CNN Architectures,Transfer Learning

Tatiana Gaintseva

План

- Архитектуры сверточных нейросетей, skip connection
- Transfer Learning

CNN Architectures

IMAGENET

14,197,122 images, 21841 synsets indexed

Explore Download Challenges Publications Updates About

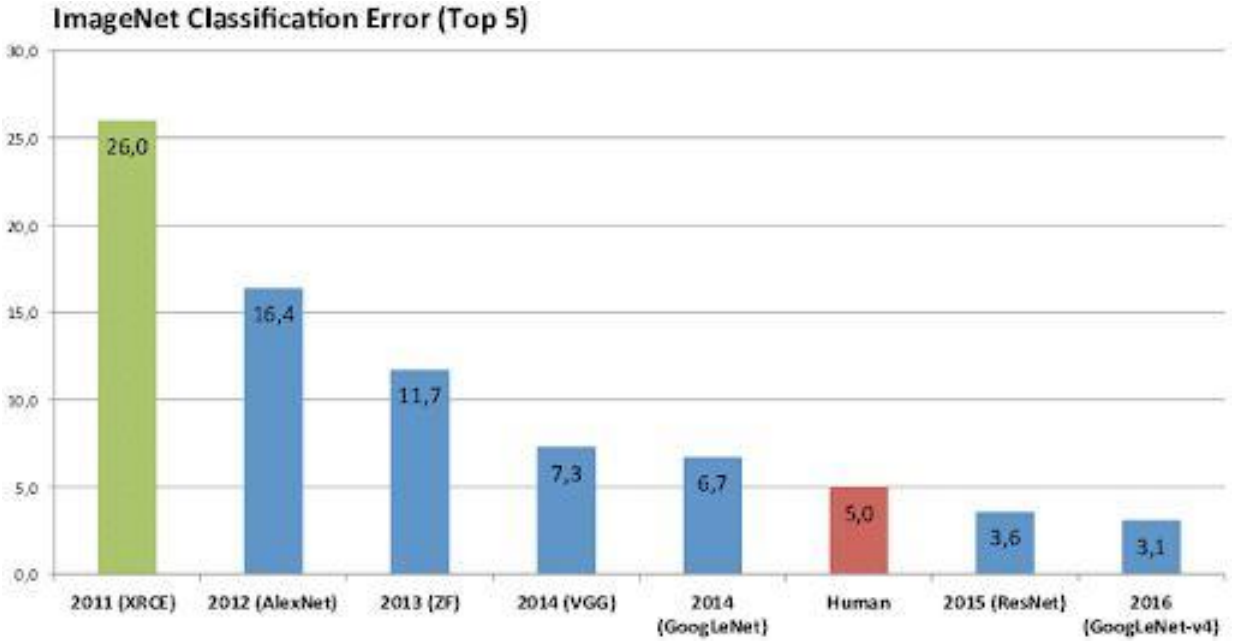
Not logged in. Login | Signup

База данных изображений, поделенных на 1000 классов.

