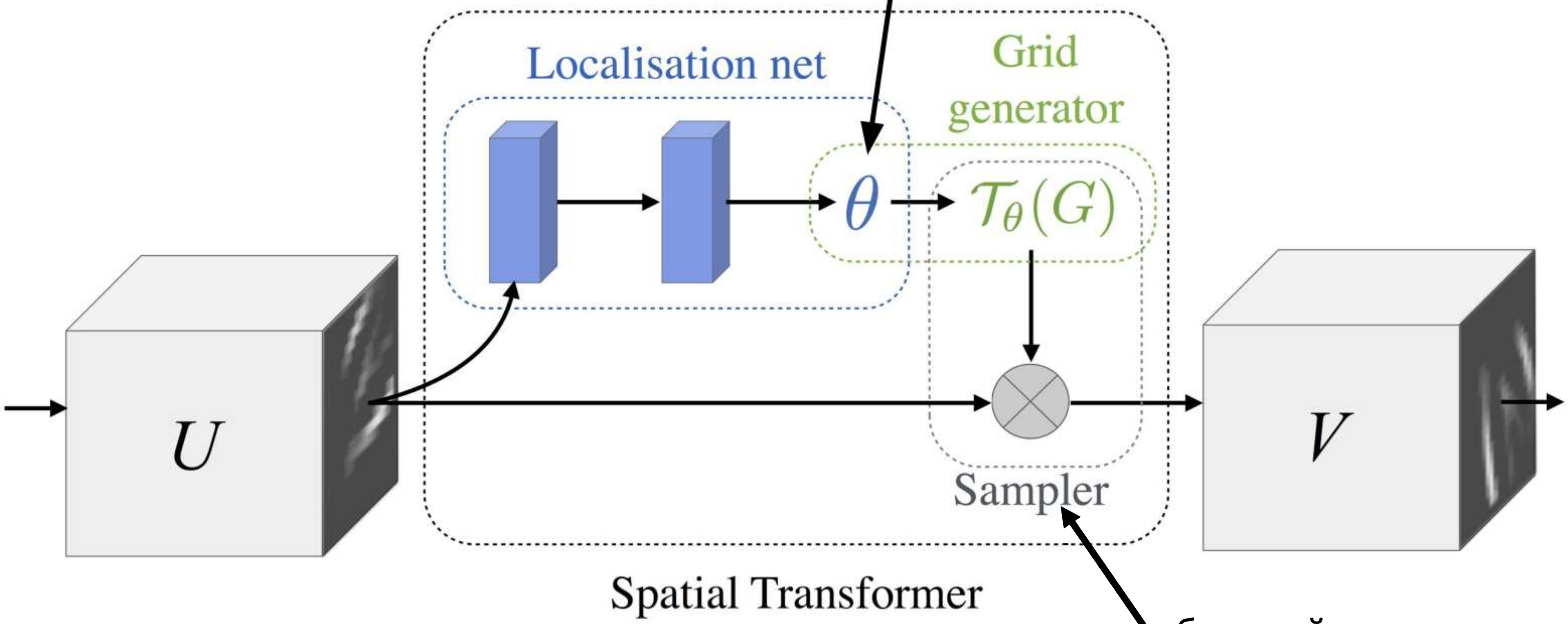
Image credit: wikipedia

Slide credit: Michael Figurnov

Другие способы сглаживания: Spatial Transformer

[Jaderberg et al., 2015]

