



ТЕХНОСФЕРА

Лекция 5 Улучшение сходимости нейросетей

Байгушев Данила

23 сентября 2019 г.

Улучшение сходимости

Ускорение сходимости

- ▶ Инициализация (Xavier, He)
- ▶ Нормализация (Batch Normalization, Layer normalization)

Борьба с переобучением

- ▶ Регуляризация (Dropout, DropConnect)

Инициализация весов

Xavier (Glorot)

Рассмотрим нечетную функцию с единичной производной в нуле в качестве активации (напр. tanh)

- ▶ Хотим начать из линейного региона, чтобы избежать затухающих градиентов

$$z^{i+1} = f(\underbrace{z^i W^i}_{s^i})$$

$$\mathbb{D}[z^i] = \mathbb{D}[x] \prod_{k=0}^{i-1} n_k \mathbb{D}[W^k]$$

$$\mathbb{D}\left[\frac{\partial L}{\partial s^i}\right] = \mathbb{D}\left[\frac{\partial L}{\partial s^d}\right] \prod_{k=i}^d n_{k+1} \mathbb{D}[W^k]$$

Где n_i — размерность i -того слоя

Xavier (Glorot)

Хорошая инициализация:

$$\forall(i, j) \left\{ \begin{array}{l} \mathbb{D}[z^i] = \mathbb{D}[z^j] \\ \mathbb{D}\left[\frac{\partial L}{\partial s^i}\right] = \mathbb{D}\left[\frac{\partial L}{\partial s^j}\right] \end{array} \right.$$

Это эквивалентно следующему:

$$\forall i \left\{ \begin{array}{l} n_i \mathbb{D}[W^i] = 1 \\ n_{i+1} \mathbb{D}[W^i] = 1 \end{array} \right.$$

Компромисс: $\mathbb{D}[W^i] = \frac{2}{n_i + n_{i+1}}$

$$W^i \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}}, \frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}}\right]$$

$$\mathbb{D}[U(a, b)] = \frac{1}{12}(b - a)^2$$

He¹

Рассмотрим ReLU в качестве активации:

- ▶ Функция не симметрична
- ▶ Не дифференцируема в нуле

$$\mathbb{D}[z^i] = \mathbb{D}[x] \left(\prod_{k=0}^{i-1} \frac{1}{2} n_k \mathbb{D}[W^k] \right) \Rightarrow \mathbb{D}[W^k] = \frac{2}{n_k}$$

$$\mathbb{D}\left[\frac{\partial L}{\partial s^i}\right] = \mathbb{D}\left[\frac{\partial L}{\partial s^d}\right] \left(\prod_{k=i}^d \frac{1}{2} n_{k+1} \mathbb{D}[W^k] \right) \Rightarrow \mathbb{D}[W^k] = \frac{2}{n_{k+1}}$$

Достаточно использовать только первое уравнение:

$$\mathbb{D}\left[\frac{\partial L}{\partial s^i}\right] = \mathbb{D}\left[\frac{\partial L}{\partial s^d}\right] \prod_{k=1}^d \frac{1}{2} n_{k+1} \mathbb{D}[W^k] = \frac{n_2}{n_d} \mathbb{D}\left[\frac{\partial L}{\partial s^d}\right]$$

n_2/n_d небольшое для сверточных сетей

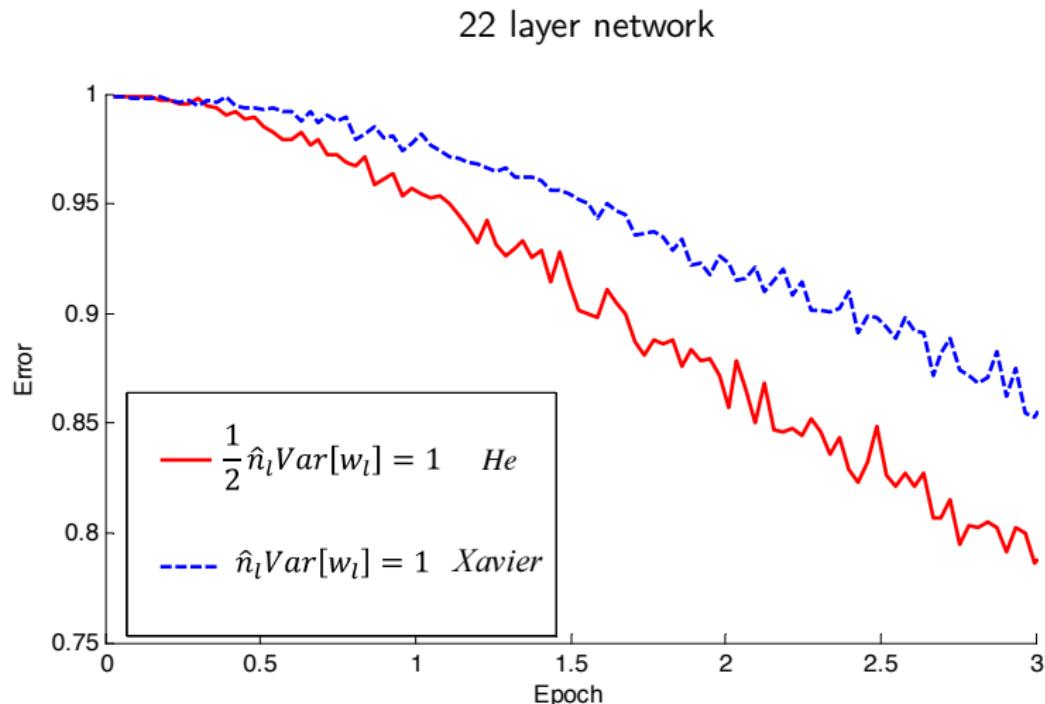
$$W^i \sim N(0, \frac{2}{n_i})$$

or

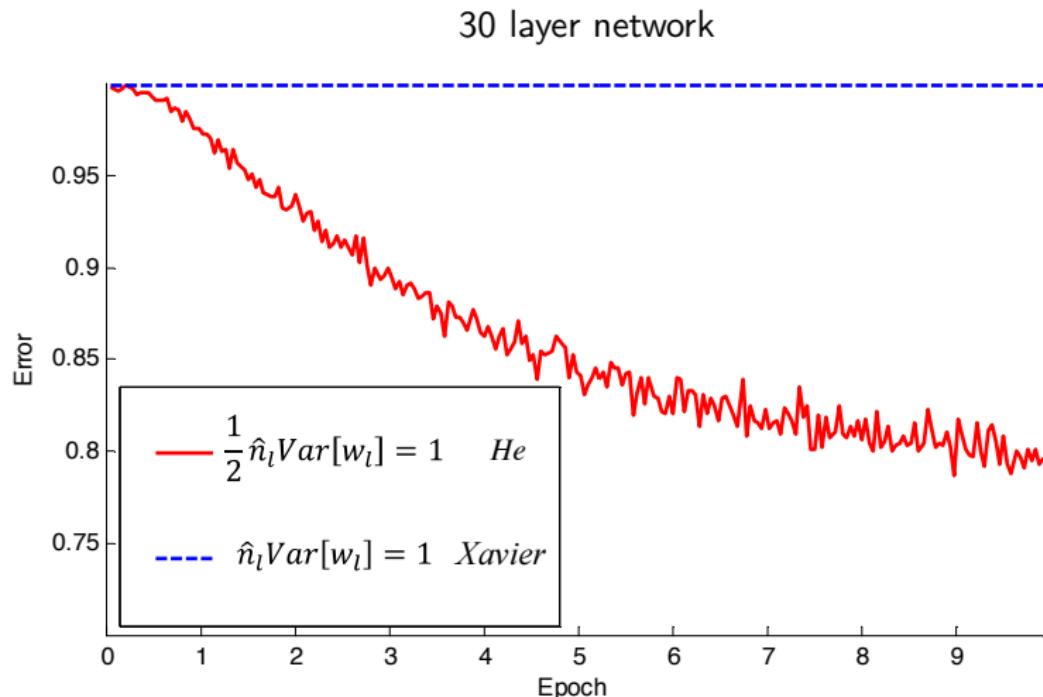
$$W^i \sim N(0, \frac{2}{n_{i+1}})$$

¹<https://arxiv.org/abs/1502.01852>

Xavier против He для ReLU



Xavier против He для ReLU



Ортогональная инициализация²

Выберем ортогональную матрицу весов W : $WW^T = I$. Тогда:

- ▶ $\|W_i x\| = \|x\|$ — норма сохраняется
- ▶ $\langle W_i, W_j \rangle = \delta_{ij}$ — все нейроны делают «разные» преобразования

Что делать для сверточных слоев?

²<https://hjweide.github.io/orthogonal-initialization-in-convolutional-layers>

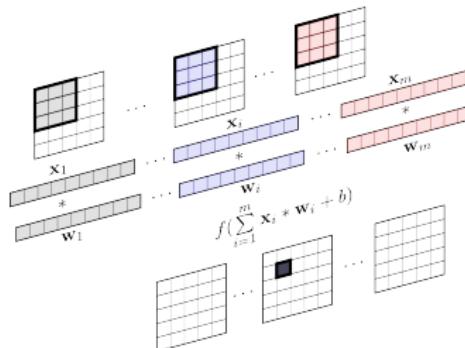
//hjweide.github.io/orthogonal-initialization-in-convolutional-layers

Ортогональная инициализация²

Выберем ортогональную матрицу весов W : $WW^T = I$. Тогда:

- ▶ $\|W_i x\| = \|x\|$ — норма сохраняется
- ▶ $\langle W_i, W_j \rangle = \delta_{ij}$ — все нейроны делают «разные» преобразования

Что делать для сверточных слоев?