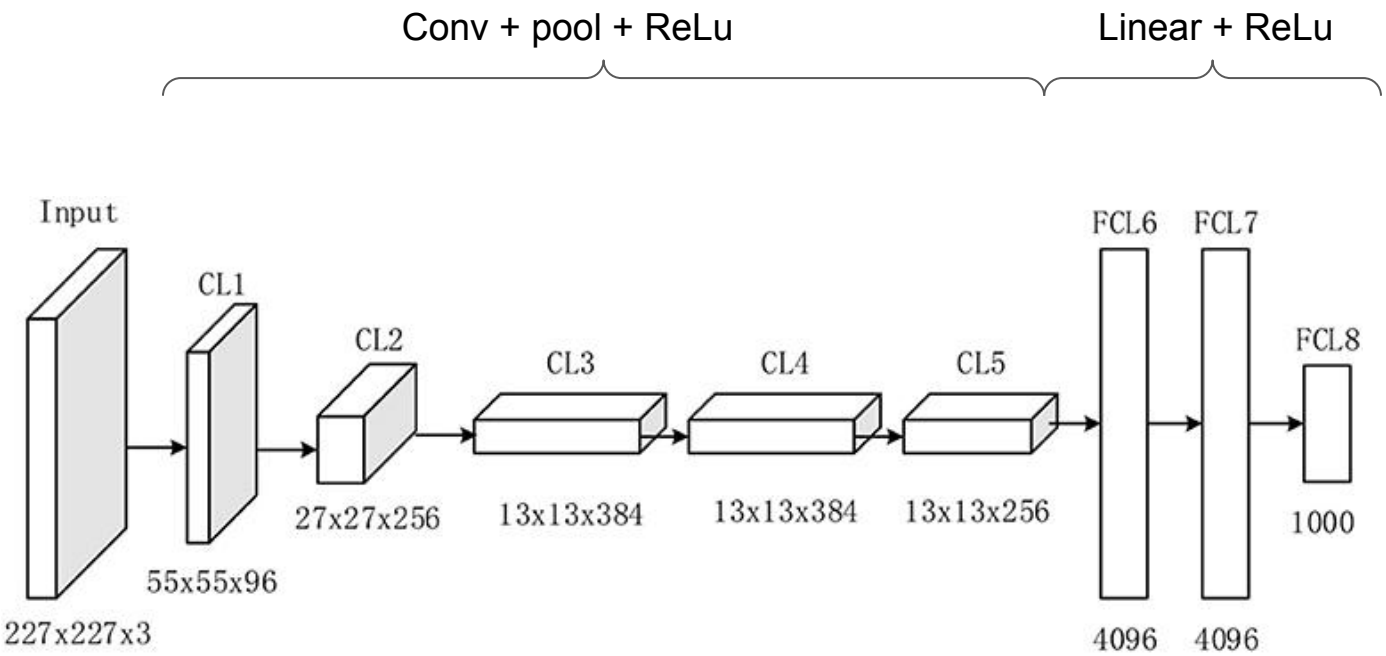
http://www.image-net.org

http://image-net.org/explore

ImageNet timeline

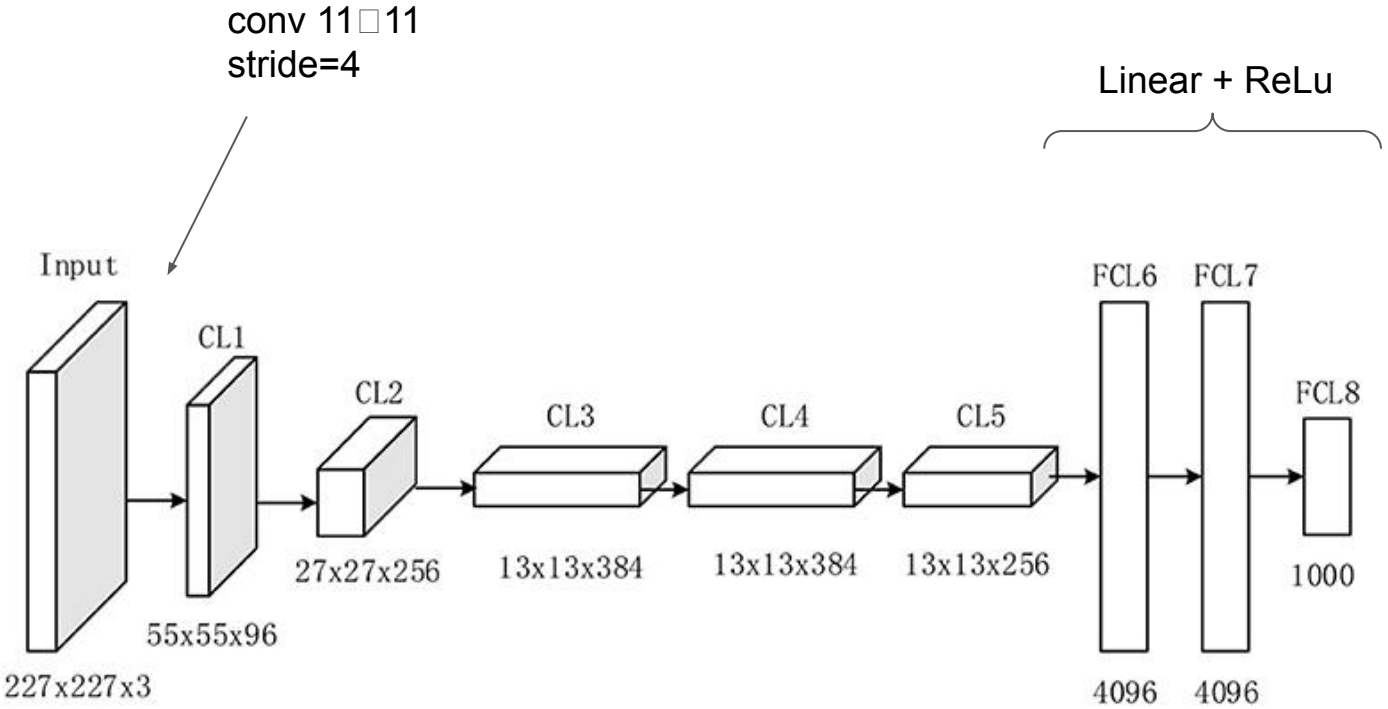


AlexNet



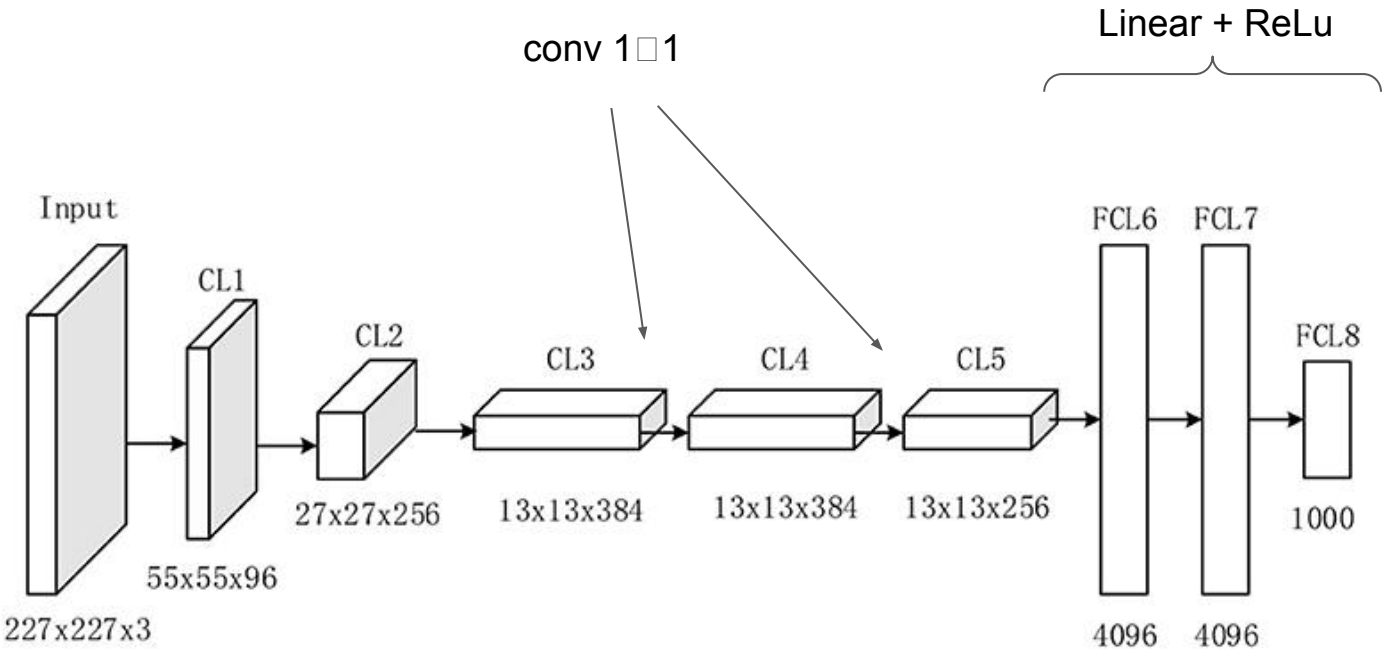
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

AlexNet



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

AlexNet



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

AlexNet

Главные достижения:

- ReLU вместо Tanh
- Parallel training на нескольких GPU
- Data Augmentation

https://github.com/pytorch/vision/blob/master/torchvision/models/alexnet.py

AlexNet -> VGG

- conv слои 3x3 (receptive field) - меньше, чем у AlexNet, что позволяет детектировать более низкоуровневые паттерны
- 3 fc слоя

https://towardsdatascience.com/vgg-neural-networks-the-next-step-after-alexnet-3f91fa9ffe2c

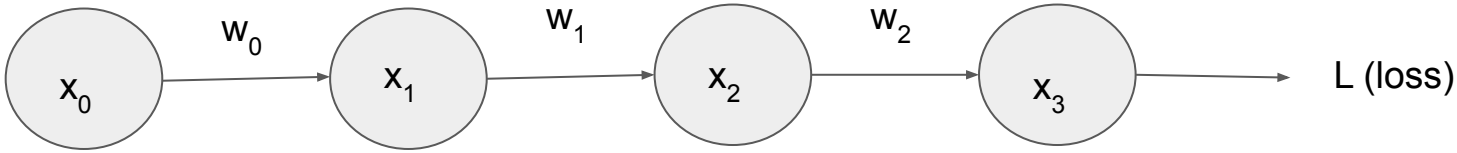
VGG

VGG16 и VGG19 распространены больше всего

Network	Top-1 error	Top-5 error
VGG-11	30.98	11.37
VGG-13	30.07	10.75
VGG-16	28.41	9.62
VGG-19	27.62	9.12
VGG-11 with batch normalization	29.62	10.19
VGG-13 with batch normalization	28.45	9.63
VGG-16 with batch normalization	26.63	8.50
VGG-19 with batch normalization	25.76	8.15

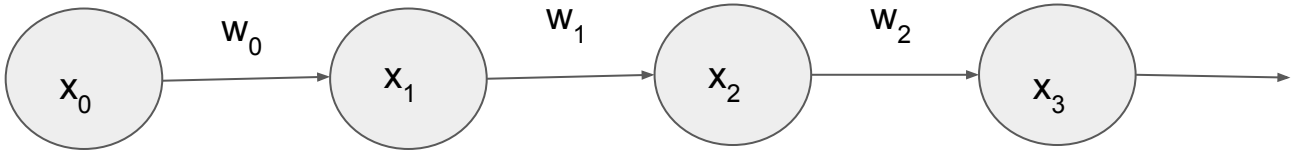
https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py

