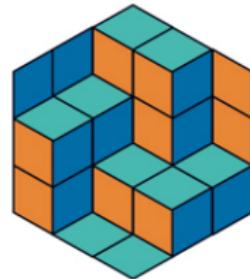


Лекция 2. Свёрточные нейронные сети, продолжение

Александр Юрьевич Авдюшенко

МКН СПбГУ

24 февраля 2022



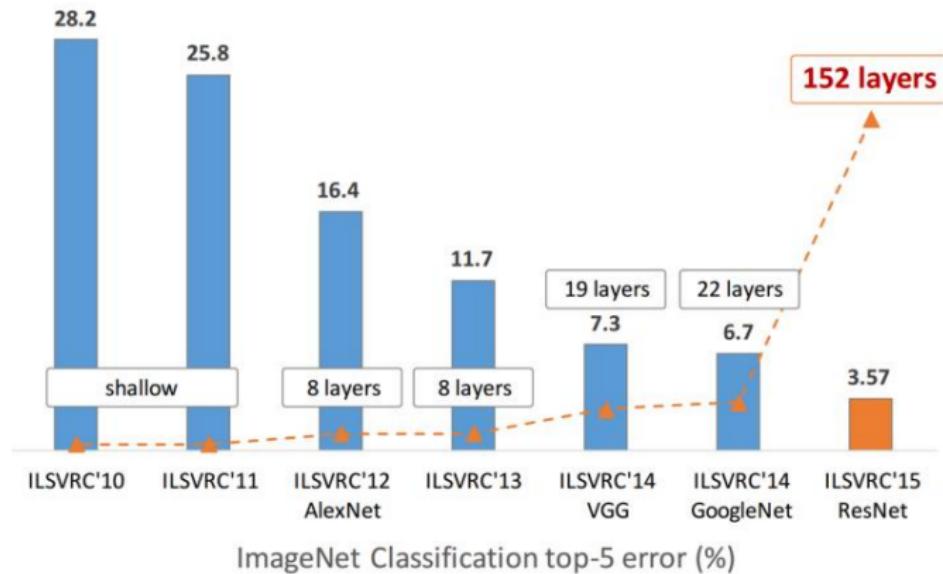
Факультет
математики
и компьютерных
наук
СПбГУ

Пятиминутка

- ▶ Определите операцию свёртки для нейронных сетей
- ▶ Выпишите основные особенности сети AlexNet
- ▶ Выпишите несколько популярных функций активации

Развитие свёрточных сетей

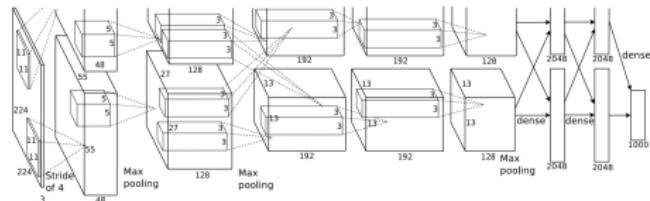
или краткая история ImageNet



AlexNet (Krizhevsky, Sutskever, Hinton, 2012)

Победитель конкурса ImageNet того времени

AlexNet CNN Architecture



- ▶ ReLU
- ▶ аугментация данных
- ▶ *dropout 0.5*
- ▶ *batch normalization* (размер батча 128)
- ▶ SGD Momentum 0.9
- ▶ L2 регуляризация 5e-4
- ▶ Learning rate 1e-2, потом уменьшение в 10 раз после стабилизации качества на тесте

Финальное качество top5 на ImageNet — 25.8% -> 16.4%

Gradient
Descent

Follow the
gradient

I'm
oscillating...
what do I do?

Learned
Optimizer

Aha! I've seen
this before...