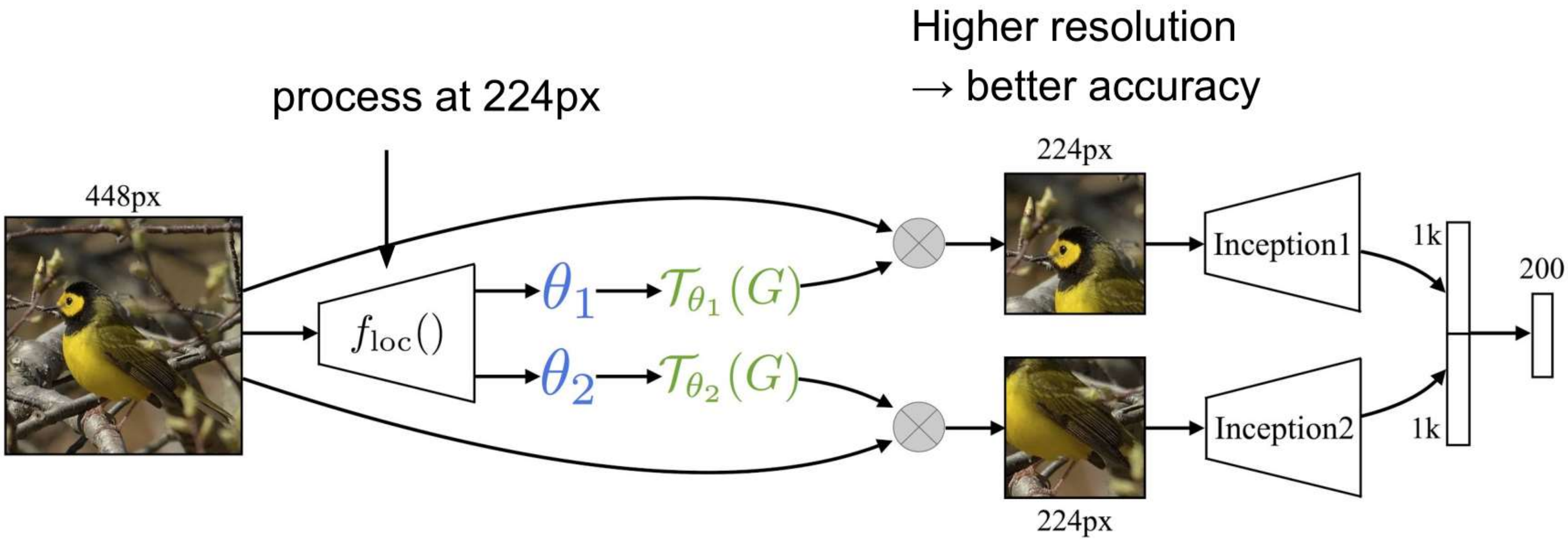
$$(x_0, y_0, w, h) \in \mathbb{R}^4$$



билинейная
интерполяция

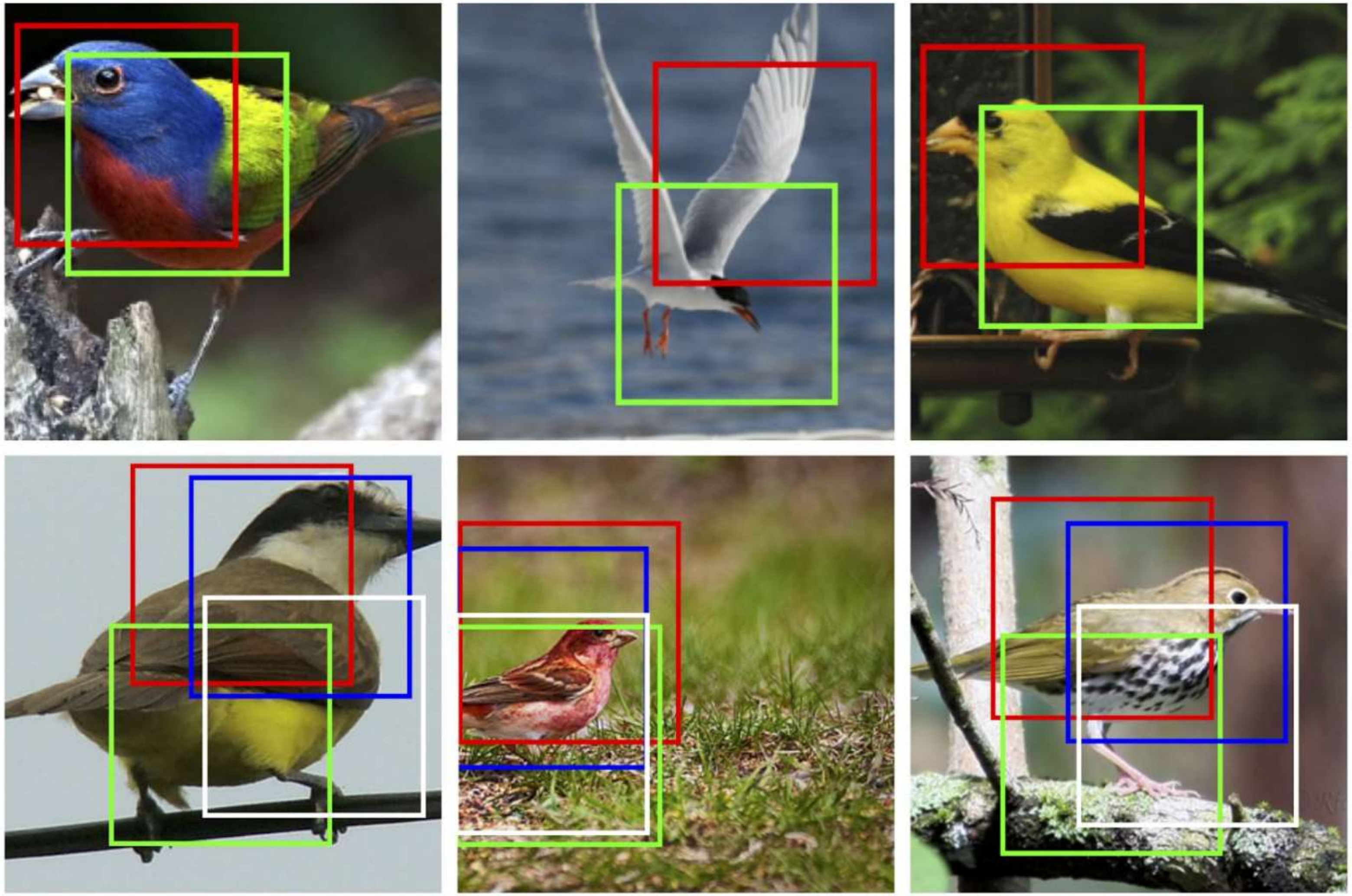
Spatial Transformer для классификации птиц

[Jaderberg et al., 2015]



Spatial Transformer для классификации птиц

[Jaderberg et al., 2015]



Заключение

- Разные способы встраивать алгоритмы в нейросети
  - Структурный пулинг
  - Итерации алгоритма => слои нейросети
  - Прямое (аналитическое) дифференцирование по входу

- С недифференцируемыми функциями можно бороться
  - Основной способ – сведение к дифференцируемым
  - Игнорировать недифференцируемую операцию
  - Стохастические активации (репараметризация – хорошо работает)
  - Совсем недифференцируемо – RL
    - Все сложно, нестабильно, долго, но иногда возможно