Затухание градиентов



$$w_0 = w_0 - \alpha \frac{\delta L}{\delta w_0} \qquad w_1 = w_1 - \alpha \frac{\delta L}{\delta w_1} \qquad w_2 = w_2 - \alpha \frac{\delta L}{\delta w_2}$$

Затухание градиентов



$$w_0 = w_0 - \alpha \frac{\delta L}{\delta w_0} \qquad w_1 = w_1 - \alpha \frac{\delta L}{\delta w_1} \qquad w_2 = w_2 - \alpha \frac{\delta L}{\delta w_2}$$

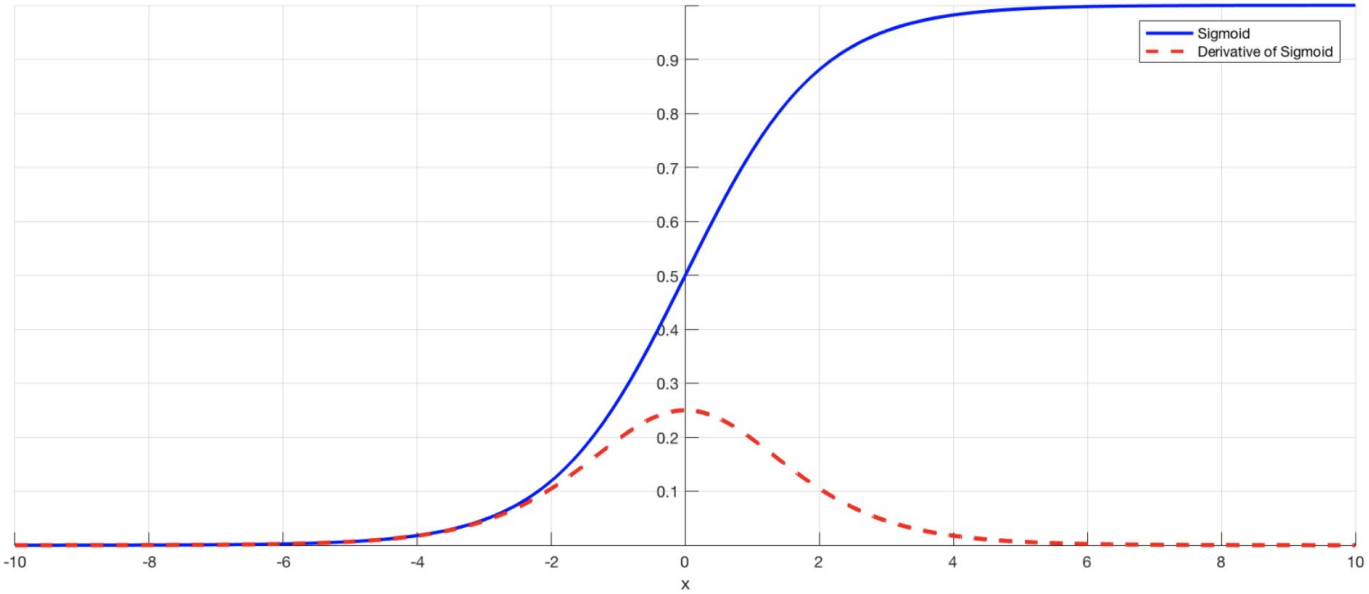
$$\frac{\delta L}{\delta w_0} = \frac{\delta L}{\delta x_3}\frac{\delta x_3}{\delta x_2}\frac{\delta x_2}{\delta x_1}\frac{\delta x_1}{\delta w_0} \qquad \frac{\delta L}{\delta w_1} = \frac{\delta L}{\delta x_3}\frac{\delta x_3}{\delta x_2}\frac{\delta x_2}{\delta w_1} \qquad \frac{\delta L}{\delta w_2} = \frac{\delta L}{\delta x_3}\frac{\delta x_3}{\delta w_2}$$