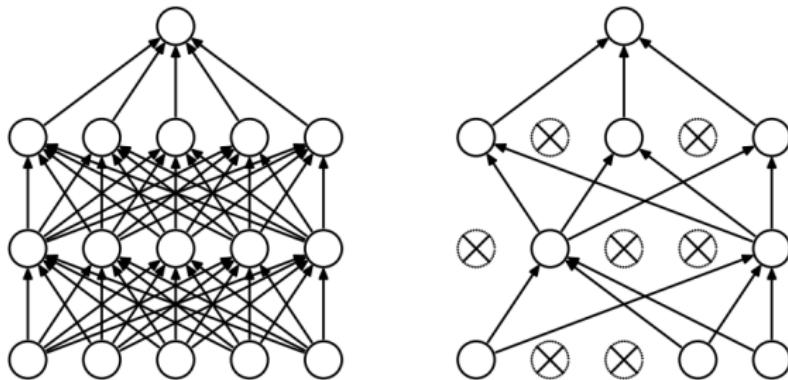


1. Генерируем ортогональную матрицу $W \in \mathbb{R}^{c' \times k^2 c}$
2. Reshape: $K \in \mathbb{R}^{c' \times c \times k \times k}$

²<https://hjweide.github.io/orthogonal-initialization-in-convolutional-layers>

Регуляризация

Dropout³



- ▶ С вероятностью p занулим выход нейрона (например, $p = 0.5$)
- ▶ В test-time домножаем веса на вероятность сохранения
- ▶ Не стоит выкидывать нейроны последнего слоя

³Dropout: A Simple Way to Prevent Neural Networks from Overfitting N. Srivastava, G. Hinton

Dropout, мотивация

- ▶ Борьба с соадаптацией – нейроны больше не могут рассчитывать на наличие соседей
- ▶ Биология: не все гены родителей будут присутствовать у потомков
- ▶ Усреднение большого (2^n) числа моделей

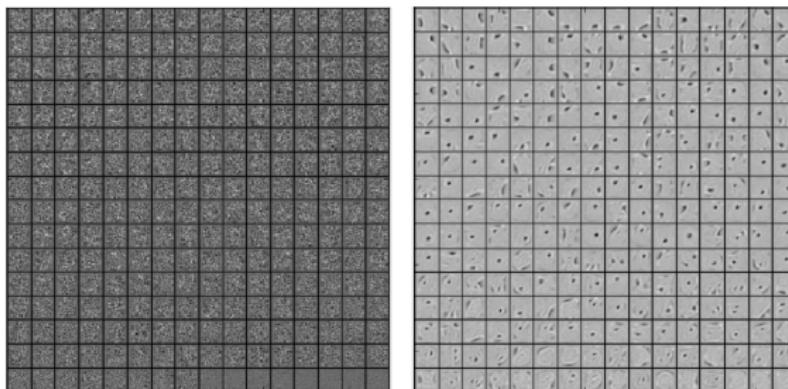
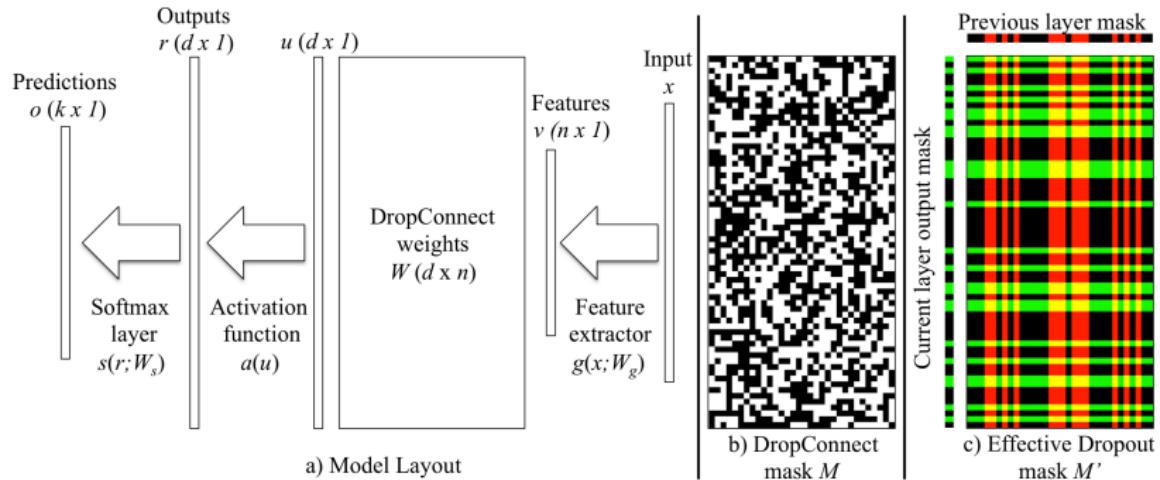


Figure: Выученные признаки на MNIST (автокодировщик с одним скрытым слоем и ReLU в качестве активации). Слева: без Dropout, справа – с Dropout

Dropconnect⁴



- Зануляем не выходы нейронов, а каждый вес по отдельности

⁴<https://cs.nyu.edu/~wanli/dropc/dropc.pdf>

Нормализация

Мотивация

- ▶ Обычно наблюдается более быстрая сходимость при декорелированных входах
- ▶ Whitening: $\hat{\mathbf{x}} = Cov[\mathbf{x}]^{-1/2}(\mathbf{x} - E[\mathbf{x}])$
- ▶ Нормализация: $\hat{x}^{(k)} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{Var[x^{(k)}]}}$ Для каждой размерности

Батч-нормализация⁵

- ▶ Covariate shift: изменение распределения входов во время обучения
- ▶ Цель — уменьшить covariate shift скрытых слоев
- ▶ Нормализуем входы в каждый слой $\hat{x}^{(k)} = \frac{x^{(k)} - \mathbb{E}[x^{(k)}]}{\sqrt{\mathbb{D}[x^{(k)}]}}$
- ▶ Статистики $\mathbb{E}x$ и $\mathbb{D}x$ оценим для каждого мини-батча
- ? Почему этот метод плох для сетей с сигмоидами?

⁵<https://arxiv.org/abs/1502.03167>

Батч-нормализация⁵

- ▶ Covariate shift: изменение распределения входов во время обучения
- ▶ Цель — уменьшить covariate shift скрытых слоев
- ▶ Нормализуем входы в каждый слой $\hat{x}^{(k)} = \frac{x^{(k)} - \mathbb{E}[x^{(k)}]}{\sqrt{\mathbb{D}[x^{(k)}]}}$
- ▶ Статистики $\mathbb{E}x$ и $\mathbb{D}x$ оценим для каждого мини-батча
- ? Почему этот метод плох для сетей с сигмоидами?
- ▶ Сигмоиды становятся почти линейными \Rightarrow линейная модель :)
- ▶ Доп. параметры: $y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)}$

⁵<https://arxiv.org/abs/1502.03167>

Алгоритм

Входы: Значения x в мини-батче $\mathcal{B} = \{x_i\}_{i=1}^m$;

Параметры: γ, β

Выход: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ среднее мини-батча}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m-1} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ дисперсия мини-батча}$$

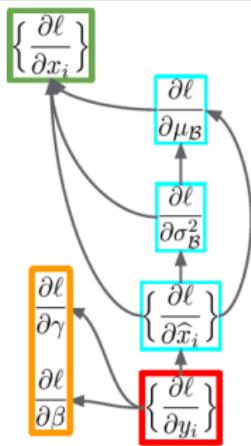
$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ нормализация}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ растяжение и сдвиг}$$

Градиент

Можно вычислить градиент при помощи chain rule

Важно помнить, что μ_B и σ_B^2 не являются константами



$$\left\{ \frac{\partial \ell}{\partial x_i} \right\} = \frac{\partial \ell}{\partial y_i} \cdot \gamma$$

$$\frac{\partial \ell}{\partial \sigma_B^2} = \sum_{i=1}^m \frac{\partial \ell}{\partial \hat{x}_i} \cdot (x_i - \mu_B) \cdot \frac{-1}{2} (\sigma_B^2 + \epsilon)^{-3/2}$$

$$\frac{\partial \ell}{\partial \mu_B} = \left(\sum_{i=1}^m \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{-1}{\sqrt{\sigma_B^2 + \epsilon}} \right) + \frac{\partial \ell}{\partial \sigma_B^2} \cdot \frac{\sum_{i=1}^m -2(x_i - \mu_B)}{m-1}$$

$$\frac{\partial \ell}{\partial x_i} = \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{1}{\sqrt{\sigma_B^2 + \epsilon}} + \frac{\partial \ell}{\partial \sigma_B^2} \cdot \frac{2(x_i - \mu_B)}{m-1} + \frac{\partial \ell}{\partial \mu_B} \cdot \frac{1}{m}$$

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^m \frac{\partial \ell}{\partial y_i} \cdot \hat{x}_i$$

$$\frac{\partial \ell}{\partial \beta} = \sum_{i=1}^m \frac{\partial \ell}{\partial y_i}$$

Предсказание

Во время предсказания батч-нормализация является линейным слоем:

$$\hat{x} = \frac{x - \mathbb{E}[x]}{\sqrt{\mathbb{D}[x] + \epsilon}}$$

$$y = \gamma \cdot \hat{x} + \beta$$

$$y = \frac{\gamma}{\sqrt{\mathbb{D}[x] + \epsilon}} \cdot x + \left(\beta - \frac{\gamma \mathbb{E}[x]}{\sqrt{\mathbb{D}[x] + \epsilon}} \right)$$

$\mathbb{E}[x]$ и $\mathbb{D}[x]$ вычисляются по всему обучающему множеству.

На практике статистики вычисляются во время обучения экспоненциальным средним: $E_{i+1} = (1 - \alpha)E_i + \alpha E_B$

Batchnorm как регуляризация

- ▶ $\frac{\partial BN((aW)u)}{\partial u} = \frac{\partial BN(Wu)}{\partial u}$
- ▶ $\frac{\partial BN((aW)u)}{\partial aW} = \frac{1}{a} \frac{\partial BN(Wu)}{\partial W}$

При увеличении весов в a раз, градиент выхода слоя по входу не меняется, а градиент по весам уменьшается в a раз.

Tips

Стоит помнить, что с батч-нормализацией:

- ▶ Надо убрать смещения
- ▶ Другое расписание learning rate: большее значение в начале обучения и быстрое уменьшение в процессе обучения
- ▶ Уменьшить силу Dropout и L_2 регуляризации
- ▶ Перемешивать обучающую выборку

Для изображений: нормализация каждого канала (одинаковые среднее и дисперсия вдоль пространственных размерностей)

Обучение

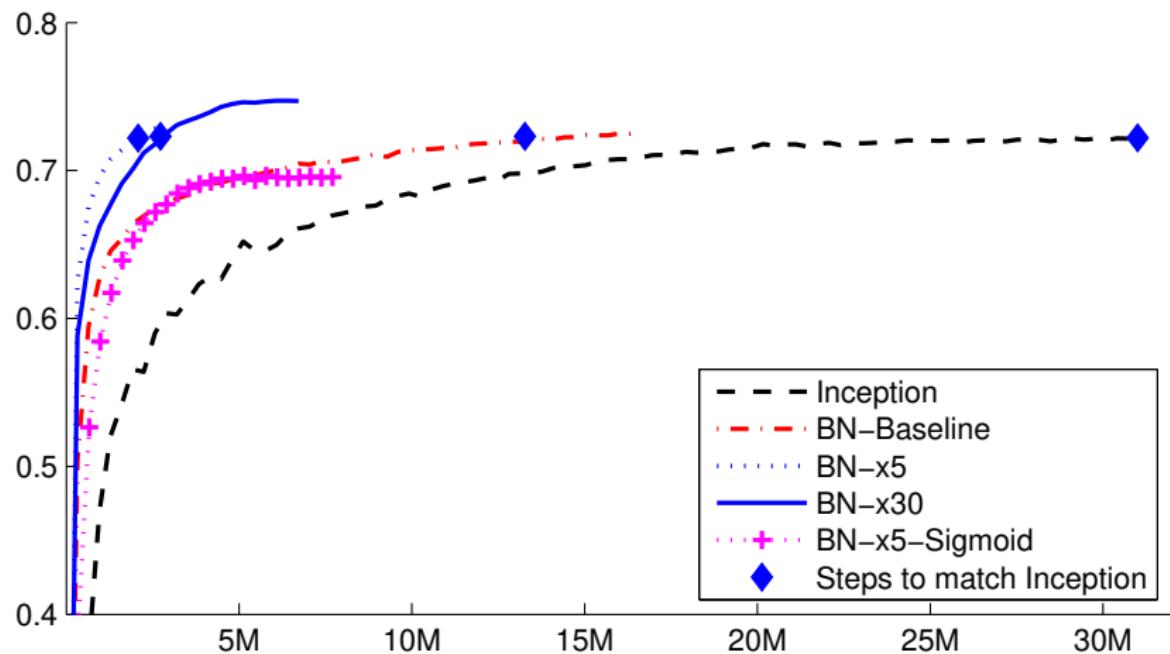
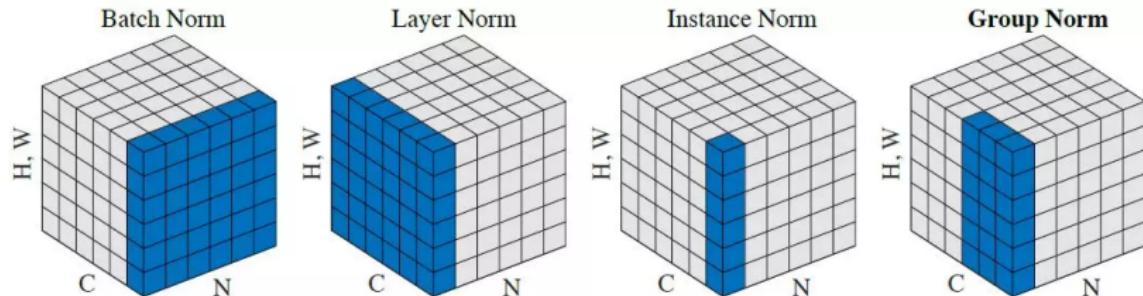


