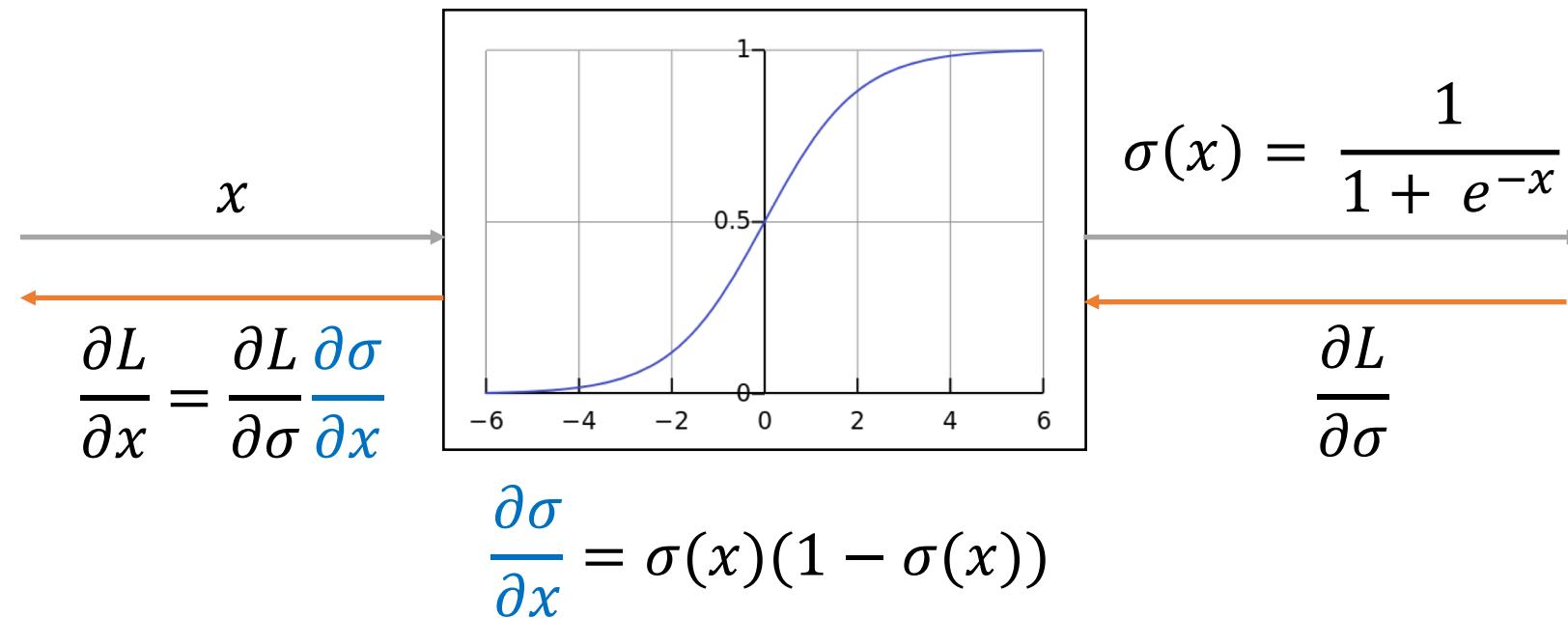


MML minor #5

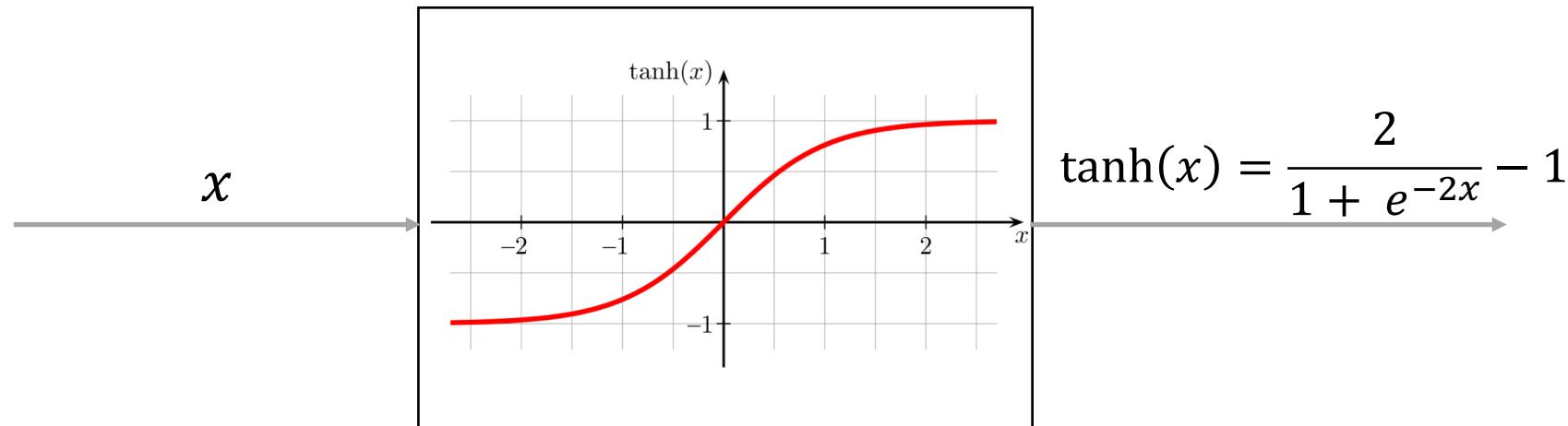
Нейронные сети: трюки

Sigmoid активация



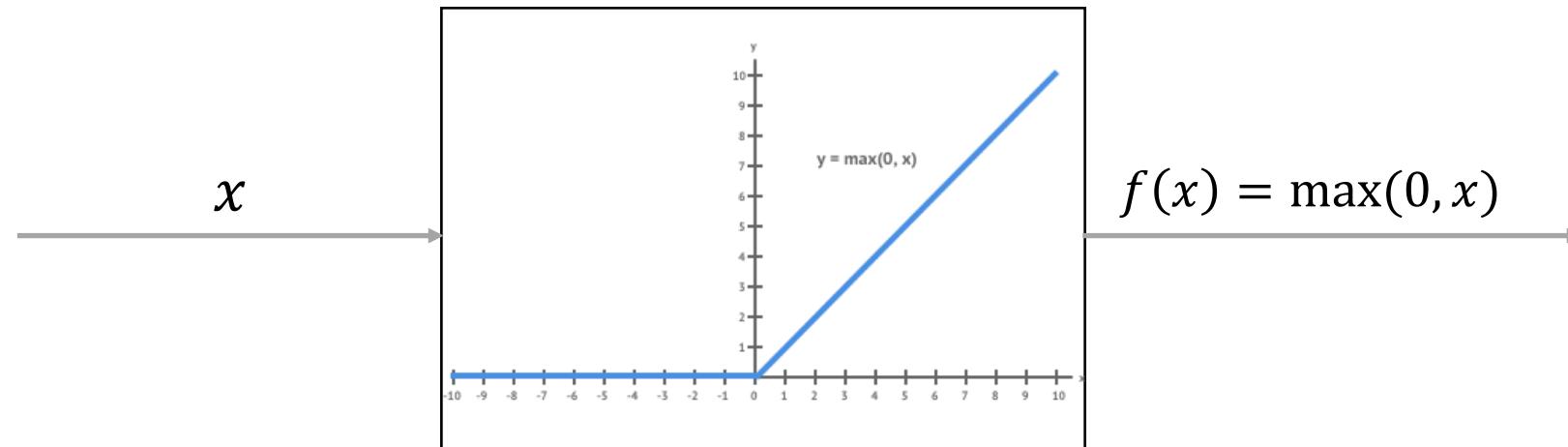
- Нейроны с сигмоидой могут насыщаться и приводить к угасающим градиентам.
- Не центрированы в нуле.
- e^x дорого вычислять.

Tanh активация