4 множителя < 1

3 множителя < 1

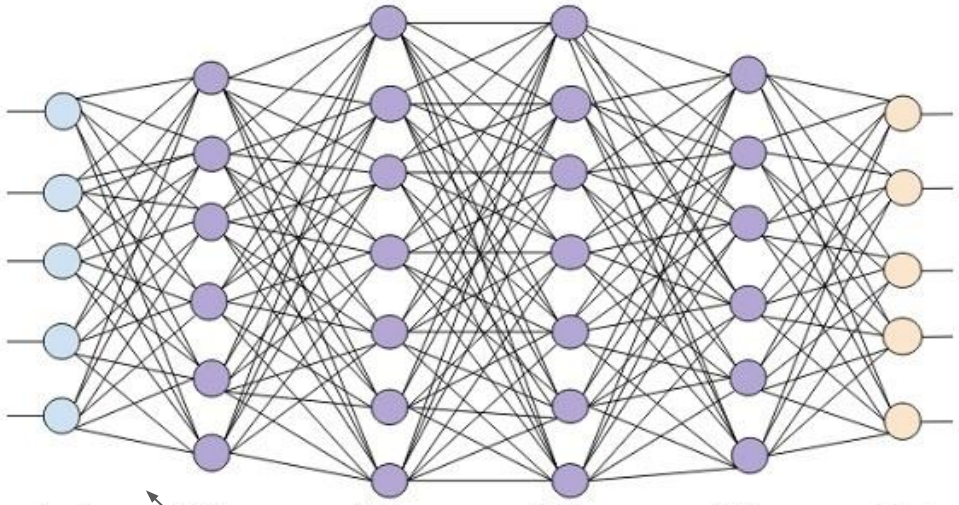
2 множителя < 1

Затухание градиентов

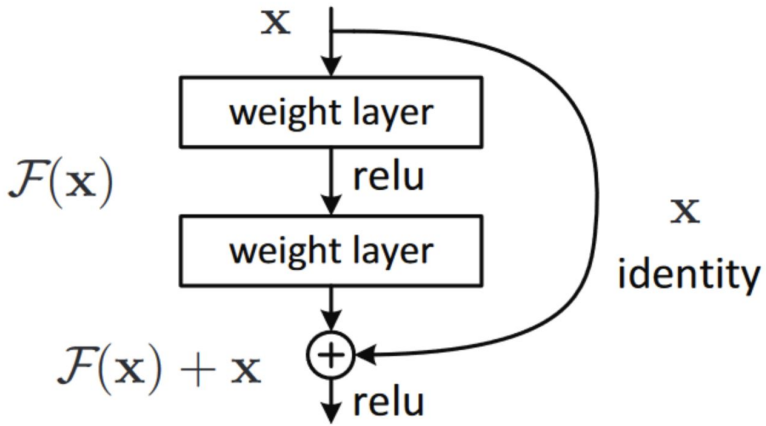


https://isaacchanghau.github.io/img/deeplearning/activationfunction/sigmoid.png

Затухание градиентов

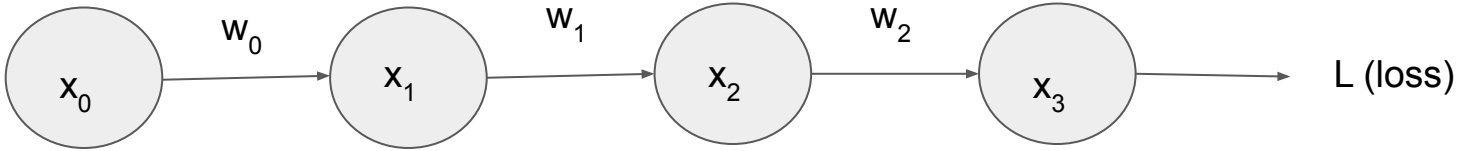


Skip Connection



Skip Connection

$$x_3 = x_2 w_2$$

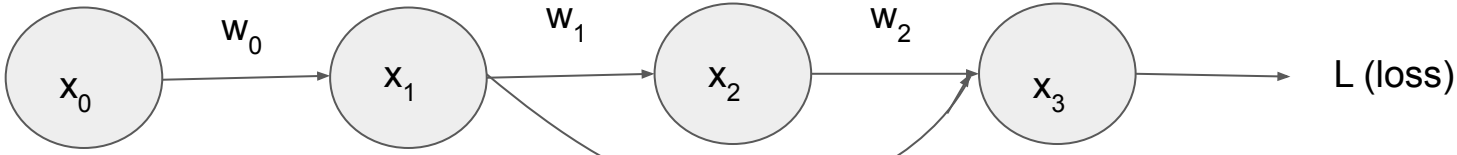


$$w_0 = w_0 - \alpha \frac{\delta L}{\delta w_0}$$

$$\frac{\delta L}{\delta w_0} = \frac{\delta L}{\delta x_3} \frac{\delta x_3}{\delta x_2} \frac{\delta x_2}{\delta x_1} \frac{\delta x_1}{\delta w_0}$$

Skip Connection

$$x_3 = x_2 w_2 + x_1$$

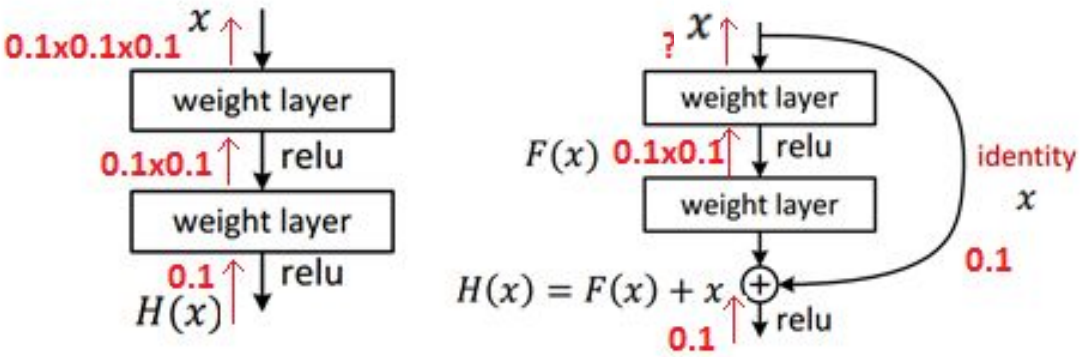


$$w_0 = w_0 - \alpha \frac{\delta L}{\delta w_0}$$

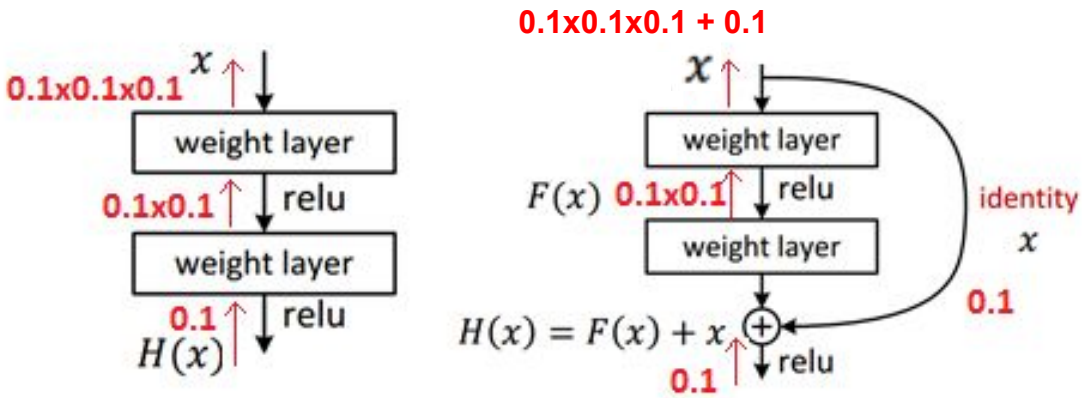
$$\frac{\delta L}{\delta w_0} = \frac{\delta L}{\delta x_3}\left(\left(\frac{\delta x_3}{\delta x_2}\frac{\delta x_2}{\delta x_1}\frac{\delta x_1}{\delta w_0}\right) + \left(\frac{\delta x_3}{\delta x_1}\frac{\delta x_1}{\delta w_0}\right)\right) =$$

$$= \frac{\delta L}{\delta x_3}\frac{\delta x_1}{\delta w_0}\left(\frac{\delta x_3}{\delta x_2}\frac{\delta x_2}{\delta x_1} + \frac{\delta x_3}{\delta x_1}\right) = \frac{\delta L}{\delta x_3}\frac{\delta x_1}{\delta w_0}\left(\frac{\delta x_3}{\delta x_2}\frac{\delta x_2}{\delta x_1} + 1\right)$$

Skip connection

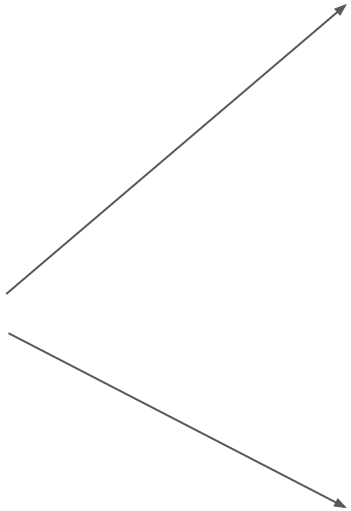


Skip connection



Skip Connection

код очень прост =)