AdaGrad

$$G = G + \mathcal{L}'_i(w) \odot \mathcal{L}'_i(w)$$
$$w = w - \eta_t \mathcal{L}'_i(w) \oslash \sqrt{G + \varepsilon}$$

η_t может быть зафиксирован, к примеру $\eta_t = 0.01$,
 \odot, \oslash — покомпонентное умножение и деление векторов
Преимущества:

- ▶ $\mathcal{L}'_i(w)$ — градиент на i -ом объекте
- ▶ отдельная адаптация шага для каждого измерения
- ▶ подходит для разреженных данных

Недостаток:

- ▶ G постоянно увеличивается, что может привести к остановке обучения

RMSProp (running mean square)

Выравнивание скоростей изменения весов скользящим средним

$$G = \alpha G + (1 - \alpha) \mathcal{L}'_i(w) \odot \mathcal{L}'_i(w)$$

$$w = w - \eta_t \mathcal{L}'_i(w) \oslash \sqrt{G + \varepsilon}$$

η_t может быть зафиксирован, к примеру $\eta_t = 0.01$

Преимущества:

- ▶ те же, что и у AdaGrad
- ▶ G не растёт бесконтрольно

Недостатки:

- ▶ в самом начале обучения G усредняется по нулевым предыдущим значениям
- ▶ нет учёта моментов

Adam (adaptive momentum)

$$\nu = \gamma\nu + (1 - \gamma)\mathcal{L}'_i(w)$$

$$\hat{\nu} = \nu(1 - \gamma^k)^{-1}$$

$$G = \alpha G + (1 - \alpha)\mathcal{L}'_i(w) \odot \mathcal{L}'_i(w)$$

$$\hat{G} = G(1 - \alpha^k)^{-1}$$

$$w = w - \eta_t \hat{\nu} \oslash (\sqrt{\hat{G}} + \varepsilon)$$

k — номер итерации, $\gamma = 0.9, \alpha = 0.999, \varepsilon = 10^{-8}$

Преимущества:

- ▶ те же, что и у RMSProp
- ▶ калибровка $\hat{\nu}, \hat{G}$ увеличивает ν, G на первых итерациях
- ▶ есть учёт моментов

Nadam (Nesterov-accelerated adaptive momentum)

Те же формулы для $\nu, \hat{\nu}, G, \hat{G}$

$$w = w - \eta_t \left(\gamma \hat{\nu} + \frac{1-\gamma}{1-\gamma^k} \mathcal{L}'_i(w) \right) \oslash (\sqrt{\hat{G}} + \varepsilon)$$

k — номер итерации, $\gamma = 0.9, \alpha = 0.999, \varepsilon = 10^{-8}$

T. Dozat. Incorporating Nesterov Momentum into Adam.
ICLR-2016

Пример сходимости

Источник: <http://www.denizyuret.com/2015/03/alec-radfords-animations-for.html>

Вопрос

Какой метод и как выбрать?

Диагональный метод Левенберга-Марквардта

напоминание

Метод Ньютона-Рафсона (второго порядка):

$$w = w - \eta(\mathcal{L}_i''(w))^{-1}\mathcal{L}_i'(w),$$

где $\mathcal{L}_i''(w) = \frac{\partial^2 \mathcal{L}_i(w)}{\partial w_{jh} \partial w_{j'h'}}$ — гессиан

Эвристика. Считаем, что гессиан диагонален:

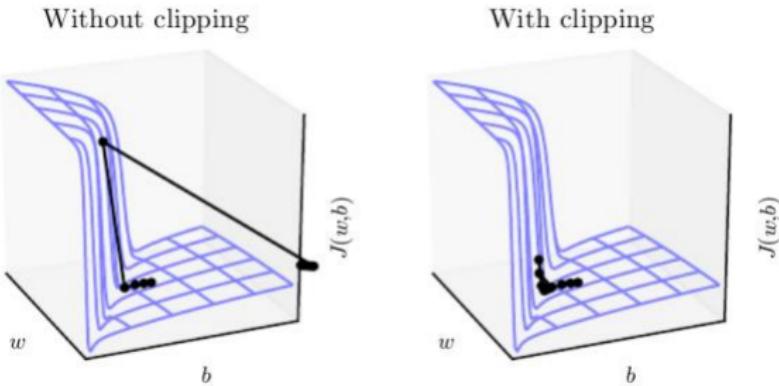
$$w_{jh} = w_{jh} - \eta \left(\frac{\partial^2 \mathcal{L}_i(w)}{\partial w_{jh}^2} + \mu \right)^{-1} \frac{\partial \mathcal{L}_i(w)}{\partial w_{jh}},$$