Figure: Обучение Inception с и без батч-нормализации.⁶

⁶x30 — увеличение темпа обучения в 30 раз

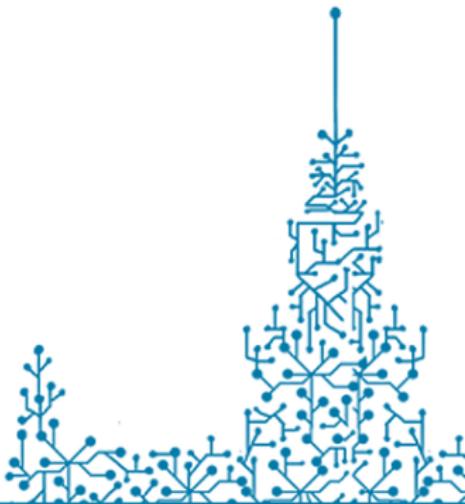
Layer Normalization ⁷



	Weight matrix re-scaling	Weight matrix re-centering	Weight vector re-scaling	Dataset re-scaling	Dataset re-centering	Single training case re-scaling
Batch norm	Invariant	No	Invariant	Invariant	Invariant	No
Weight norm	Invariant	No	Invariant	No	No	No
Layer norm	Invariant	Invariant	No	Invariant	No	Invariant

⁷<https://arxiv.org/abs/1607.06450>

Вопросы





ТЕХНОСФЕРА

Лекция 6 Архитектуры глубинных сетей

Полыковский Даниил
Храбров Кузьма

17 марта 2021 г.

Архитектуры CNN

ImageNet, описание



- ▶ 1000 классов
- ▶ около 1000 изображений в каждом классе
- ▶ около 1 000 000 изображений всего
- ▶ несколько номинаций: таких как распознавание и детектирование/локализация

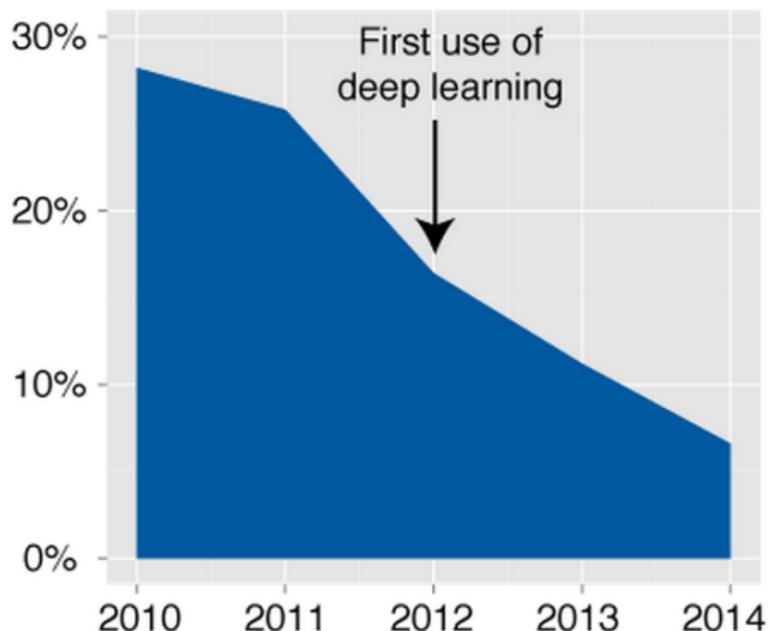
ImageNet, правила

mite mite black widow cockroach tick starfish	container ship container ship lifeboat amphibian fireboat drilling platform	motor scooter motor scooter go-kart moped bumper car golfcart	leopard leopard jaguar cheetah snow leopard Egyptian cat
grille convertible grille pickup beach wagon fire engine	mushroom agaric mushroom jelly fungus gill fungus dead-man's-fingers	cherry dalmatian grape elderberry fordshire bullterrier currant	Madagascar cat squirrel monkey spider monkey titi indri howler monkey

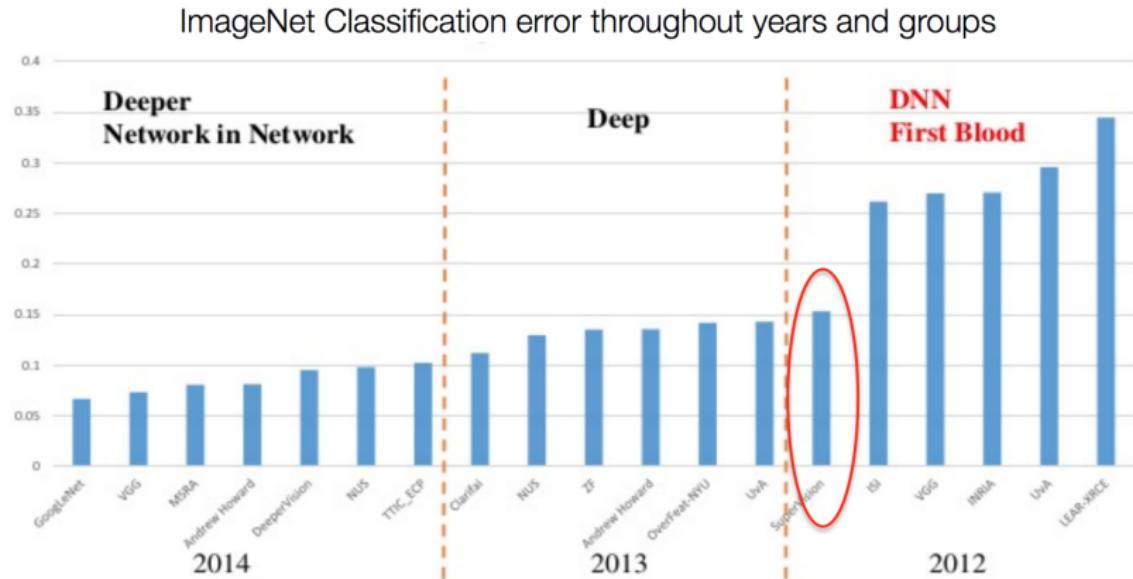
Рис.: Примеры прогнозов

ImageNet, прогресс

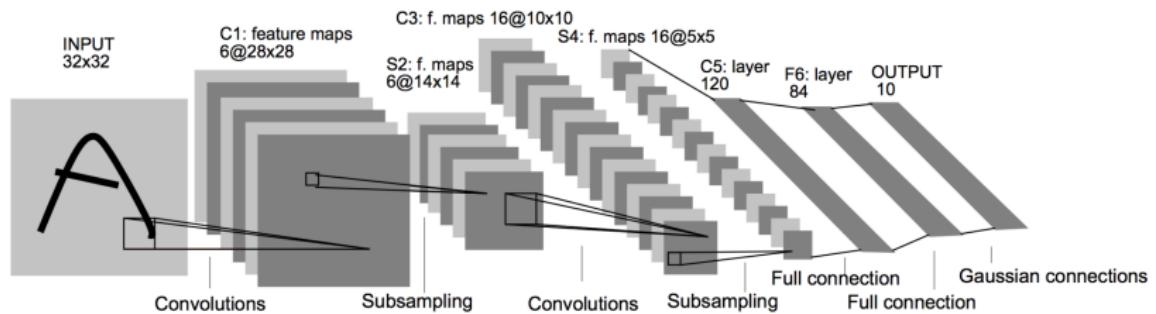
Objection classification error rate



ImageNet, прогресс

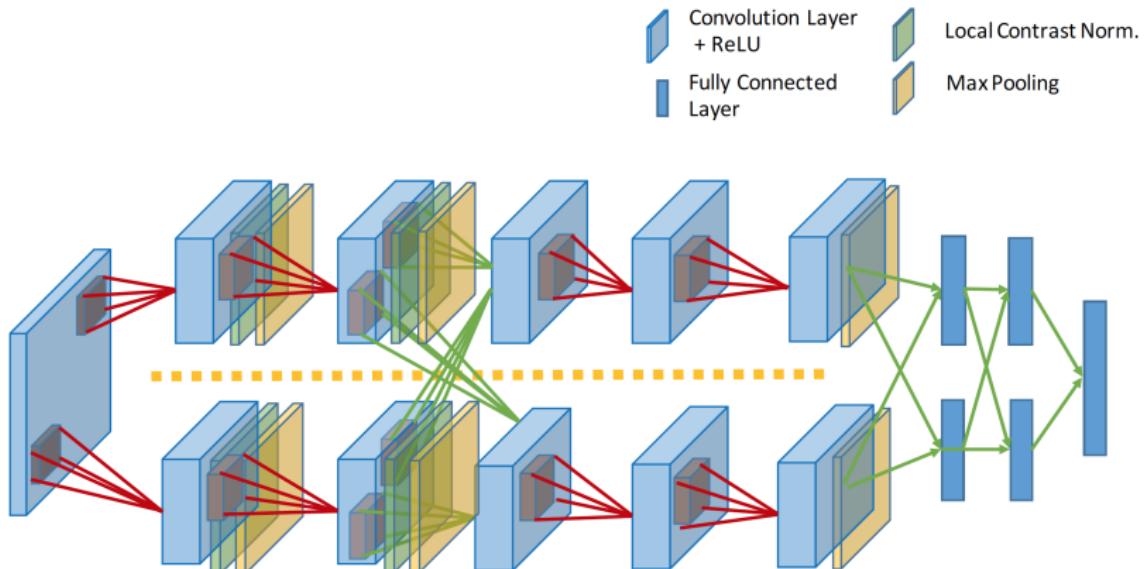


Lenet¹



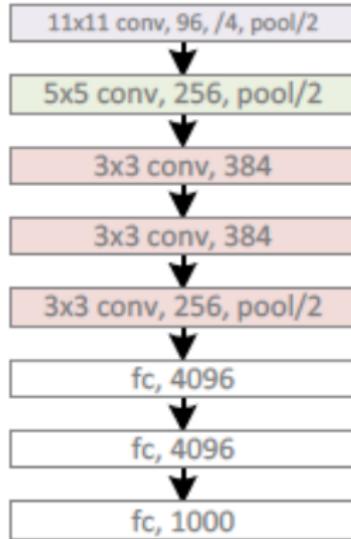
¹<http://yann.lecun.com/exdb/lenet>

AlexNet²



²Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton: ImageNet Classification with Deep Convolutional Neural Networks, 2012

<http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf>



- ▶ 5 сверточных и 3 полносвязных слоя
- ▶ 60M параметров, 650к нейронов
- ▶ эффективное распараллеливание на 2 GPU/CUDA
- ▶ свёртки 11x11, 5x5, 3x3
- ▶ ReLU, т.к. не надо вычислять exp

AlexNet

Local response normalization

$$b_{x,y}^i = a_{x,y}^i / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a_{x,y}^j)^2 \right)^\beta$$

Аугментации: случаный 224x224 кроп, горизонтальный поворот, добавление хитрого шума к RGB-каналам

$$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1 \lambda_1, \alpha_2 \lambda_2, \alpha_3 \lambda_3]^T$$

где p_i и λ_i - главные компоненты и собственные значения, соответственно, а $\alpha_i \sim \mathcal{N}(0, 0.1)$

Model	Top-1	Top-5
<i>Sparse coding [2]</i>	47.1%	28.2%
<i>SIFT + FVs [24]</i>	45.7%	25.7%
CNN	37.5%	17.0%

Table 1: Comparison of results on ILSVRC-2010 test set. In *italics* are best results achieved by others.