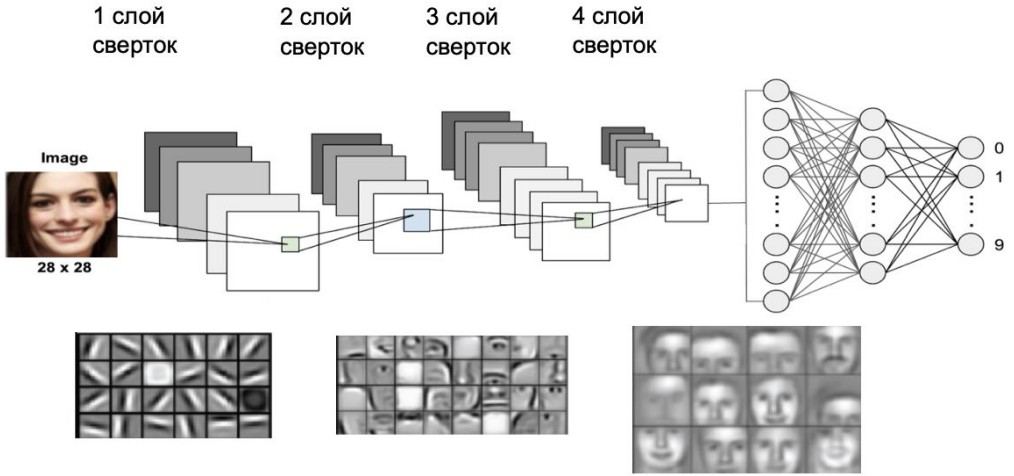


```
def forward(self, x: Tensor) -> Tensor:
    identity = x

    out = self.conv1(x)
    out = self.bn1(out)
    out = self.relu(out)

    out = self.conv2(out)
    out = self.bn2(out)

    if self.downsample is not None:
        identity = self.downsample(x)

    out += identity
    out = self.relu(out)

    return out
```

https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py

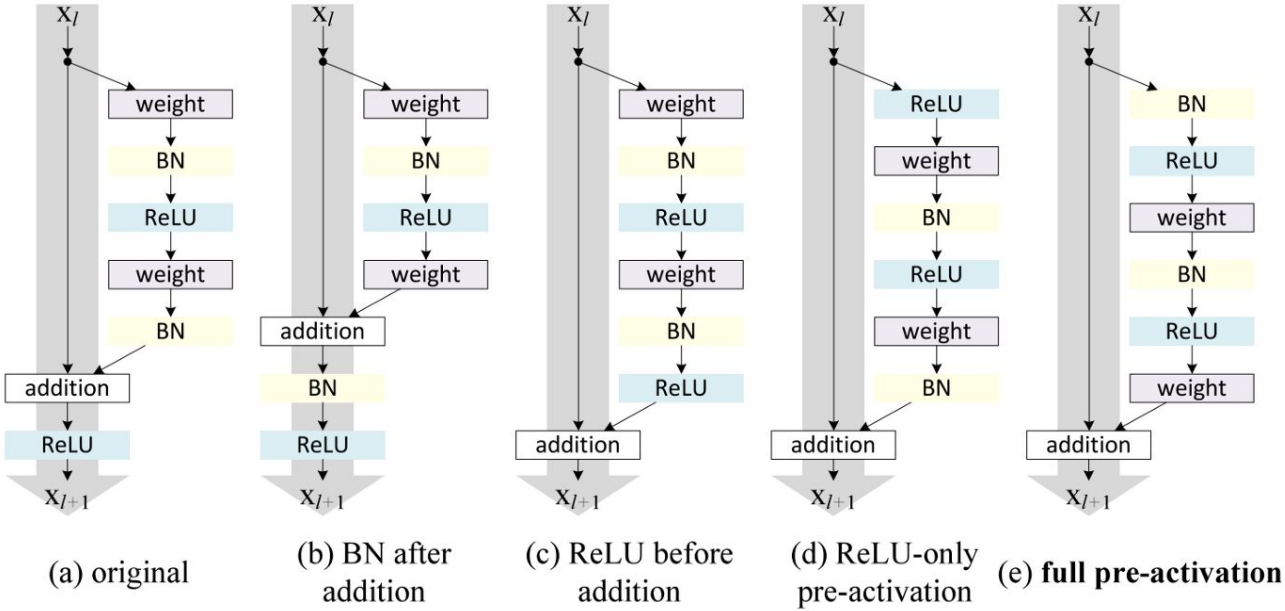
Skip connection

- используется чаще всего в CNN (включая ResNet)
- помогает в борьбе с затуханием градиентов
- помогает “протолкнуть” в нижние слои сети информацию из верхних слоев



ResNet: CNN + skip connection

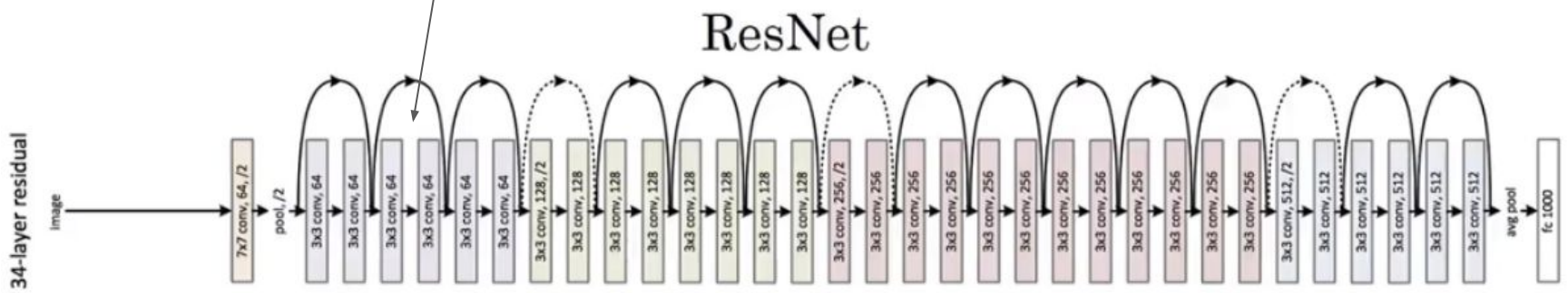
Варианты ResNet Residual blocks



https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035

ResNet

```
self.conv1 = conv3x3(inplanes, planes, stride)
self.bn1 = norm_layer(planes)
self.relu = nn.ReLU(inplace=True)
self.conv2 = conv3x3(planes, planes)
self.bn2 = norm_layer(planes)
self.downsample = downsample
self.stride = stride
```

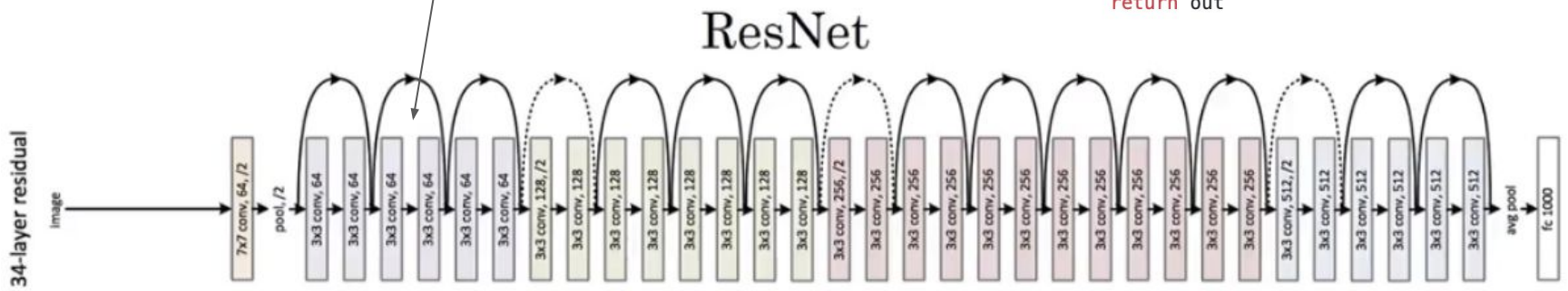


https://github.com/pytorch/vision/blob/mastera/torchvision/models/resnet.py

ResNet

```
self.conv1 = conv3x3(inplanes, planes, stride)
self.bn1 = norm_layer(planes)
self.relu = nn.ReLU(inplace=True)
self.conv2 = conv3x3(planes, planes)
self.bn2 = norm_layer(planes)
self.downsample = downsample
self.stride = stride
```

```
def forward(self, x):
    identity = x

    out = self.conv1(x)
    out = self.bn1(out)
    out = self.relu(out)

    out = self.conv2(out)
    out = self.bn2(out)

    if self.downsample is not None:
        identity = self.downsample(x)

    out += identity
    out = self.relu(out)

    return out
```