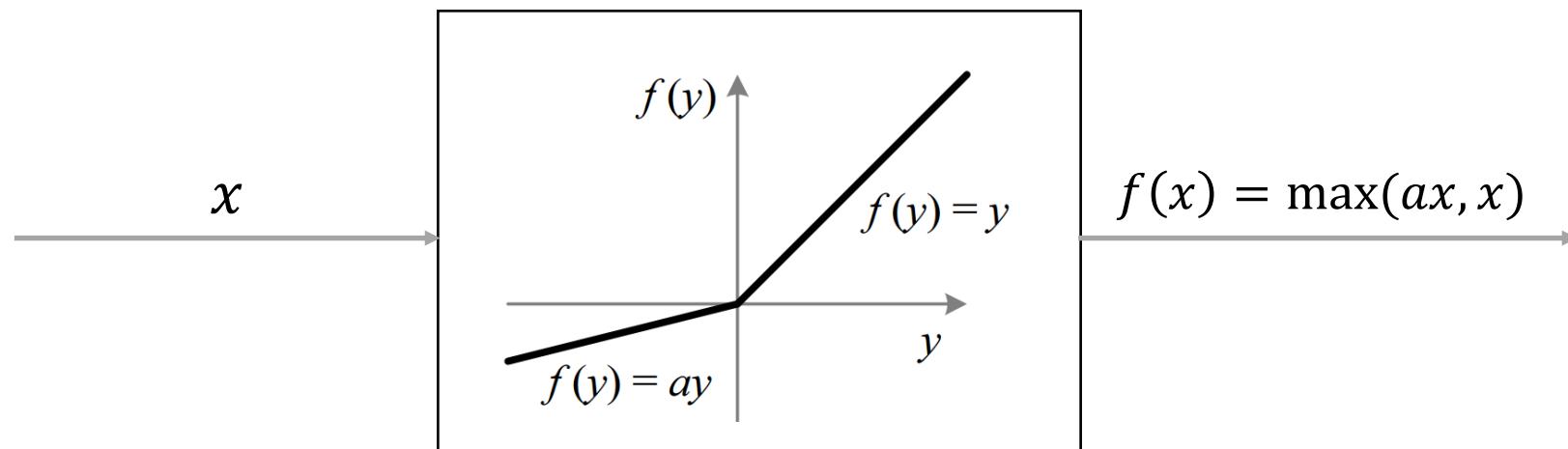
- Центрирован в нуле.
- Но все еще как сигмоида.

ReLU активация



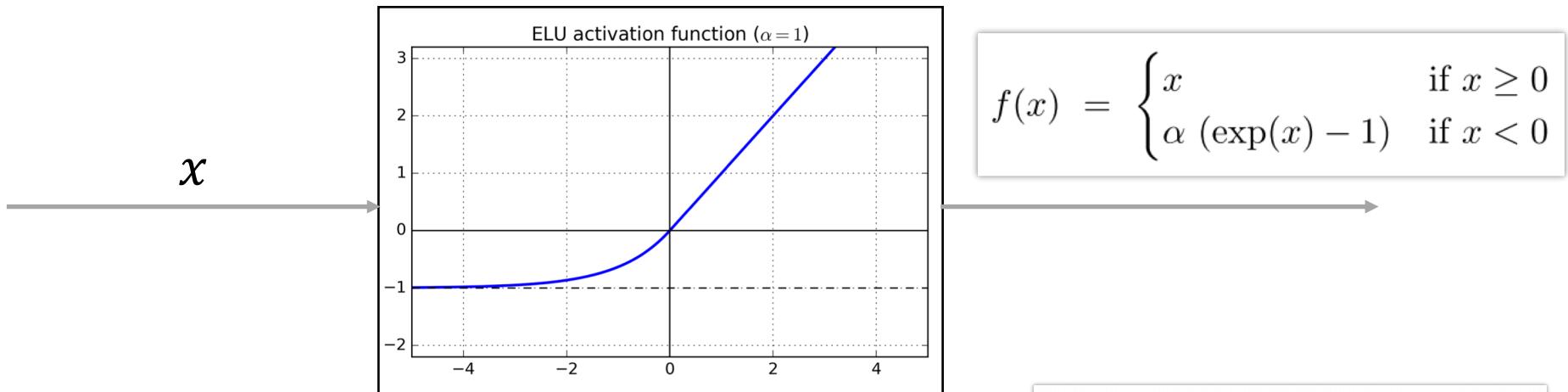
- Быстро считается.
- Градиенты не угасают при $x > 0$.
- На практике ускоряет сходимость!
- Не центрирован в нуле.
- Могут умереть: если не было активации - не будет обновления!