VGG³

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256	conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256
conv3-256	conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256
maxpool					
conv3-512	conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512
conv3-512	conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512
maxpool					
conv3-512	conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512
conv3-512	conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

► VGG-19 (E):

144M

параметров

► Very Deep Convolutional Networks for Large-Scale Image Recognition^a

► Только свертки 3x3

^aK. Simonyan, A. Zisserman

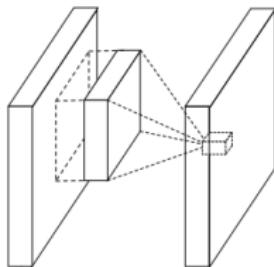
³<http://arxiv.org/pdf/1409.1556.pdf>

VGG

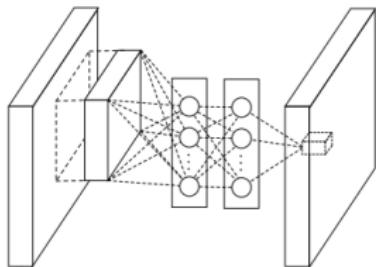
Table 7: **Comparison with the state of the art in ILSVRC classification.** Our method is denoted as “VGG”. Only the results obtained without outside training data are reported.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	23.7	6.8	6.8
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-		7.9
GoogLeNet (Szegedy et al., 2014) (7 nets)	-		6.7
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

NIN⁴

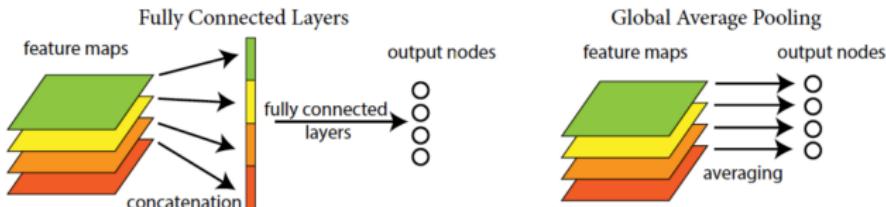


(a) Linear convolution layer



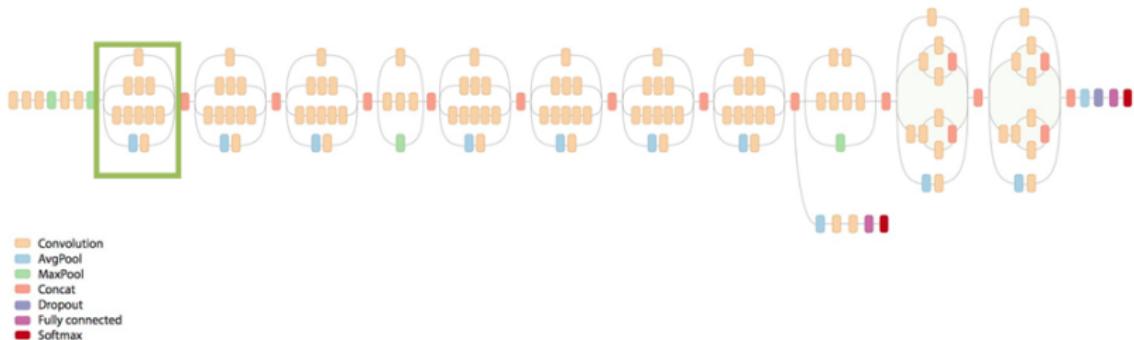
(b) Mlpconv layer

- ▶ “Полносвязные слои” (свертки 1×1) внутри свертки
- ▶ Глобальный пулинг для вытягивания изображения (+ доп. регуляризация)

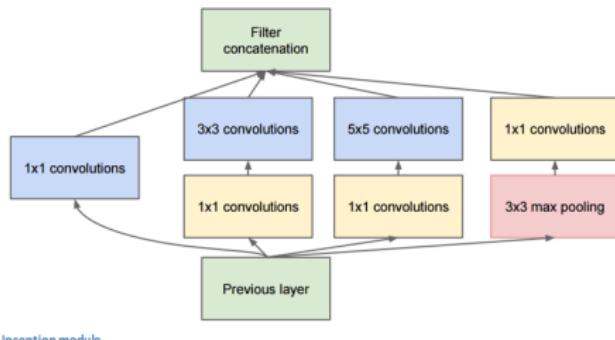


⁴<https://arxiv.org/pdf/1312.4400v3.pdf>

GoogLeNet⁵

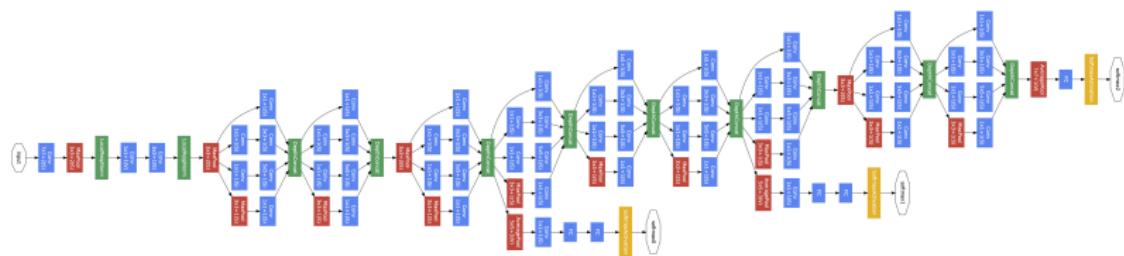


Green box shows parallel region of GoogLeNet



⁵ http://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Szegedy_Going_Deeper_With_2015_CVPR_paper.pdf

GoogLeNet



Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)



VGG, 19 layers
(ILSVRC 2014)



ResNet, 152 layers
(ILSVRC 2015)



- ▶ Нагляднее: [http://josephpcohen.com/w/
visualizing-cnn-architectures-side-by-side-with-mxnet/](http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/)

ResNet⁶

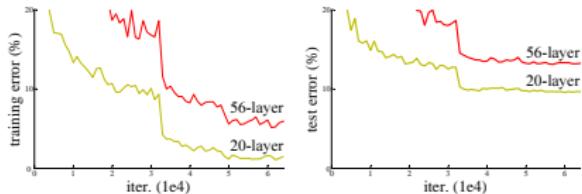


Рис.: Обучение “обычной” сети

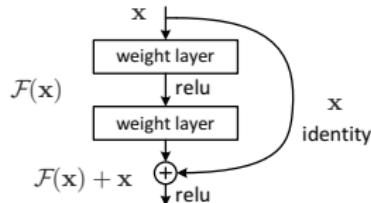


Рис.: Residual block

- ▶ Результат на 56 слоях хуже. Проблема не в переобучении
- ▶ Решение заведомо существует: 20 слоев, затем $\mathcal{F}(x)=x$
- ▶ Выучить $\mathcal{F}(x)=x$ тяжело, а $\mathcal{F}(x)=0$ просто
- ▶ Residual block решает эту проблему

⁶<https://arxiv.org/pdf/1512.03385v1.pdf>

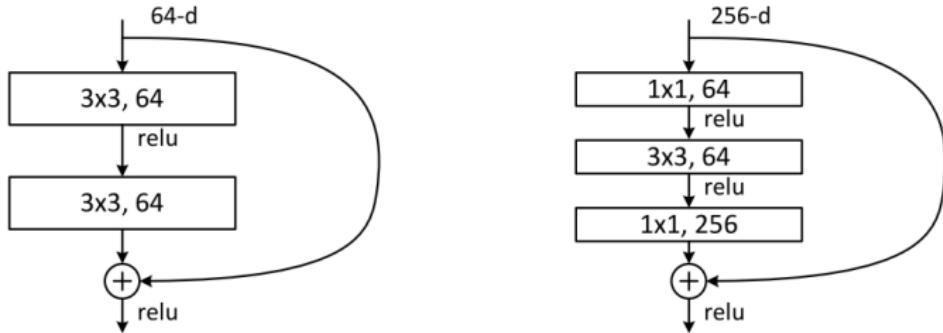


Figure 5. A deeper residual function \mathcal{F} for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a “bottleneck” building block for ResNet-50/101/152.

Модификации моделей

Существует множество модификаций ResNet и Inception: Inception v2(3), ResNet in ResNet⁷, DenseNet⁸, ResNeXt, WideResNets⁹

⁷<https://arxiv.org/pdf/1603.08029v1.pdf>

⁸<https://arxiv.org/pdf/1608.06993v1.pdf>

⁹<https://arxiv.org/abs/1605.07146>

Модификации Inception

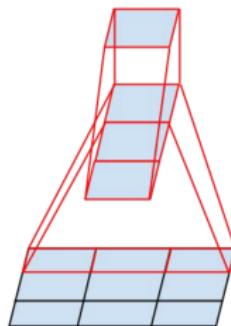


Figure 3. Mini-network replacing the 3×3 convolutions. The lower layer of this network consists of a 3×1 convolution with 3 output units.