https://github.com/pytorch/vision/blob/mastera/torchvision/models/resnet.py

ResNet

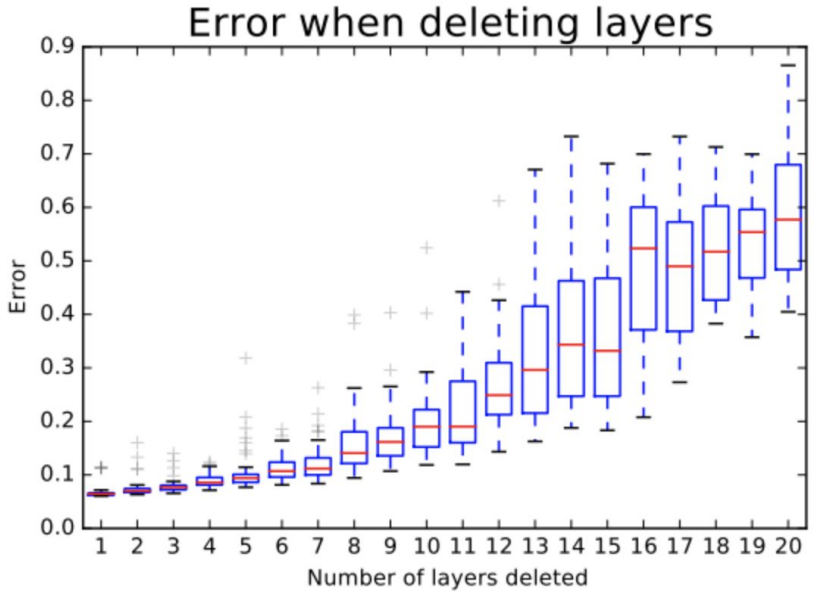
- ResNet-18
- ResNet-34
- ResNet-49
- ResNet-50
- ResNet-74
- ResNet-90
- ResNet-100
- ResNet-101
- ResNet-152

Network	Top-1 error	Top-5 error
ResNet-18	30.24	10.92
ResNet-34	26.70	8.58
ResNet-50	23.85	7.13
ResNet-101	22.63	6.44
ResNet-152	21.69	5.94

ResNet

Интересный факт: из ResNet можно выкинуть несколько слоев и ее performance снизится не слишком сильно.

Это достигается тем, что у ResNet есть “обходные пути” для информации (skip-connection), и при прерывании одного из них информация все еще может пройти по другому



DenseNet

DenseBlock:

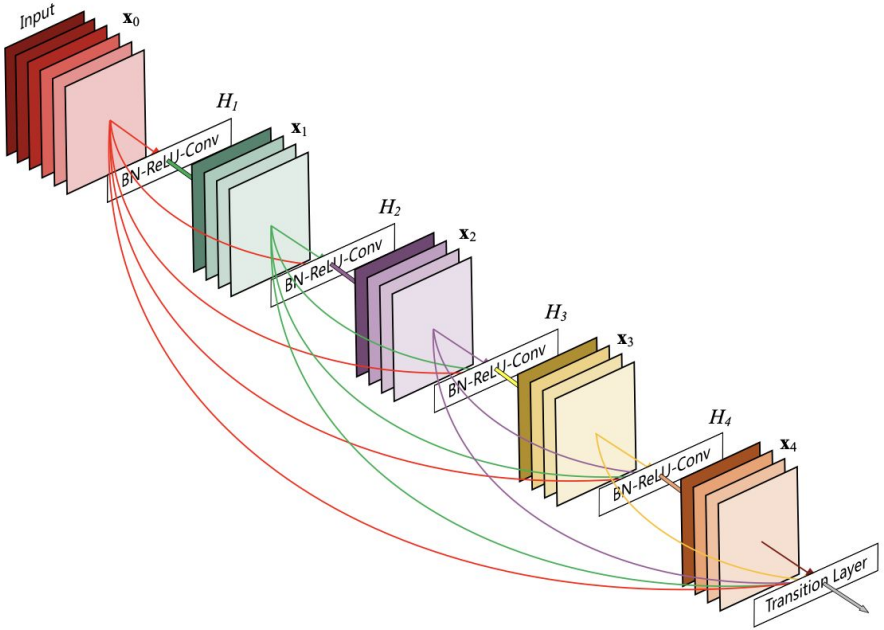
Каждый последующий слой сети получает на вход все выходы всех предыдущих сетей

https://arxiv.org/abs/1608.06993

DenseNet

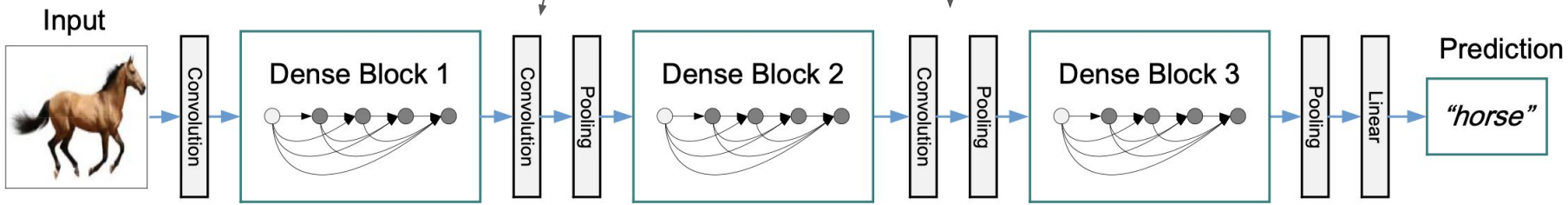
Dense Block c growth rate = 4

BN-relu-3x3 conv



DenseNet

Transition layers:
1x1 conv + pooling, чтобы добиться одинаковой размерности карт активаций на входе каждого dense block'а