Leaky ReLU активация

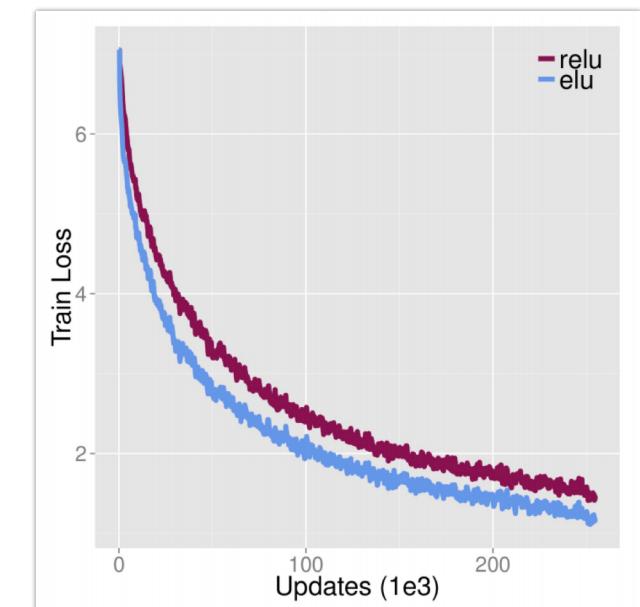


- Всегда будут обновления!
- Примерно центрирован в нуле
- $a \neq 1$

ELU активация

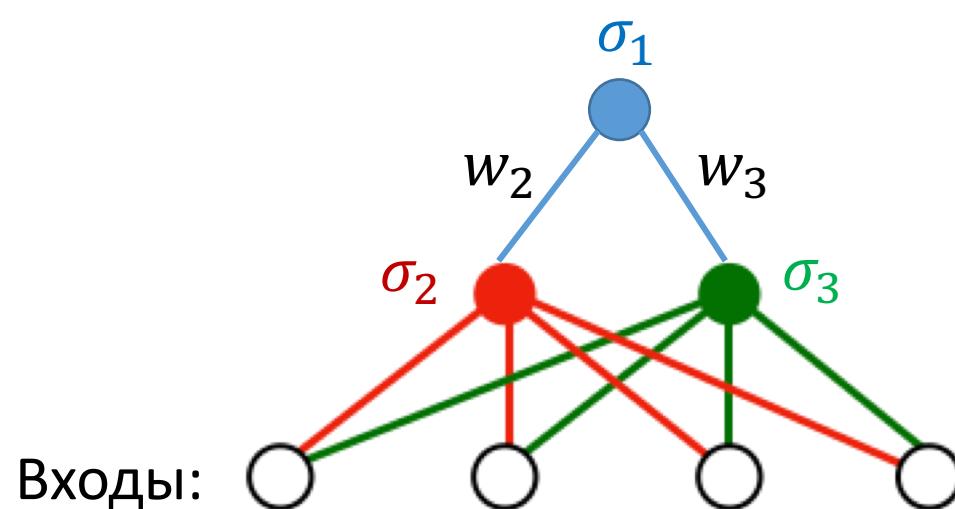


- Примерно центрирован в нуле
- Сходимость быстрее ReLU



Инициализация весов

Давайте начнем с нулей?



$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \sigma_1} \sigma_1(1 - \sigma_1) \sigma_2$$

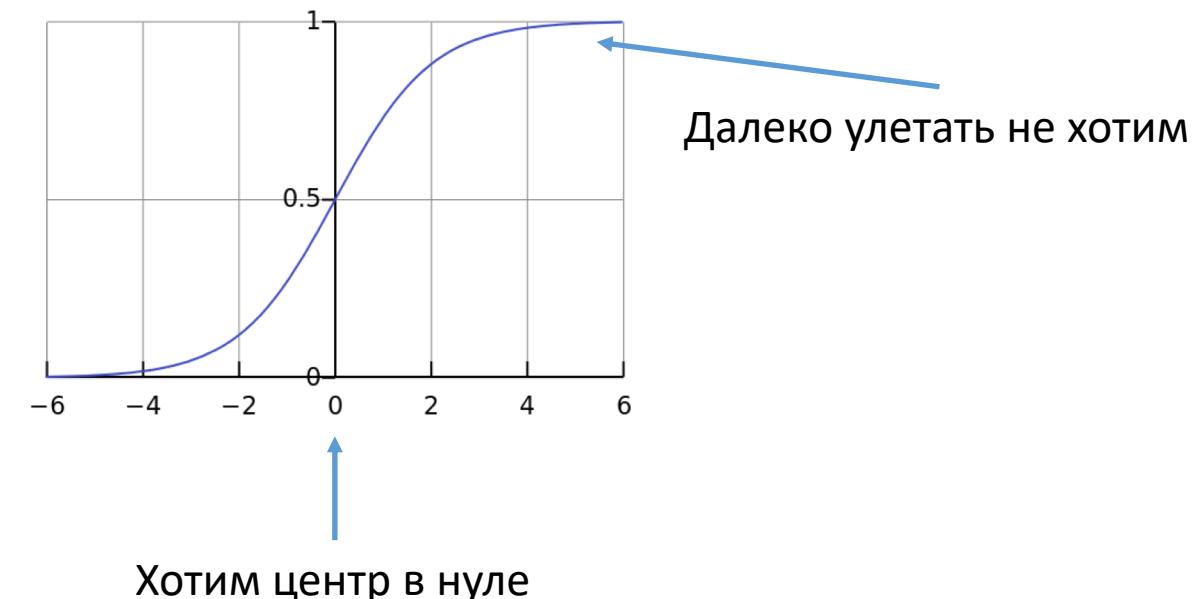
$$\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial \sigma_1} \sigma_1(1 - \sigma_1) \sigma_3$$

σ_2 и σ_3 обновляются
одинаково!