Рис.: Факторизация свертки 3×3

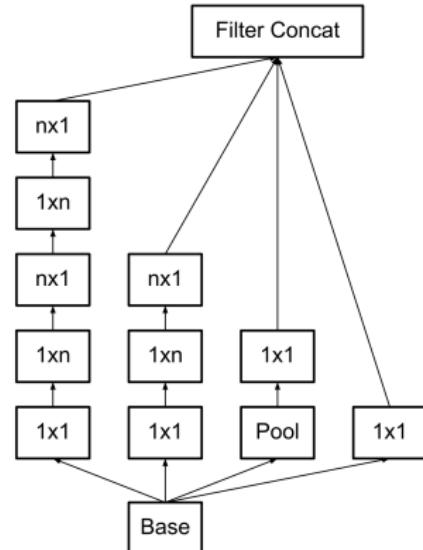


Figure 6. Inception modules after the factorization of the $n \times n$ convolutions. In our proposed architecture, we chose $n = 7$ for the 17×17 grid. (The filter sizes are picked using principle 3)

Рис.: Inception block with factorization

DenseNet

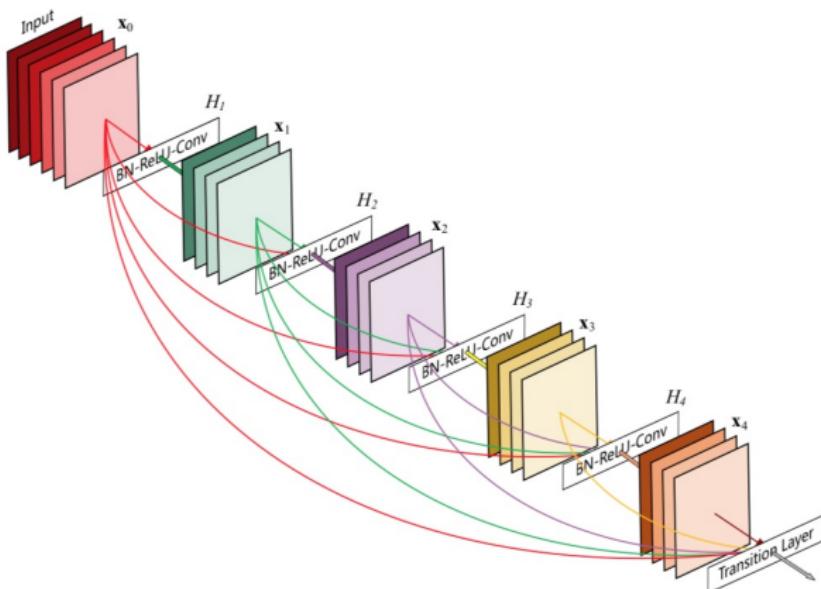


Figure 1: A 5-layer dense block with a growth rate of $k = 4$.
Each layer takes all preceding feature-maps as input.

Рис.: <https://arxiv.org/abs/1608.06993>

ResNeXt

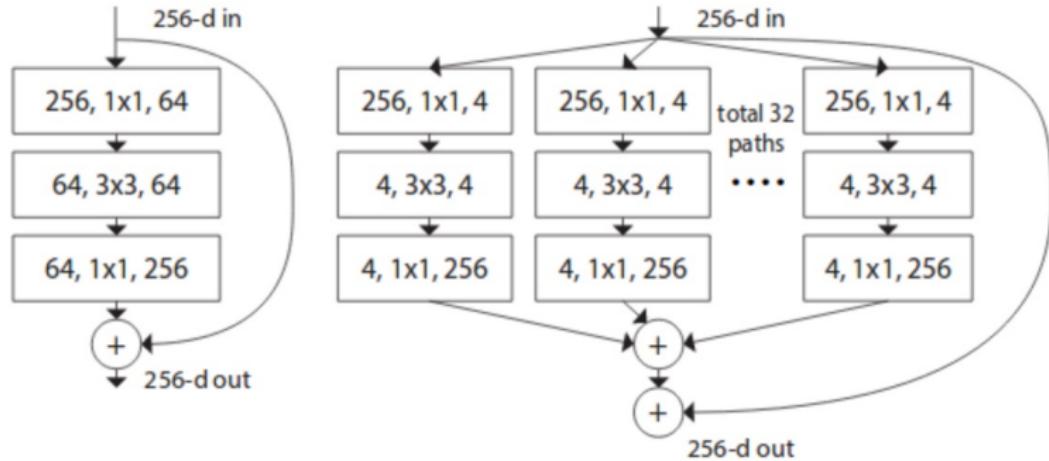


Figure 1. **Left:** A block of ResNet [14]. **Right:** A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).

ResNet (left) and ResNeXt (right) Architecture.

Сравнение моделей

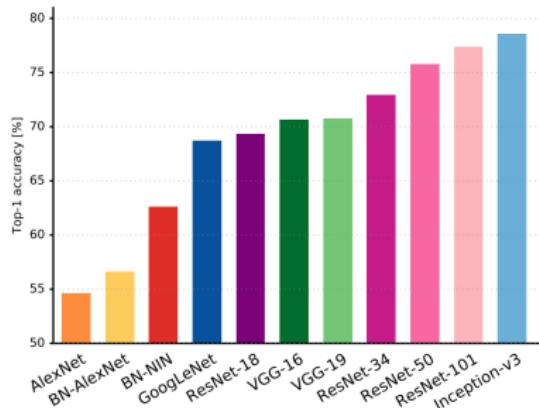


Figure 1: **Top1 vs. network.** Single-crop top-1 validation accuracies for top scoring single-model architectures. We introduce with this chart our choice of colour scheme, which will be used throughout this publication to distinguish effectively different architectures and their correspondent authors. Notice that network of the same group share colour, for example ResNet are all variations of pink.

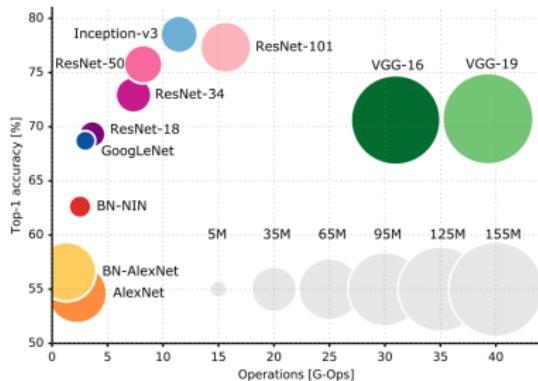


Figure 2: **Top1 vs. operations, size \propto parameters.** Top-1 one-crop accuracy versus amount of operations required for a single forward pass. The size of the blobs is proportional to the number of network parameters; a legend is reported in the bottom right corner, spanning from 5×10^6 to 155×10^6 params.

Рис.: Accuracy

Сравнение моделей

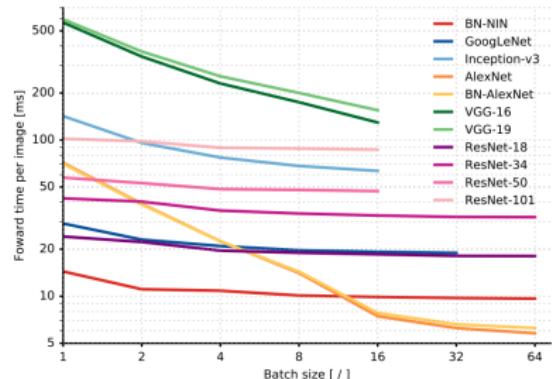
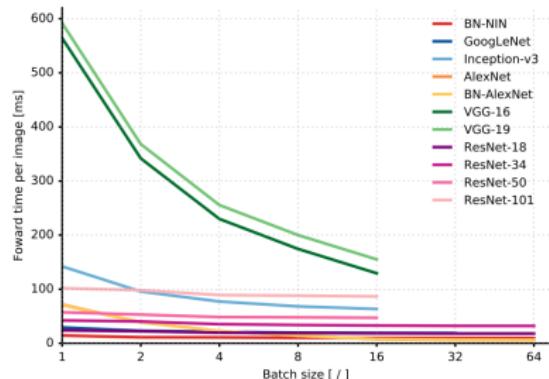


Figure 3: **Inference time vs. batch size.** These two charts show inference time across different batch sizes with a linear and logarithmic ordinate respectively and logarithmic abscissa. Missing data points are due to lack of enough system memory required to process bigger batches.

Рис.: Скорость

Сравнение моделей

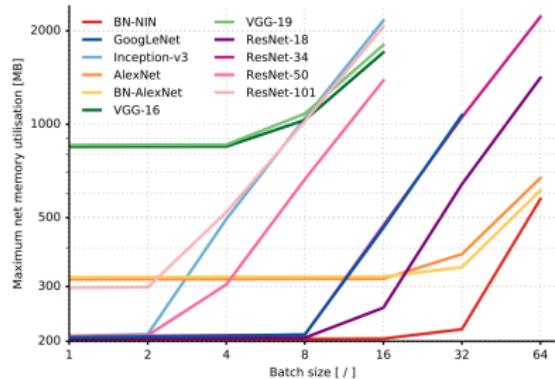


Figure 5: **Memory vs. batch size.** Maximum system memory utilisation for batches of different sizes. Memory usage shows a knee graph, due to the network model memory static allocation and the variable memory used by batch size.

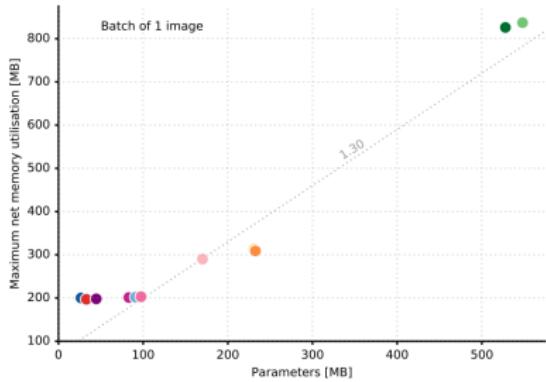
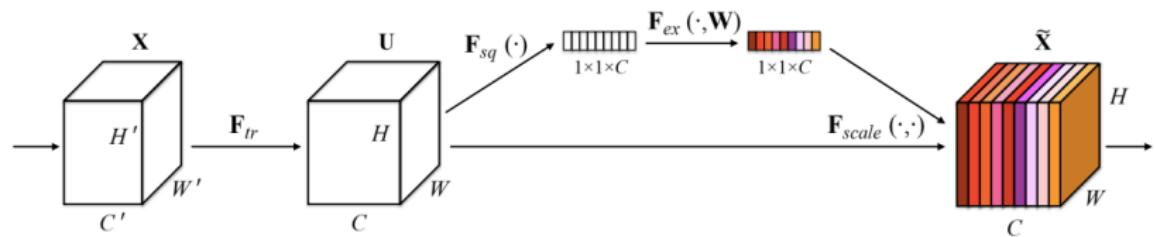


Figure 6: **Memory vs. parameters count.** Detailed view on static parameters allocation and corresponding memory utilisation. Minimum memory of 200 MB, linear afterwards with slope 1.30.

Рис.: Потребление памяти

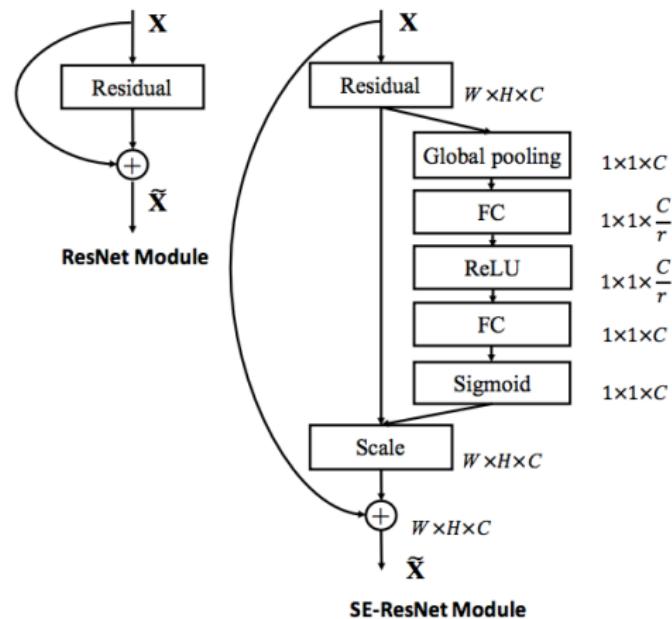
Squeeze-and-Excitation Networks



10

¹⁰<https://arxiv.org/abs/1709.01507>

Squeeze-and-Excitation Networks



MobileNet



Figure 1. MobileNet models can be applied to various recognition tasks for efficient on device intelligence.