(+relu, BN & 1x1 conv после каждой свертки внутри блока)

https://github.com/pytorch/vision/blob/master/torchvision/models/densenet.py

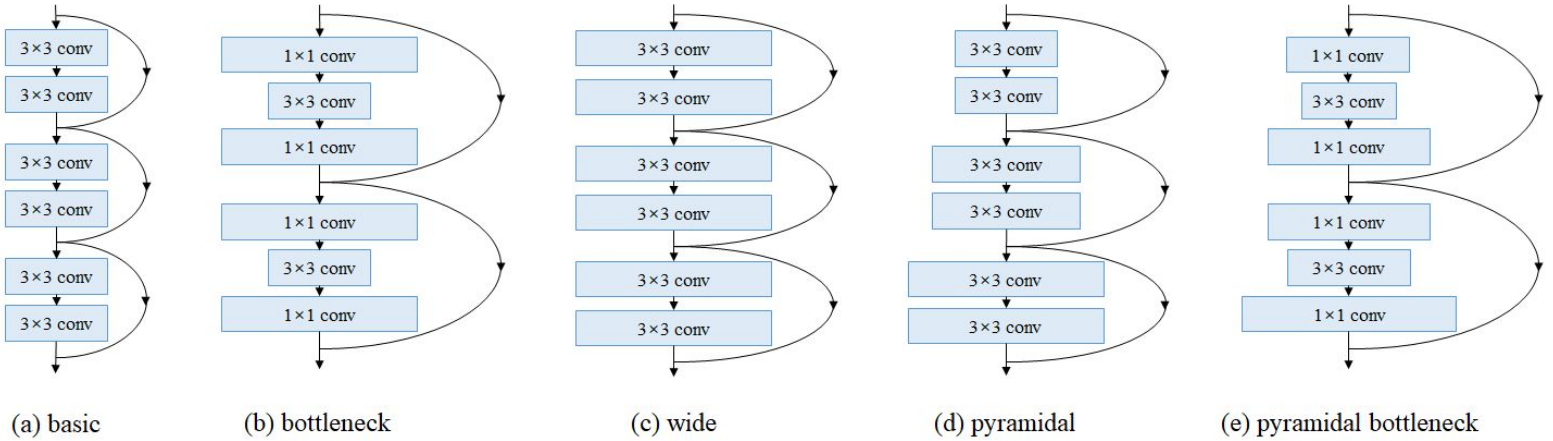
DenseNet Advantages

- strong gradient flow
- количество слоев и параметров не очень большое
- conv слои выделяют более разнообразные фичи
- нижние conv слои принимают во внимание low-complex паттерны из более верхних слоев, которые могут быть полезны для детекции некоих low-level паттернов.
  Также это помогает DenseNet лучше обучаться на маленьких датасетах.

https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803

Bottleneck Block

ResNet и DenseNet состоят из блоков.
Блоки, конечно, могут иметь разную архитектуру, не только ту, что мы рассматривали выше.



Часто используемые стандартные архитектуры сетей для классификации

TORCHVISION.MODELS

The models subpackage contains definitions of models for addressing different tasks, including: image classification, pixelwise semantic segmentation, object detection, instance segmentation, person keypoint detection and video classification.

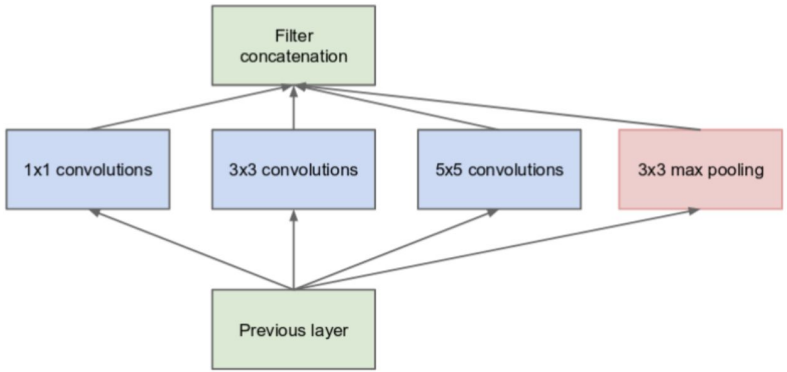
Classification

The models subpackage contains definitions for the following model architectures for image classification:

- AlexNet
- VGG
- ResNet
- SqueezeNet
- DenseNet
- Inception v3
- GoogLeNet
- ShuffleNet v2
- MobileNet v2
- ResNeXt
- Wide ResNet
- MNASNet

Inception

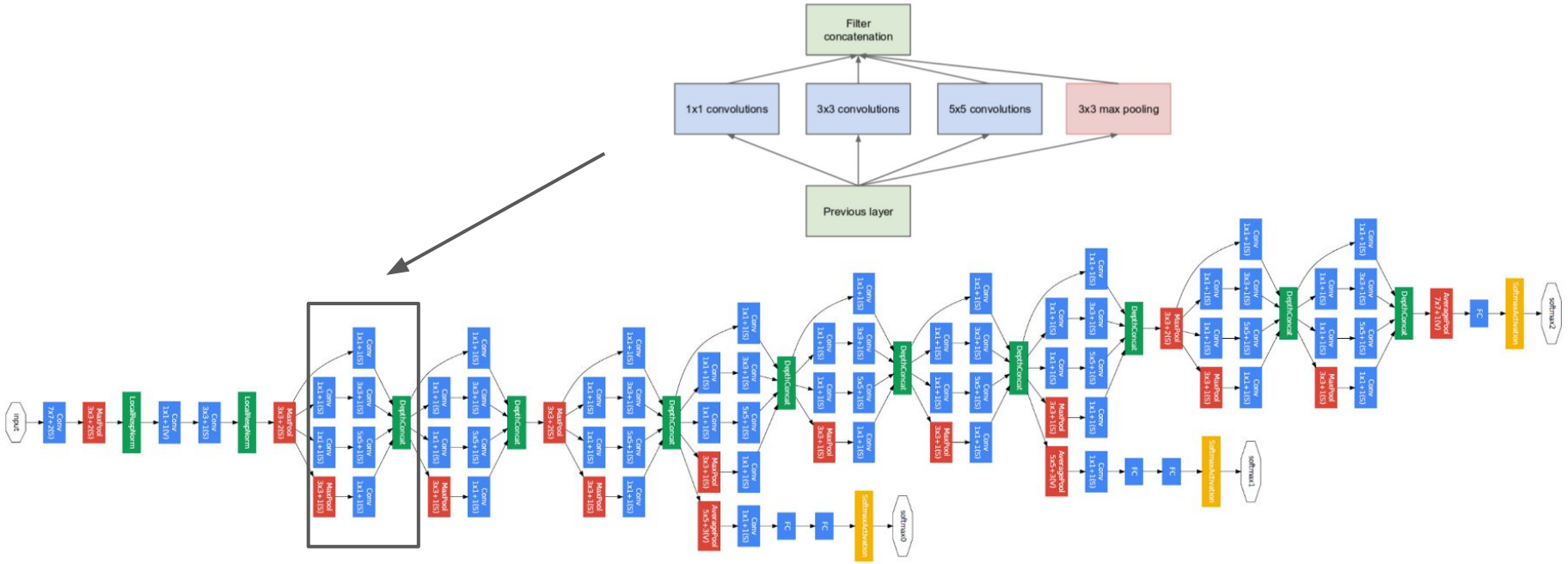
Идея: сделать несколько сверточные слоев с разными размерами фильтров на одном уровне