- Нужно сломать симметрию!
- Может случайным шумом?
- Но насколько большим? $0.03 \cdot \mathcal{N}(0,1)$?

Инициализация весов

- Нейрон до активации: $\sum_{i=1}^n x_i w_i$.
- Если $E(x_i) = E(w_i) = 0$ и мы генерируем веса независимо от входов, тогда $E(\sum_{i=1}^n x_i w_i) = 0$.
- Но дисперсия может расти!
- И это замедлит сходимость!



Инициализация весов

- Дисперсия до активации $\sum_{i=1}^n x_i w_i$:

$$Var(\sum_{i=1}^n x_i w_i) = \text{некоррелированные } x_i w_i \text{ и } x_j w_j \text{ из-за i.i.d. } w_k$$

$$= \sum_{i=1}^n Var(x_i w_i) = \text{ } w_i \text{ и } x_i \text{ независимы}$$

$$= \sum_{i=1}^n \left([E(x_i)]^2 Var(w_i) + [E(w_i)]^2 Var(x_i) + Var(x_i)Var(w_i) \right) = \text{ } w_i \text{ и } x_i \text{ имеют 0 среднее}$$

$$= \sum_{i=1}^n Var(x_i)Var(w_i) = Var(x)[n Var(w)]$$

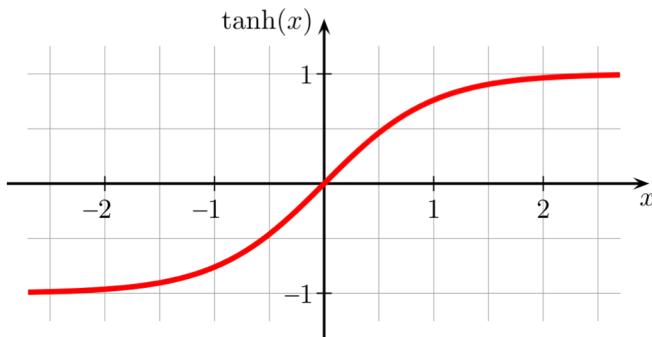
↑
Хотим 1

Инициализация весов

- Воспользуемся фактом: $Var(aw) = a^2Var(w)$.
- Для того, чтобы $[n Var(aw)]$ была 1
нужно умножить $\mathcal{N}(0,1)$ веса ($Var(w) = 1$)
на $a = 1/\sqrt{n}$.
- Также известна как Xavier инициализация (Glorot et al.),
в статье умножают на $\sqrt{2}/\sqrt{n_{in} + n_{out}}$.
- Инициализация для ReLU (He et al.) умножает на $\sqrt{2}/\sqrt{n_{in}}$.

Помогает сходимости

- \tanh для маленьких значений похож на линейную функцию, наши выкладки становятся *примерно верны*



- Однаковый масштаб активаций \Rightarrow градиентов на разных слоях помогает сходимости (все веса меняются одинаково быстро)