11

¹¹<https://arxiv.org/pdf/1704.04861.pdf>,
<https://arxiv.org/pdf/1801.04381.pdf>

MobileNet. Depthwise separable convolution

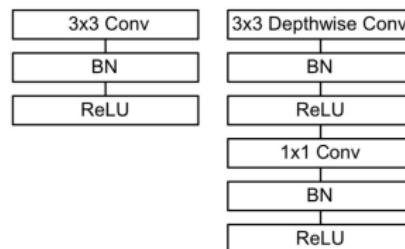
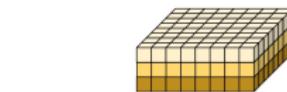
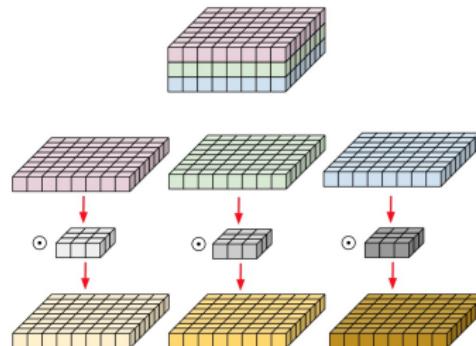
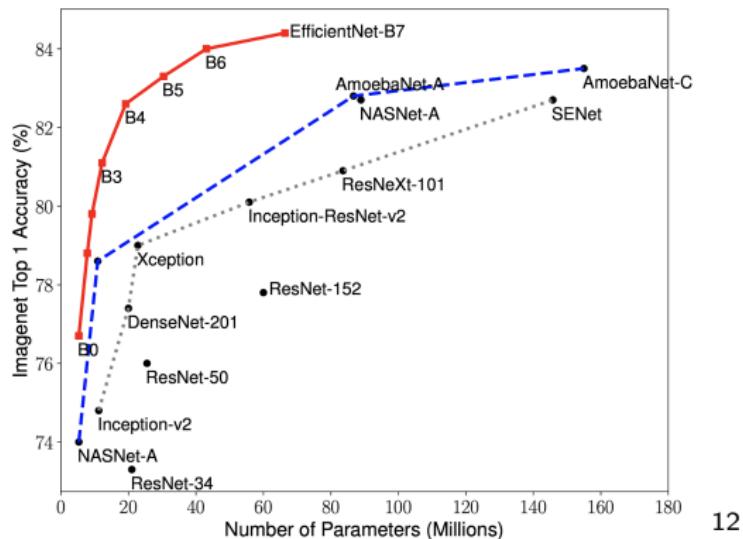


Figure 3. Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

Latest results



12

12 <https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet>

Case study

National Data Science Bowl

National Data Science Bowl¹⁴



Рис.: Примеры изображений из набора данных: всего 121 несбалансированный класс, 30 000 изображений

Победители¹³ использовали архитектуру VGG-16, а так же ряд специальных трюков.

¹³ <http://benanne.github.io/2015/03/17/plankton.html>

¹⁴ <https://www.kaggle.com/c/datasciencebowl>

Data augmentation, #1

Data augmentation:

- ▶ rotation: random with angle between 0 and 360 degree (uniform)
- ▶ translation: random with shift between -10 and 10 pixels (uniform)
- ▶ rescaling: random with scale factor between 1/1.6 and 1.6 (log-uniform)
- ▶ flipping: yes or no (bernoulli)
- ▶ shearing: random with angle between -20 and 20 degree (uniform)
- ▶ stretching: random with stretch factor between 1/1.3 and 1.3 (log-uniform)

Данные нужно аугментировать в реальном времени (иначе — значительное увеличение размера датасета). Не стоит недооценивать test-time аугментацию.

Data augmentation, #2

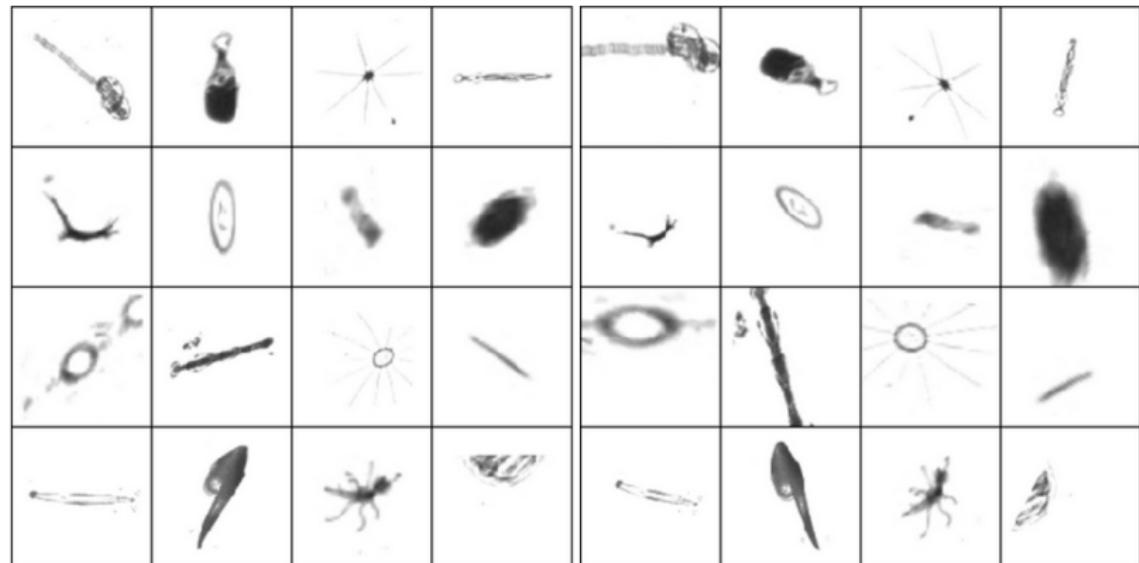


Рис.: Оригиналы и аугментированные образы

Cyclic pooling

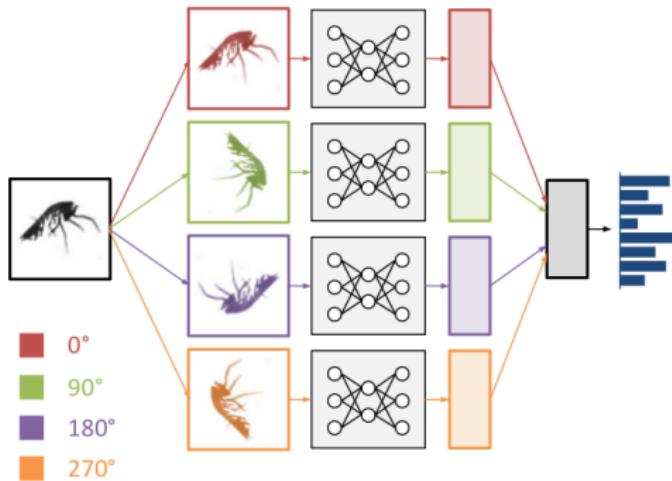


Рис.: Schematic representation of a convnet with cyclic pooling

- ▶ объединение результатов извлечения признаков после прохода по нескольким копиям одной сети параллельно
- ▶ это позволило сократить размер батча в 4 раза (со 128 до 32)
- ▶ root-mean-square pooling оказался эффективнее других

Rolling feature maps

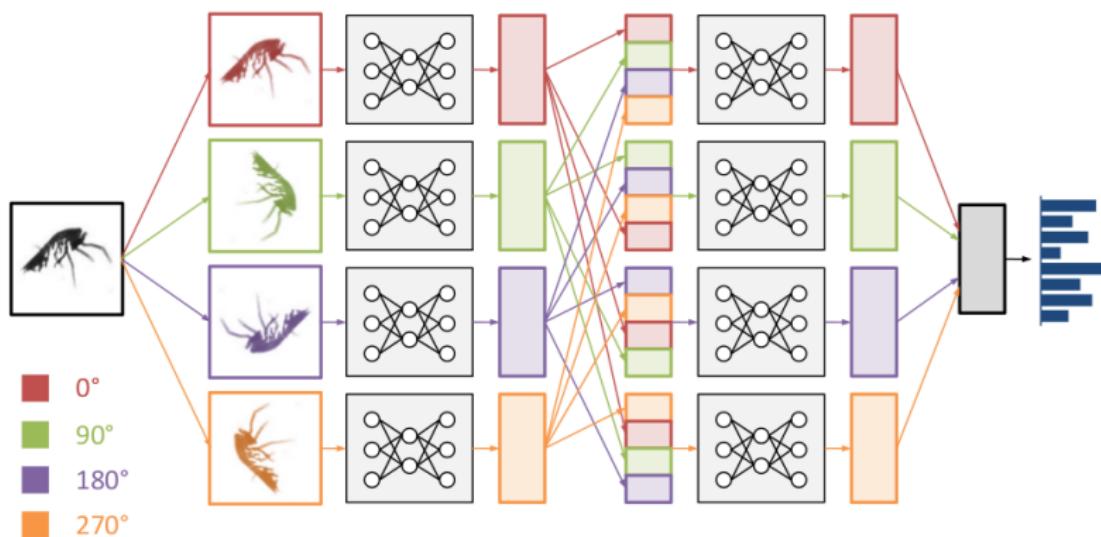


Рис.: Schematic representation of a roll operation inside a convnet with cyclic pooling.

Советы

- ▶ leaky/parameterized ReLU: $f(x) = \max(x, a \cdot x)$, $a \leq 0$
- ▶ добавление к сверточным признакам других признаков перед полносвязным слоем (Hu moments, Zernike moments, atc)
- ▶ использовать предобученные модели
- ▶ self-training с тестовой выборкой
- ▶ аугментировать данные