η — темп обучения, можно полагать $\eta = 1$

μ — параметр, предотвращающий обнуление знаменателя

Отношение η/μ есть темп обучения на ровных участках функционала $\mathcal{L}_i(w)$, где вторая производная обнуляется.

Взрыв градиента (gradient exploding)



Можно отсекать (пр. `clip`) по значениям или по модулю значений.

Источник: <https://neptune.ai/blog/understanding-gradient-clipping-and-how-it-can-fix-exploding-gradients-problem>

Методы регуляризации

Dropout — метод случайных отключений нейронов

Этап обучения: делая градиентный шаг $\mathcal{L}_i(w) \rightarrow \min_w$,

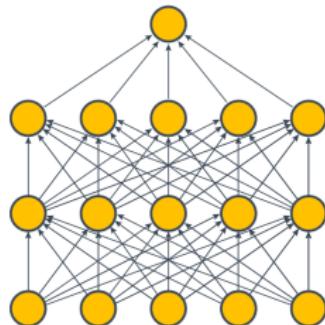
отключаем h -ый нейрон ℓ -го слоя с вероятностью p_ℓ :

$$x_{ih}^{\ell+1} = \xi_h^\ell \sigma_h(\sum_j w_{jh} x_{ij}^\ell), \quad P(\xi_h^\ell = 0) = p_\ell$$

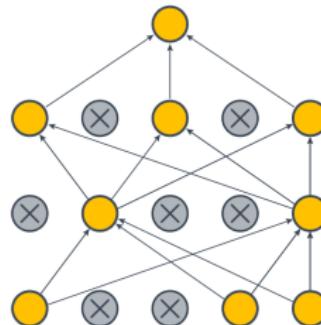
Этап применения: включаем все нейроны, но с поправкой:

$$x_{ih}^{\ell+1} = (1 - p_\ell) \sigma_h(\sum_j w_{jh} x_{ij}^\ell)$$

Standard Neural Net



After applying dropout



Вопрос

Какая особенность появляется при таком применении?

Inverted Dropout

На практике чаще используют не Dropout, а Inverted Dropout

Этап обучения:

$$x_{ih}^{\ell+1} = \frac{1}{1-p_\ell} \xi_h^\ell \sigma_h \left(\sum_j w_{jh} x_{ij}^\ell \right), \quad P(\xi_h^\ell = 0) = p_\ell$$

Этап применения не требует ни модификаций, ни знания p_ℓ :

$$x_{ih}^{\ell+1} = \sigma_h \left(\sum_j w_{jh} x_{ij}^\ell \right)$$

L_2 -регуляризация предотвращает рост параметров на обучении:

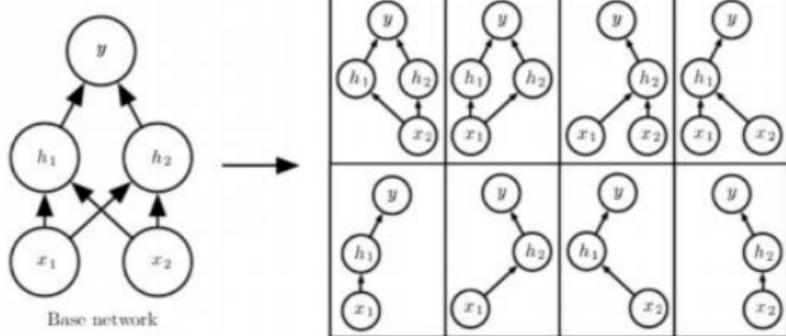
$$\mathcal{L}_i(w) + \frac{\lambda}{2} \|w\|^2 \rightarrow \min_w$$