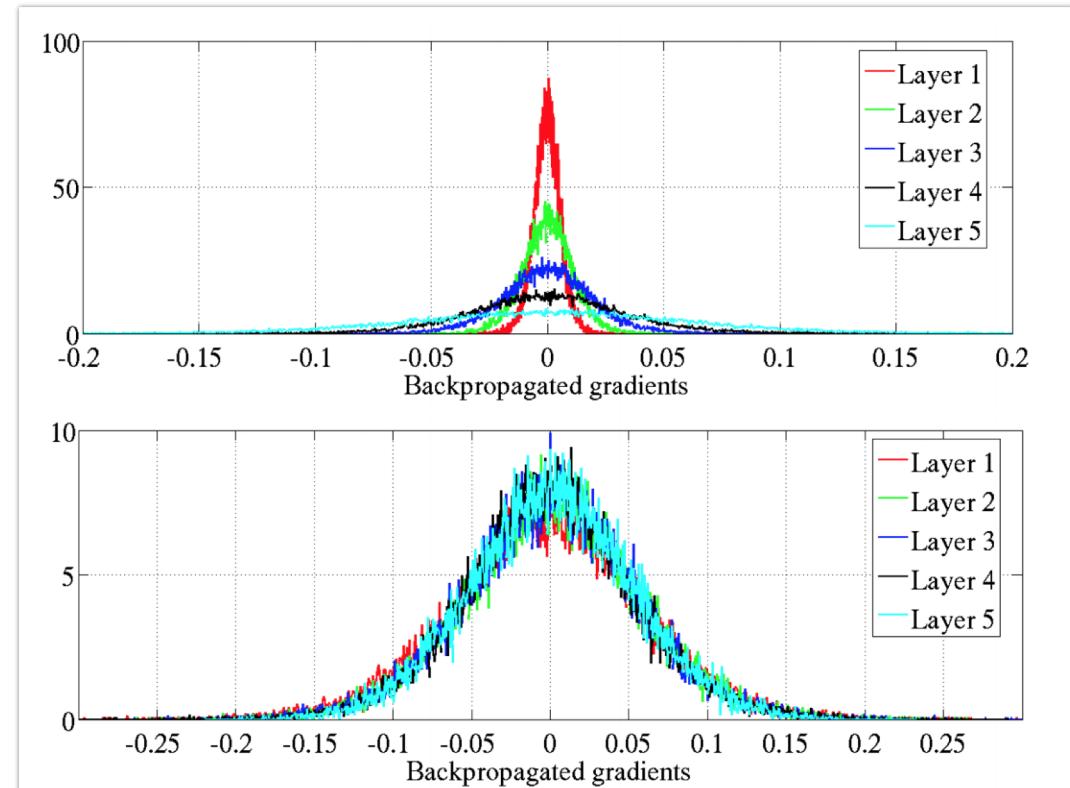
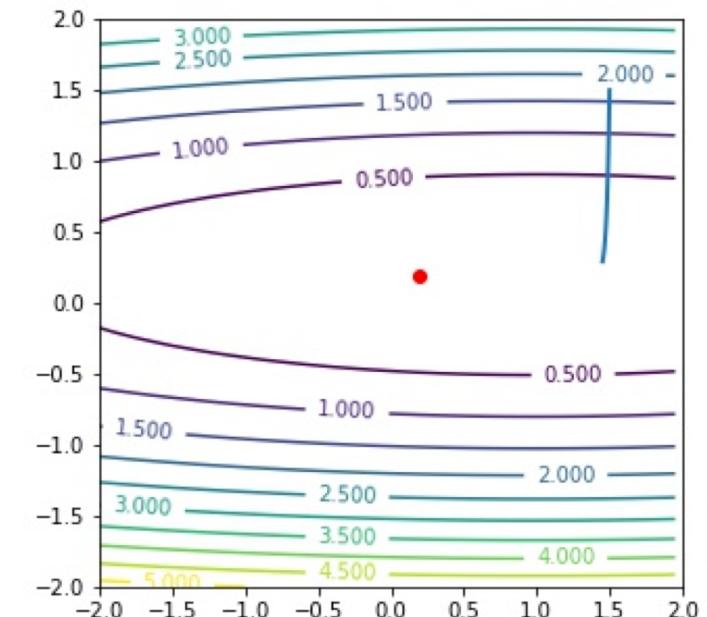
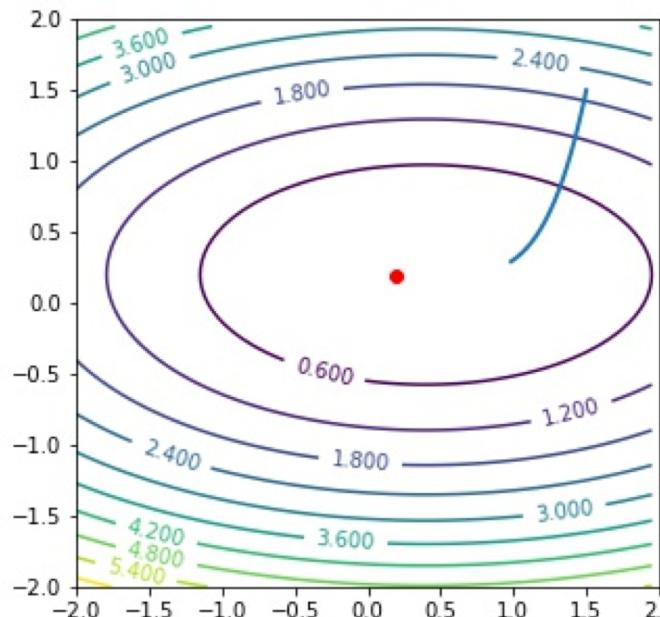
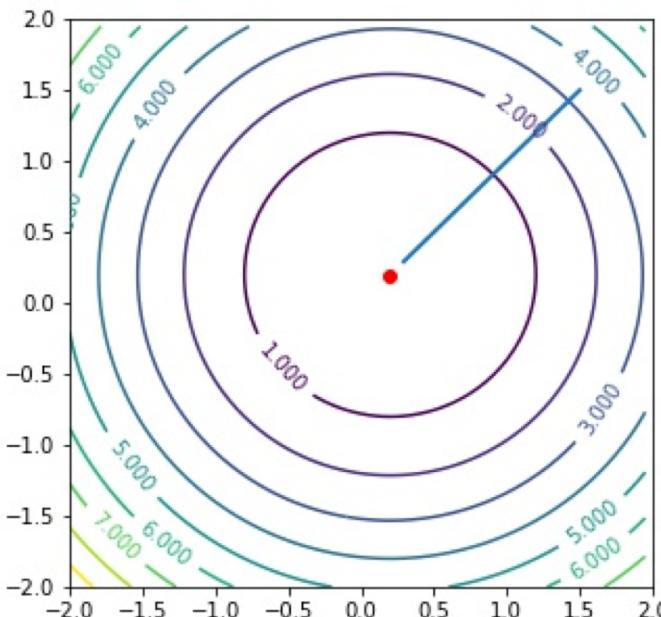


Figure 7: Back-propagated gradients normalized histograms with hyperbolic tangent activation, with standard (top) vs normalized (bottom) initialization. Top: 0-peak decreases for higher layers.

Помогает сходимости

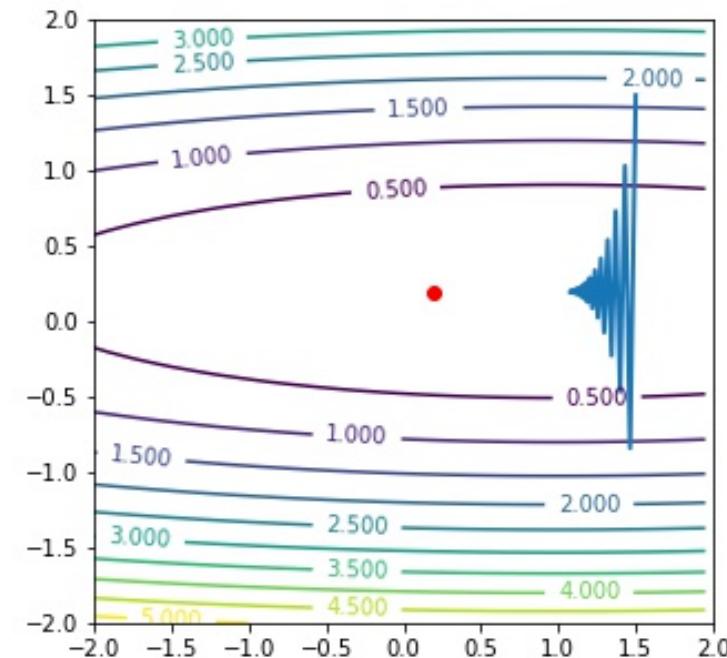
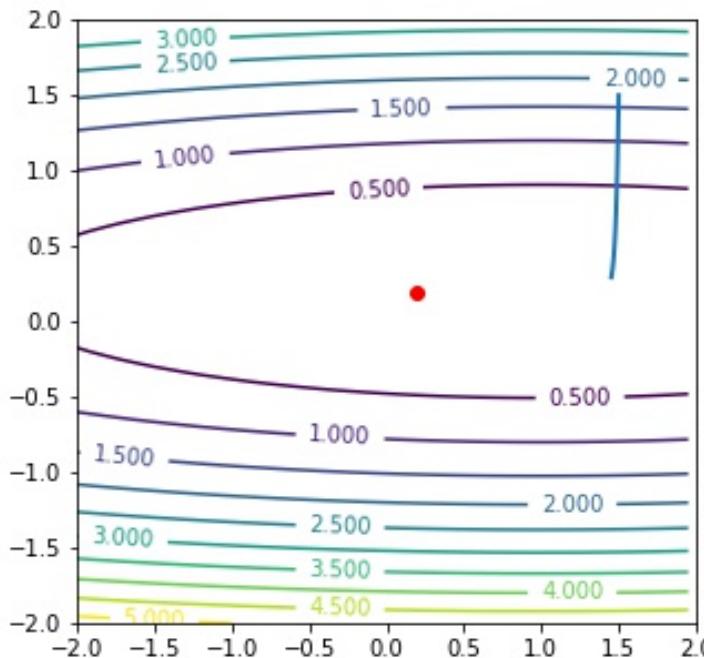
- Фиксируем learning rate и количество шагов