Другие задачи компьютерного зрения

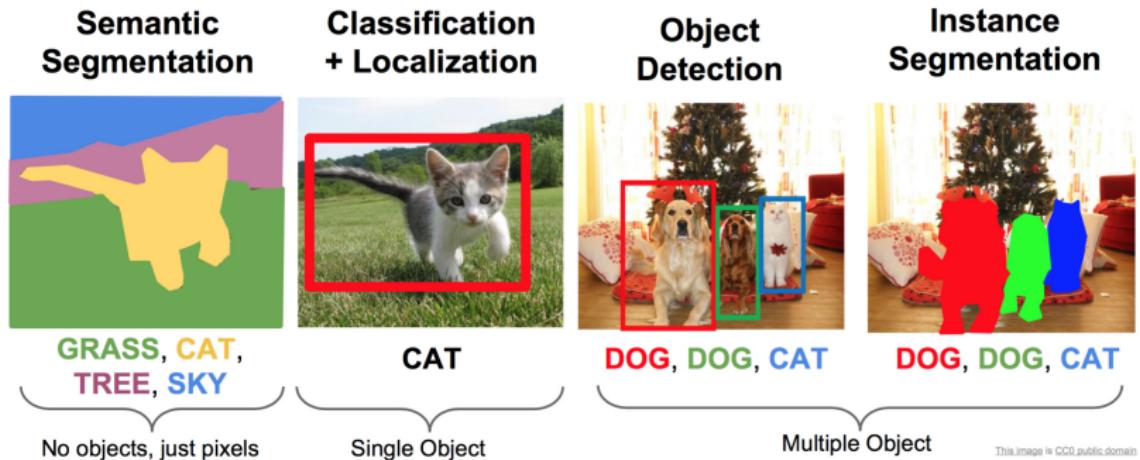
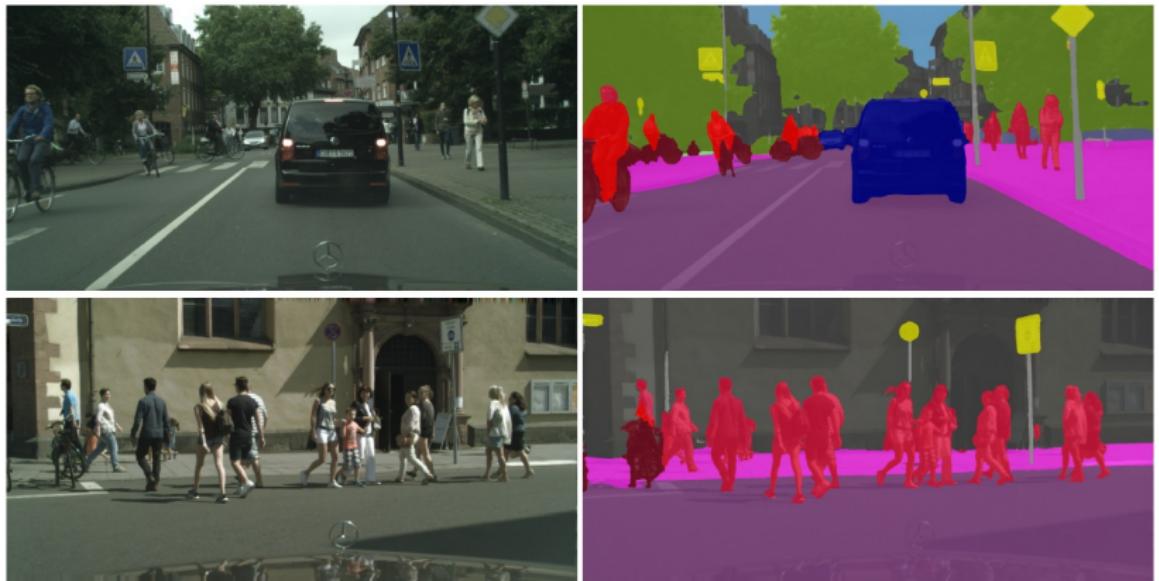


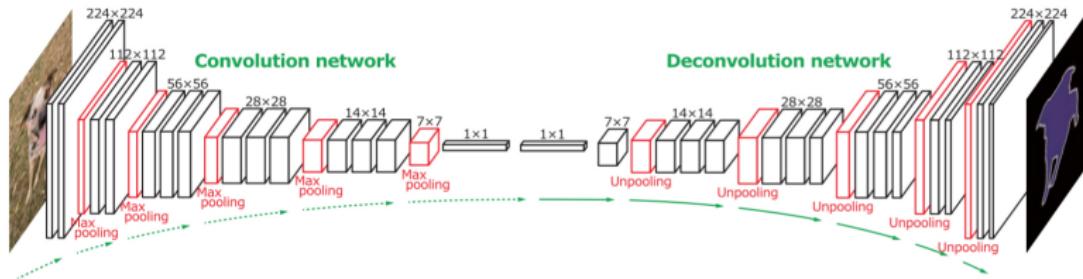
Рис.: Semantic segmentation, Localization, Detection and Instance Segmentation (Source: CS231n (Stanford) lecture slides)

Semantic segmentation



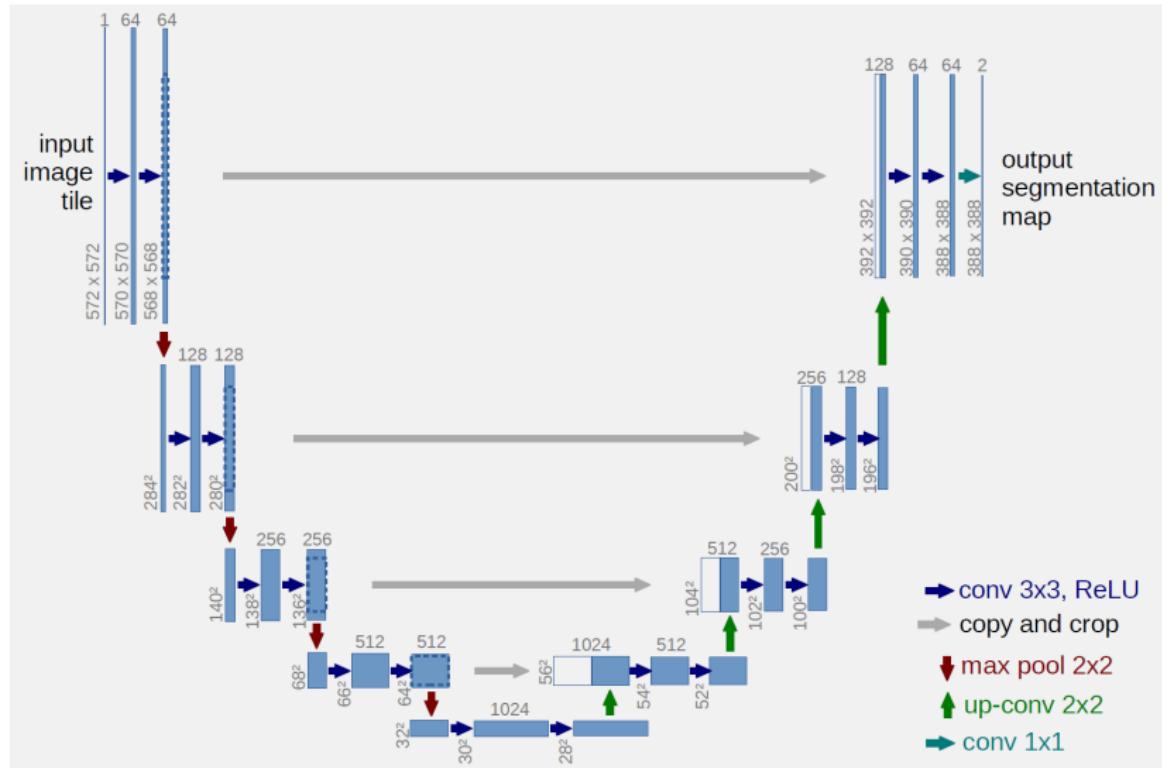
Задача - понять к какой категории относится каждая точка картинки.

Semantic segmentation. Подходы. 15



15 <https://arxiv.org/abs/1505.04366v1>

U-NET¹⁶



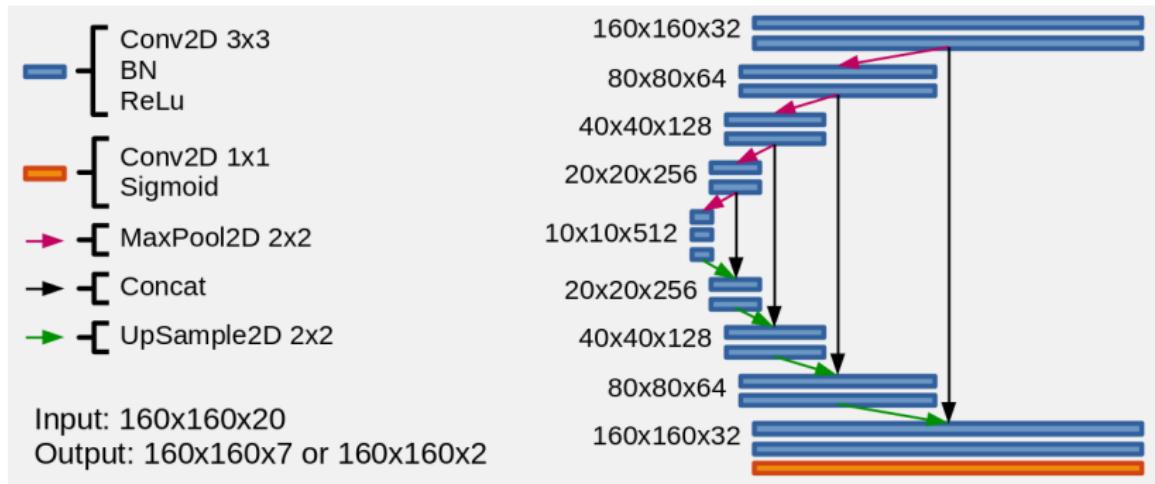
¹⁶<https://arxiv.org/abs/1505.04597>

U-NET

Задача:

- ▶ 25 спутниковых изображений в Train
- ▶ 425 спутниковых изображений в Test
- ▶ 10 классов объектов
- ▶ Изображения участков поверхности земли 1x1 км