Градиентный шаг с Dropout и L_2 -регуляризацией:

$$w = w(1 - \eta\lambda) - \eta \frac{1}{1-p_\ell} \xi_h^\ell \mathcal{L}'_i(w)$$

Интерпретации Dropout

- ▶ аппроксимируем простое голосование по 2^N сетям с общим набором N нейронов
- ▶ регуляризация: выбираем из всех сетей более устойчивую к утрате pN нейронов, моделируя надёжность мозга
- ▶ уменьшаем переобучение, заставляя разные части сети решать одну и ту же исходную задачу



Batch normalization (пакетная нормализация данных)

$B = \{x_i\}$ — пакеты (mini-batch) данных

Усреднение градиентов $\mathcal{L}_i(w)$ по пакету ускоряет сходимость.

$B^\ell = \{u_i^\ell\}$ — векторы объектов x_i на выходе ℓ -го слоя

1. Нормировать каждую j -ю компоненту вектора u_i^ℓ по пакету

$$\hat{u}_{ij}^\ell = \frac{u_{ij}^\ell - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}; \quad \mu_j = \frac{1}{|B|} \sum_{x_i \in B} u_{ij}^\ell; \quad \sigma_j^2 = \frac{1}{|B|} \sum_{x_i \in B} (u_{ij}^\ell - \mu_j)^2$$

2. Добавить линейный слой с настраивыми весами:

$$\tilde{u}_{ij}^\ell = \gamma_j^\ell \hat{u}_{ij}^\ell + \beta_j^\ell$$

3. Параметры $\gamma_j^\ell, \beta_j^\ell$ настраиваются BackProp

Начальное приближение весов

**When you initialise your ML
noob friend's NN weights with zeros**



Следим за изменениями дисперсий

как значений нейронов, так и градиентов при обратном распространении

Выравнивание дисперсий и нейронов, и градиентов в разных слоях — инициализация Ксавье для \tanh :

$$w_j \sim U \left[-\frac{6}{\sqrt{n_{in} + n_{out}}}, \frac{6}{\sqrt{n_{in} + n_{out}}} \right]$$

n_{in} , n_{out} — количество нейронов в предыдущем и текущем слоях соответственно

Вывод и подробности есть [в учебнике ШАД](#).

Следим за изменениями дисперсий

как значений нейронов, так и градиентов при обратном распространении

Выравнивание дисперсий и нейронов, и градиентов в разных слоях — инициализация Ксавье для \tanh :

$$w_j \sim U \left[-\frac{6}{\sqrt{n_{in} + n_{out}}}, \frac{6}{\sqrt{n_{in} + n_{out}}} \right]$$

n_{in} , n_{out} — количество нейронов в предыдущем и текущем слоях соответственно

Вывод и подробности есть в учебнике ШАД.

Вопрос

Что делать в случае активации $ReLU$?

Другие способы инициализации

1. Послойное обучение нейронов, как линейных моделей
 - ▶ либо по случайной подвыборке
 - ▶ либо по случайному подмножеству входов
 - ▶ либо из различных случайных начальных приближений

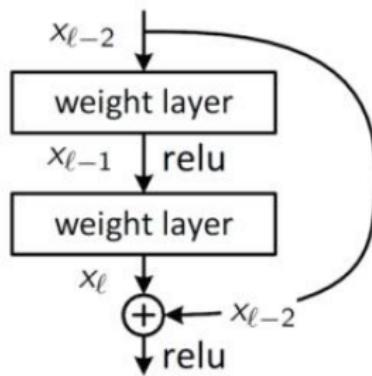
Таким образом обеспечивается различность нейронов.
2. Инициализация весами предобученной модели (например, автоэнкодера)