Градиентные методы быстрее сходятся, если градиенты в одном масштабе!

Увеличение learning rate не особо помогает

- Увеличим learning rate:



По оси x движение быстрее, а по оси y слишком большие шаги!

Batch нормализация

- Мы знаем как ограничить дисперсию при инициализации.
- Но что будет с весами во время back-propagation?
- Batch нормализация пытается *нормировать* выходы нейронов (до или после активации).

Batch нормализация

- Нормализуем выход нейрона h_i :

$$h_i = \gamma_i \frac{h_i - \mu_i}{\sqrt{\sigma_i^2}} + \beta_i$$

среднее 0, дисперсия 1

Batch нормализация

- Нормализуем выход нейрона h_i :

$$h_i = \gamma_i \frac{h_i - \mu_i}{\sqrt{\sigma_i^2}} + \beta_i$$

среднее 0, дисперсия 1

- Где взять μ_i и σ_i^2 ? Оценим их по **текущему батчу!**

Потому что во время обучения сеть все время меняется!

Batch нормализация

- Нормализуем выход нейрона h_i :

$$h_i = \gamma_i \frac{h_i - \mu_i}{\sqrt{\sigma_i^2}} + \beta_i$$

среднее 0, дисперсия 1

- Где взять μ_i и σ_i^2 ? Оценим их по **текущему батчу!**
- Во время тестирования будем использовать скользящее среднее:

Нам не нужны шумы при предсказании!

$$0 < \alpha < 1$$

$$\mu_i = \alpha \cdot \mathbf{mean}_{\text{batch}} + (1 - \alpha) \cdot \mu_i$$

$$\sigma_i^2 = \alpha \cdot \mathbf{variance}_{\text{batch}} + (1 - \alpha) \cdot \sigma_i^2$$

Batch нормализация

- Нормализуем выход нейрона h_i :

$$h_i = \gamma_i \frac{h_i - \mu_i}{\sqrt{\sigma_i^2}} + \beta_i$$

среднее 0, дисперсия 1

- Где взять μ_i и σ_i^2 ? Оценим их по **текущему батчу**!
- Во время тестирования будем использовать скользящее среднее:

$$0 < \alpha < 1$$

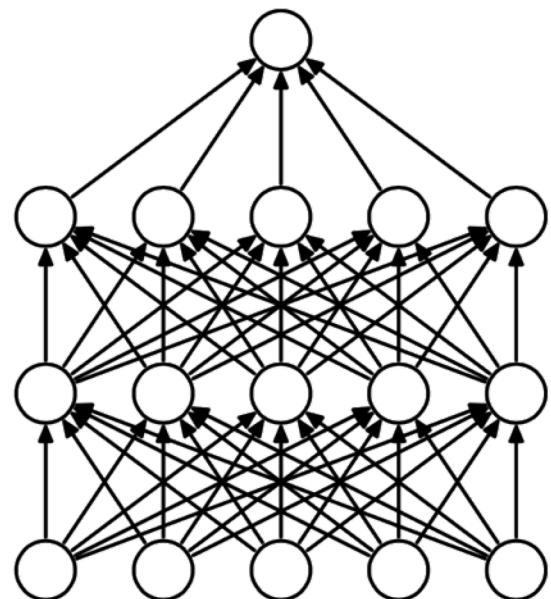
$$\mu_i = \alpha \cdot \mathbf{mean}_{\text{batch}} + (1 - \alpha) \cdot \mu_i$$

$$\sigma_i^2 = \alpha \cdot \mathbf{variance}_{\text{batch}} + (1 - \alpha) \cdot \sigma_i^2$$

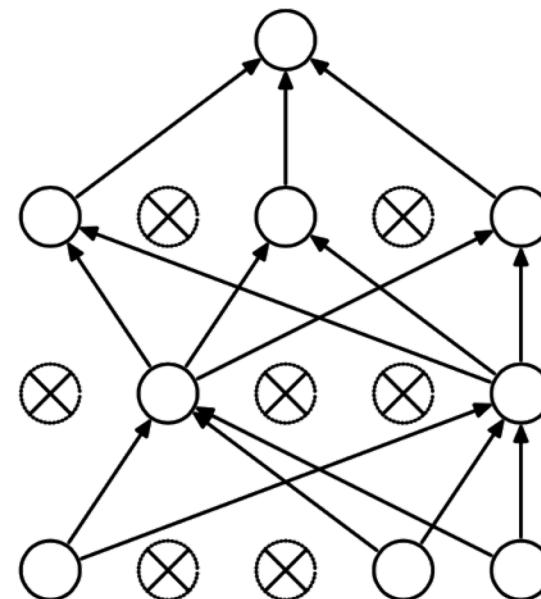
- Что делать с γ_i и β_i ? Линейная комбинация – это дифференцируемая операция, применим **backpropagation**!