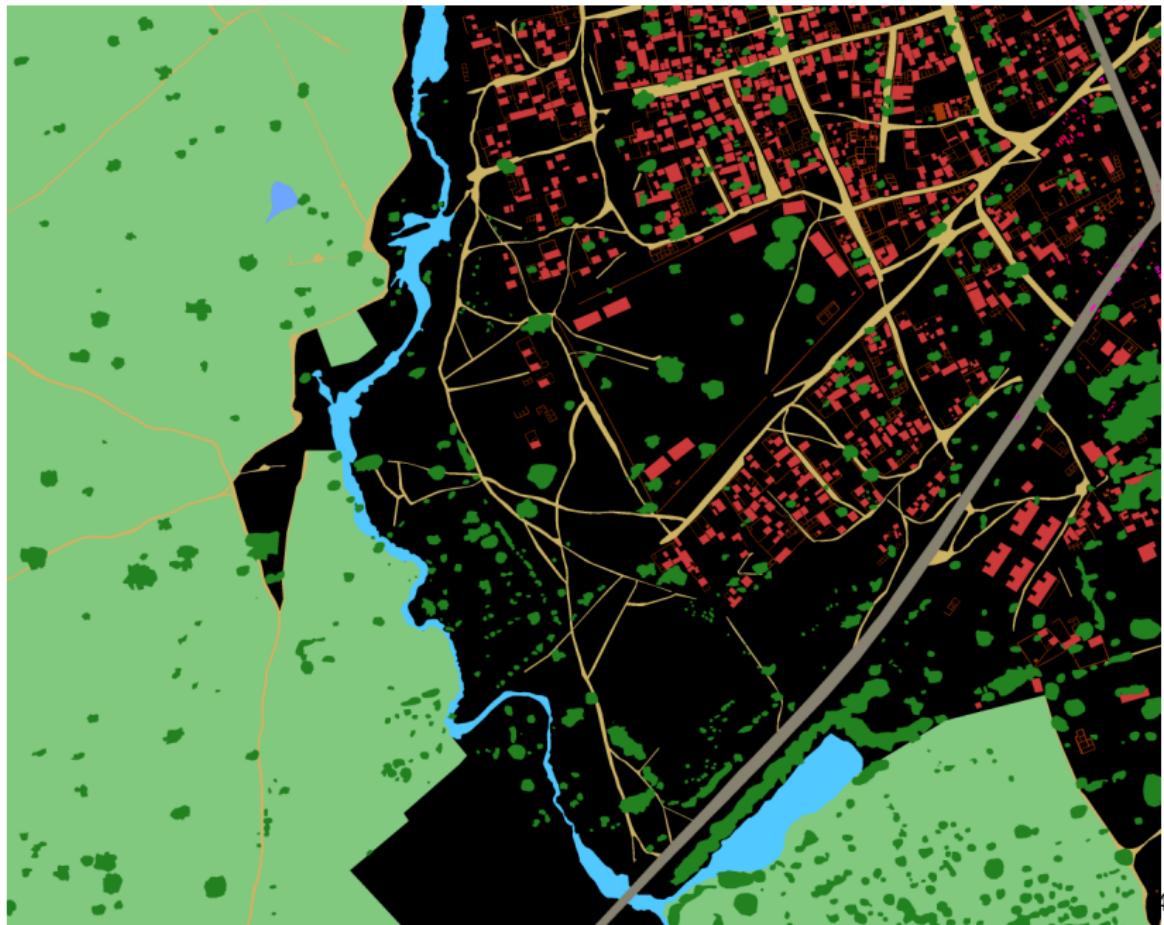
U-NET



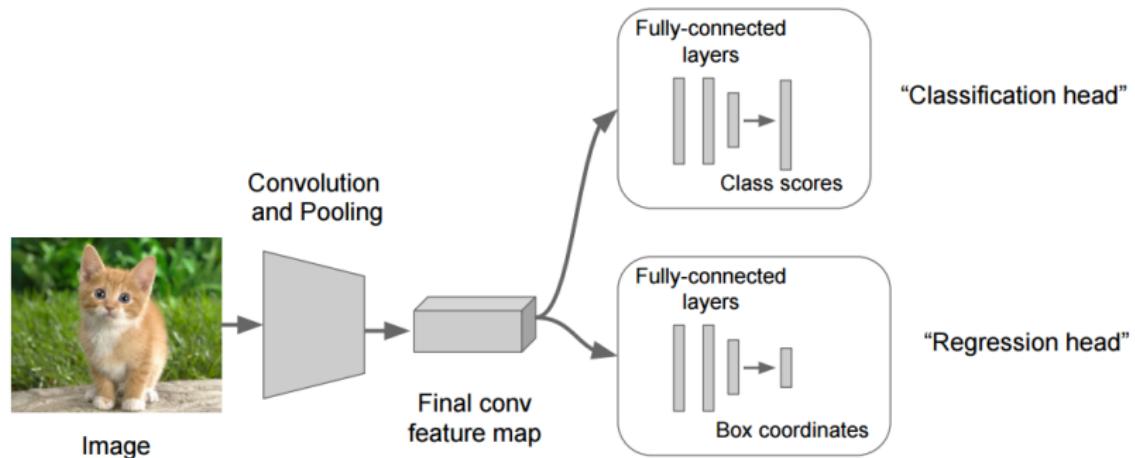
U-NET



U-NET



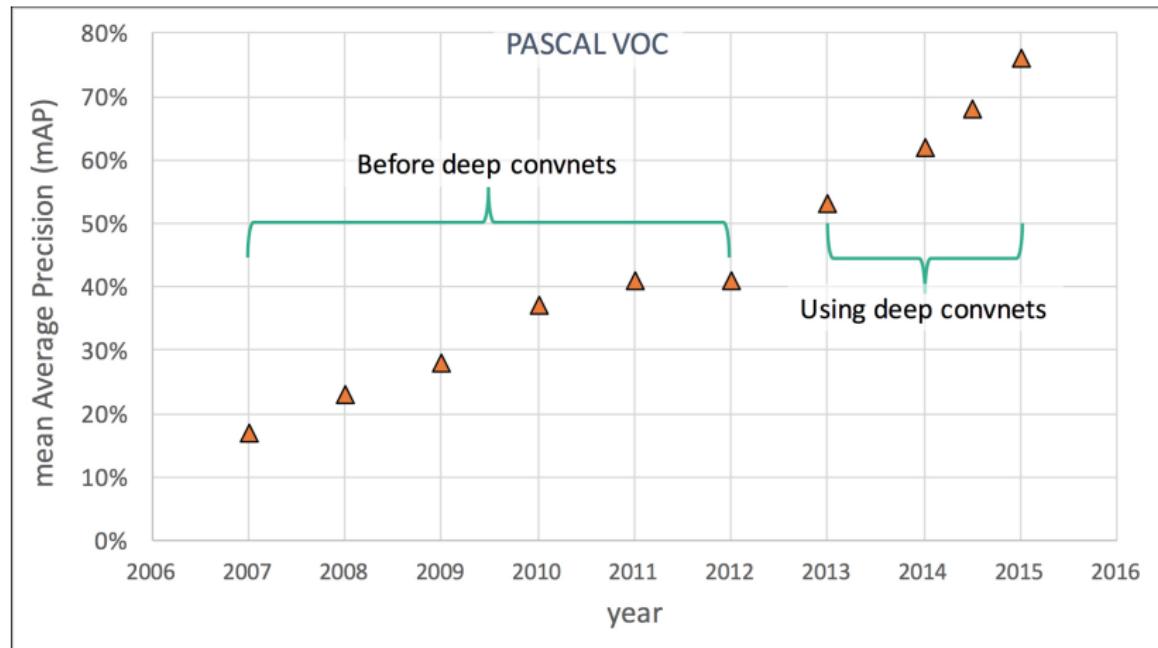
Object Classification + localization.



Object detection. Pascal VOC competition



Object detection. История



Detection. Подходы

- ▶ Регрессия. Так же как и с одним объектом.
Обычно не известно, какие объекты мы ищем и сколько их.
- ▶ Классификация.
Выбираем области, в которых могут быть объекты и применяем классификатор.

Detection. Выбор областей. Скользящее окно



Рис.: Background

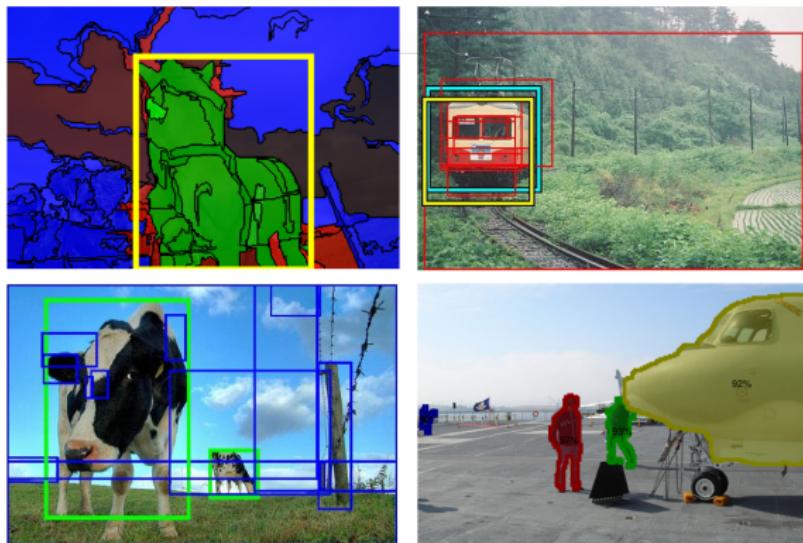


Рис.: Dog



Рис.: Cat

Detection. Автоматический выбор областей.



Shape Sharing for Object Segmentation. J. Kim and K. Grauman. ECCV 2012.

Uijlings et al, "Selective Search for Object Recognition", IJCV 2013

Category-Independent Object Proposals with Diverse Ranking. Ian Endres and Derek Hoiem. PAMI 2014.

Zitnick and Dollar, "Edge boxes: Locating object proposals from edges", ECCV 2014

Detection. R-CNN.

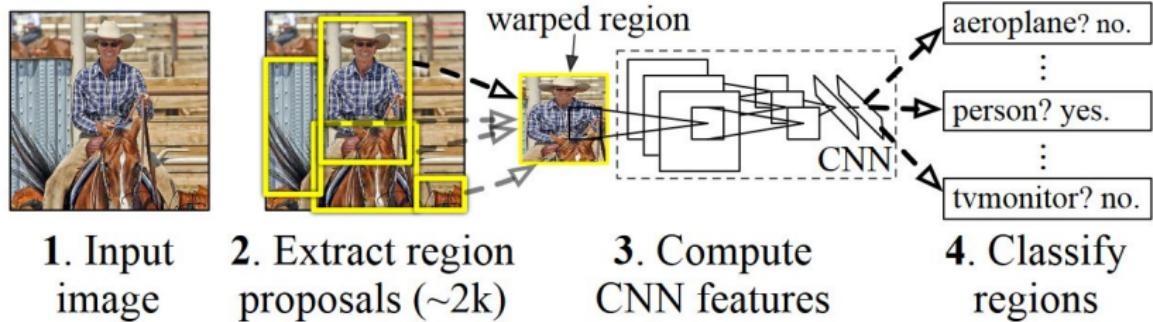


Рис.: R-CNN paper

Detection. Faster.

Fast R-CNN

Faster R-CNN

YOLO

Instance Segmentation. Mask R-CNN

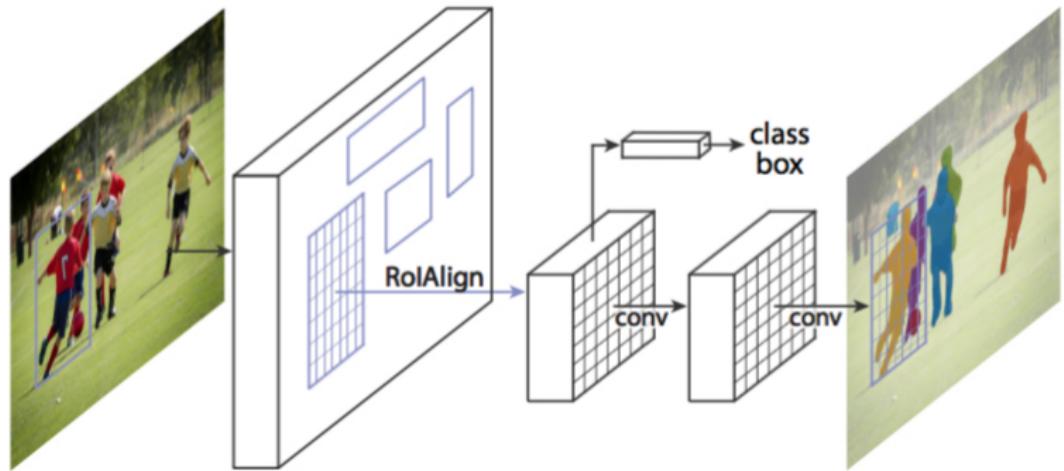
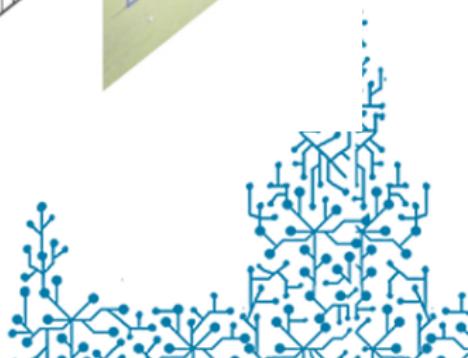


Рис.: Mask R-CNN paper



Вопросы