ResNet — Residual Net

Skip-connection (сквозная связь) слоя ℓ с предшествующим слоем $\ell - d$:

$$x_\ell = \sigma(Wx_{\ell-1}) + x_{\ell-d}$$

Слой ℓ выучивает не новое векторное представление x_ℓ , а его приращение $x_\ell - x_{\ell-d}$

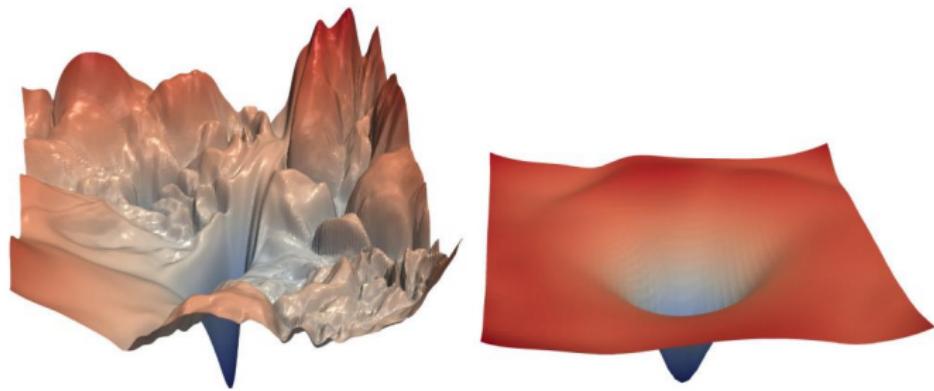


- ▶ (главное) Получается увеличивать число слоев
- ▶ Приращения более устойчивы → лучше сходимость

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition. 2015.

R.K. Srivastava, K. Greff, J. Schmidhuber. Highway Networks. 2015

ResNet: визуализация оптимизационного критерия



without skip connections

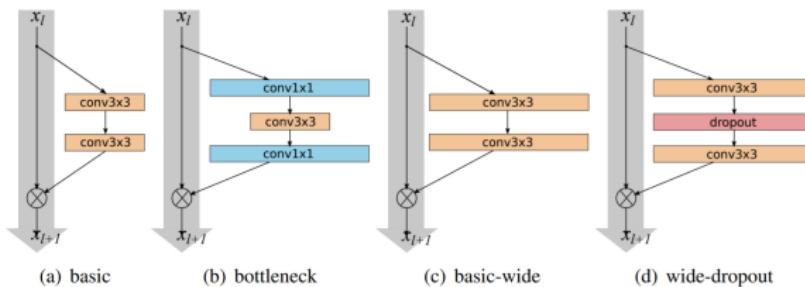
with skip connections

Hao Li et al. Visualizing the Loss Landscape of Neural Nets. 2018.

WideResNet — WRN

Недостатки ResNet

- ▶ слишком глубокие — до 1000 слоев :)
- ▶ долго обучаются до SotA



- ▶ простая WRN сеть из 16 слоев победила все ResNet
- ▶ WRN-40-4 (т.е. 40 слоев и 4x ширина) обучается в 8 раз быстрее, чем ResNet-1001

Резюме

- ▶ Свёрточные сети очень хорошо подходят для обработки изображений
- ▶ Различные алгоритмы оптимизации: adam, RMSProp
- ▶ Методы регуляризации: dropout, L_2 и batch normalization
- ▶ Ещё важен выбор начального приближения (инициализация весов)
- ▶ ResNet и skip-connections, WideResNet

Резюме

- ▶ Свёрточные сети очень хорошо подходят для обработки изображений
- ▶ Различные алгоритмы оптимизации: adam, RMSProp
- ▶ Методы регуляризации: dropout, L_2 и batch normalization
- ▶ Ещё важен выбор начального приближения (инициализация весов)
- ▶ ResNet и skip-connections, WideResNet

Что ещё можно посмотреть?

- ▶ Лекция курса в Стенфорде о свёрточных сетях
- ▶ Лекция К.В. Воронцова «Нейронные сети: градиентные методы оптимизации»