Dropout регуляризация

- Оставляем нейрон с вероятностью p , а иначе выключаем (заменяем на 0)
- Таким образом мы семплируем сеть на каждом шаге обучения и меняем не все параметры \Rightarrow регуляризуем



(a) Standard Neural Net

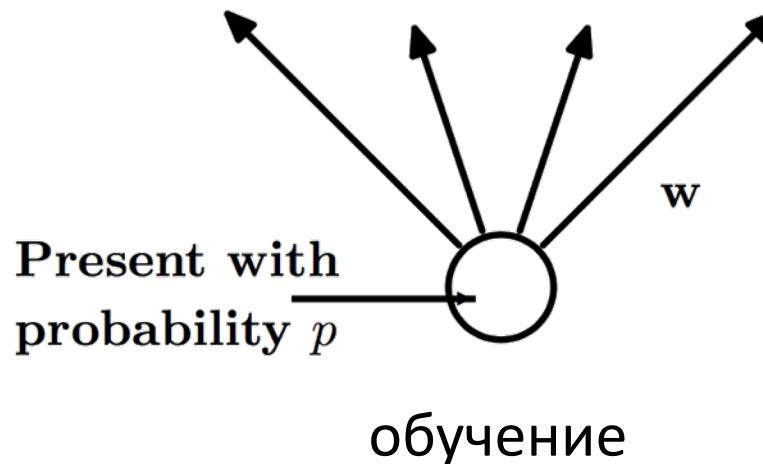


(b) After applying dropout.

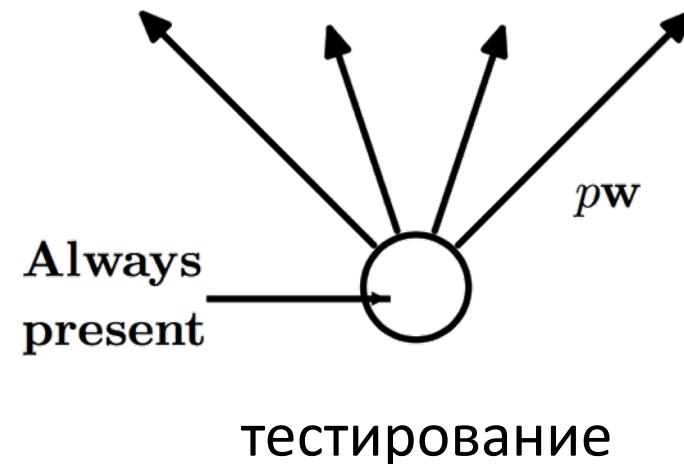
Dropout

- Во время тестирования все нейроны включены, но умножены на p для поддержания масштаба:

$$\text{Ожидаемый вес: } p \cdot w + (1 - p) \cdot 0$$



обучение



тестирование

- Авторы утверждают, что это *отдаленно* напоминает ансамбль из более маленьких нейросетей

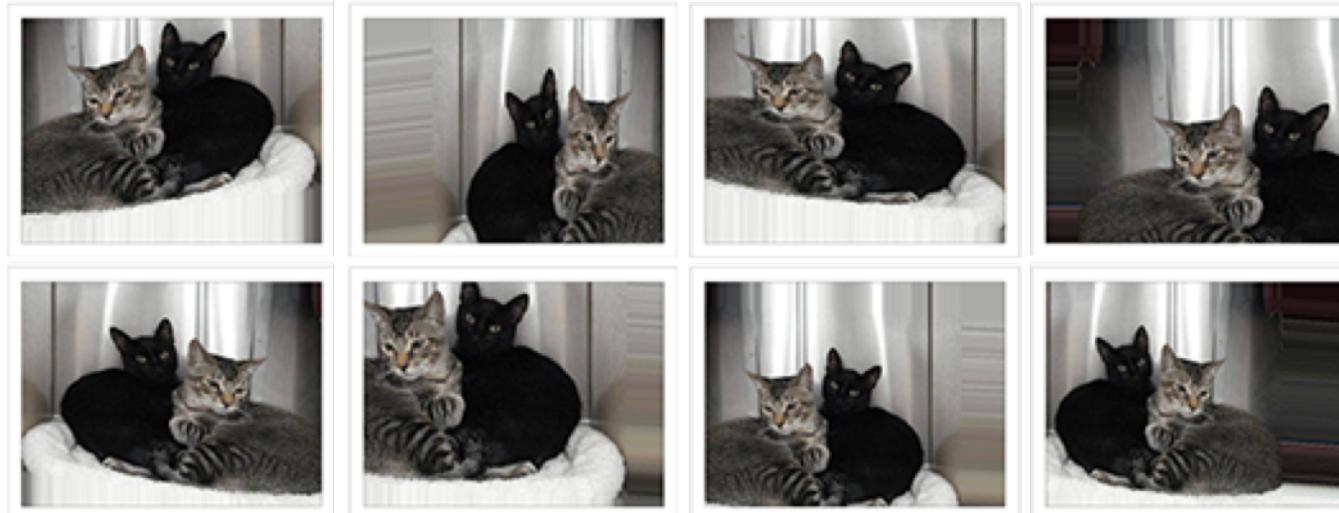
Аугментация данных

- В современных сетях миллионы параметров!
- Но размеры выборок ограничены!
- Можем сгенерировать новые примеры при помощи отражений, поворотов, цветовых изменений, масштабирования, ...
- **Нужно ли добавлять сдвиги?**



Аугментация данных

- В современных сетях миллионы параметров!
- Но размеры выборок ограничены!
- Можем сгенерировать новые примеры при помощи отражений, поворотов, цветовых изменений, масштабирования, ...
- **Сдвиги не нужны, лучше добавить пулинг!**



ImageNet классификация

- 1000 классов, 1.2 миллиона примеров
- Топ 5 ошибка человека: ~5%



flamingo



cock



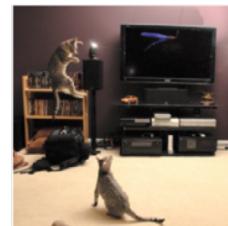
ruffed grouse



quail



partridge



Egyptian cat



Persian cat



Siamese cat

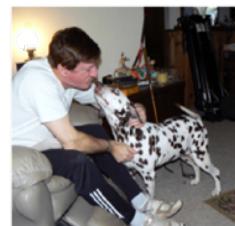


tabby



lynx

...



dalmatian



keeshond



miniature schnauzer



standard schnauzer

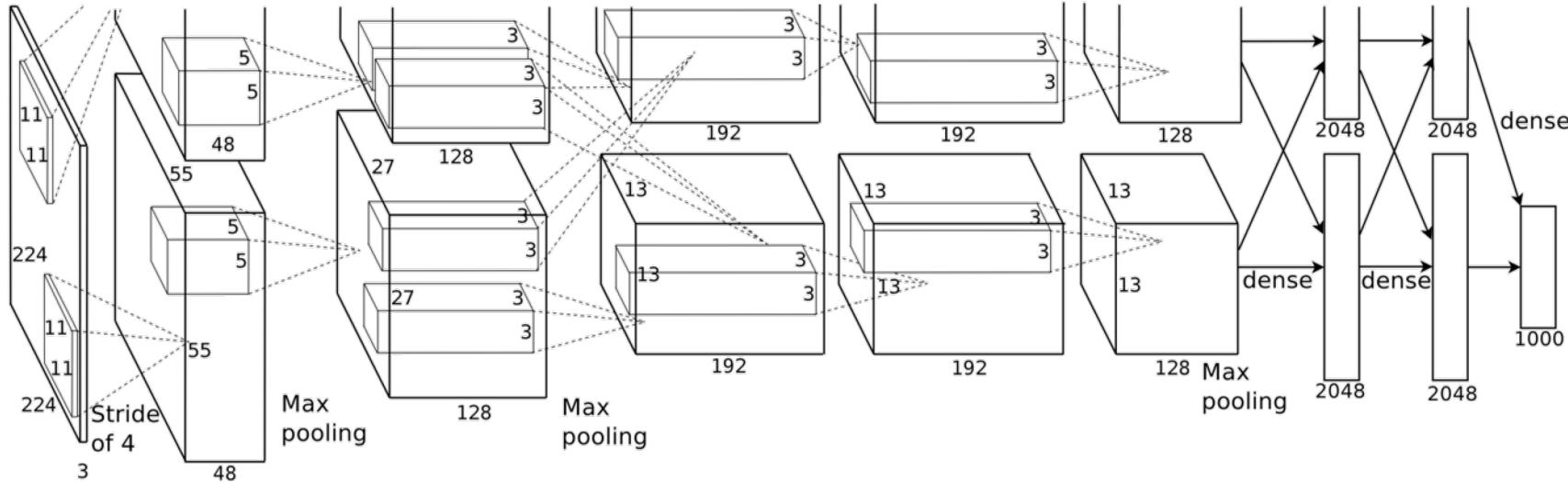


giant schnauzer

...

AlexNet (2012)

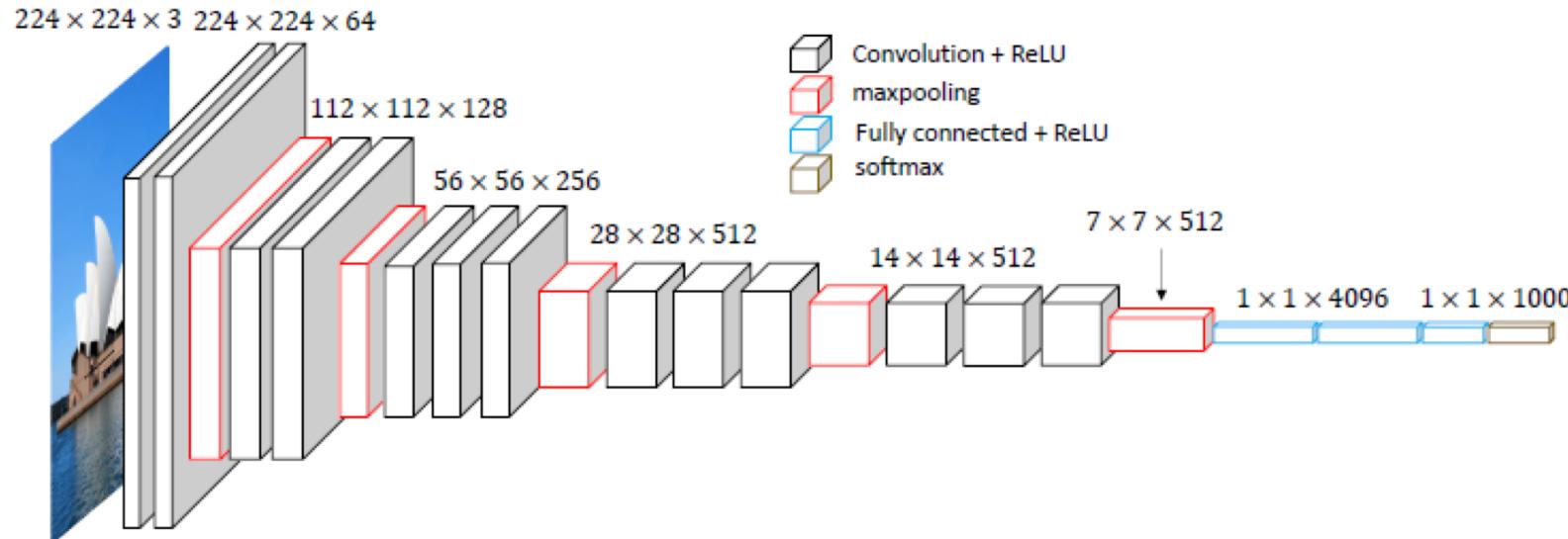
- Первая глубокая сеть для ImageNet
- В свое время уронила top 5 ошибку с 26% до 15%



- 11x11, 5x5, 3x3 свертки, макс пуллинг, dropout, аугментация данных, ReLU, SGD с моментом
- 60 миллионов параметров
- Обучается на 2 GPU 6 дней

VGG (2015)

- Только свертки 3x3, но сильно больше фильтров!
- ImageNet top 5 ошибка: 8.0% (одна модель)



- Обучение такое же как у AlexNet
- 138 миллионов параметров
- Учится на 4 GPU 2-3 недели

Ссылки

- <http://cnl.salk.edu/~schraudo/teach/NNcourse/precond.html>