(a) Inception module, naïve version

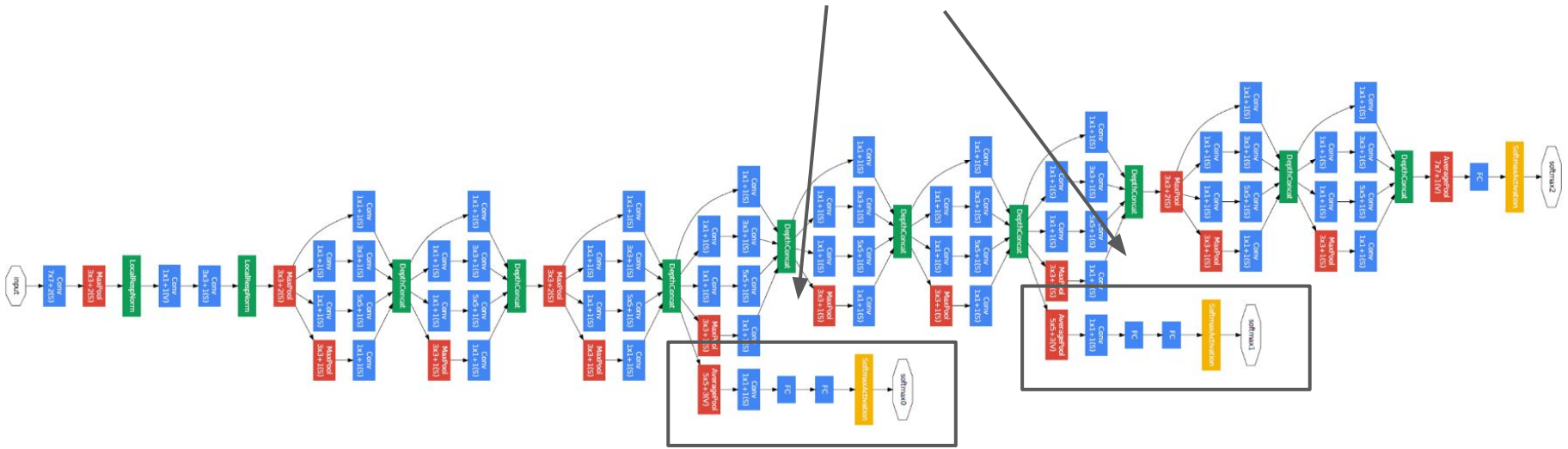
https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202

GoogleNet (Inception)



GoogleNet (Inception)

Дополнительные классификаторы для борьбы с затуханием градиентов



Transfer Learning

Transfer Learning

Часто датасет под какую-либо задачу для обучения сети содержит мало объектов.
И если обучать сеть на этом датасете с нуля, сеть переобучится.

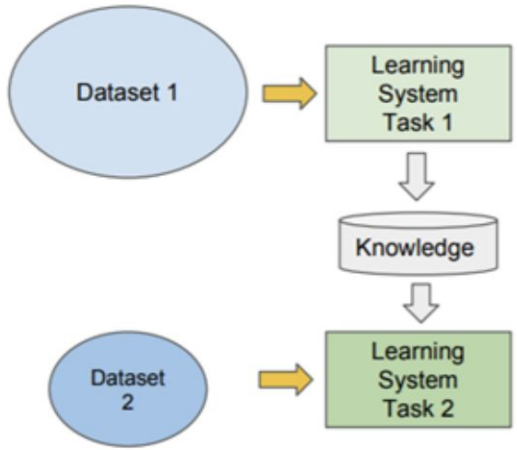
Пример: медицинские датасеты изображений опухолей, машинный перевод с малораспространенных языков (татарский), etc

Transfer Learning

Часто датасет под какую-либо задачу для обучения сети содержит мало объектов.
И если обучать сеть на этом датасете с нуля, сеть переобучится.

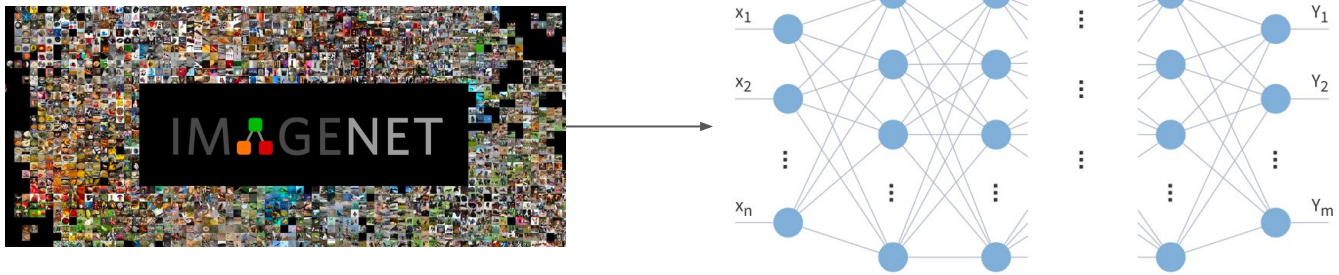
Пример: медицинские датасеты изображений опухолей, машинный перевод с малораспространенных языков (татарский), etc

Идея:
использовать знания, полученные другими сетями на похожих задачах.



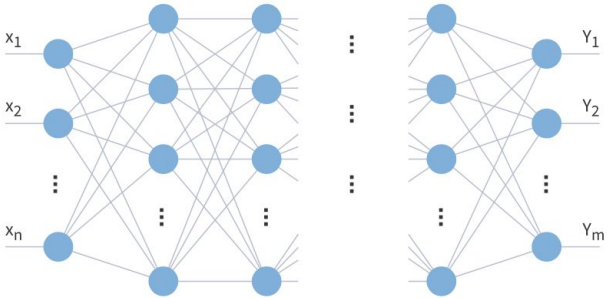
Дообучение сети (Fine-tuning)

Сеть ResNet, (пред)обученная на ImageNet

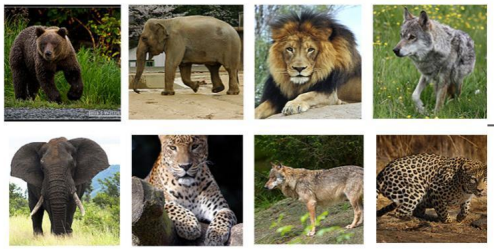


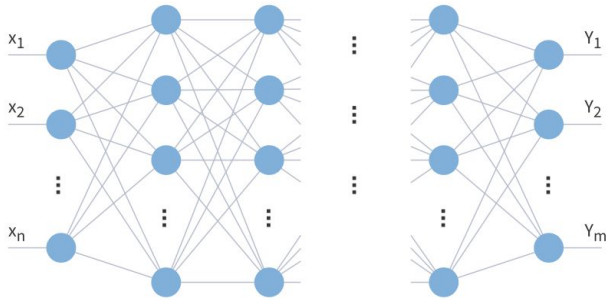
Обучаем на ImageNet

1.



2.

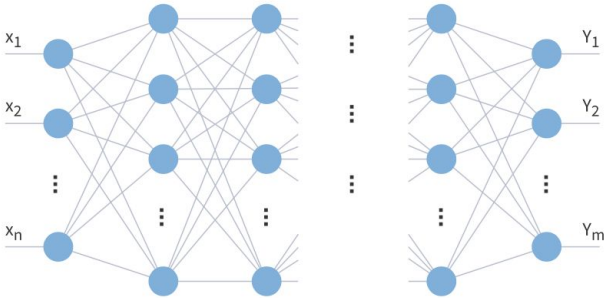




Дообучаем на нашем датасете диких животных

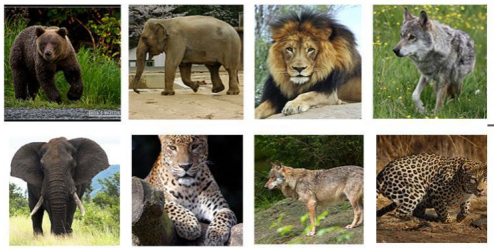
Обучаем на ImageNet

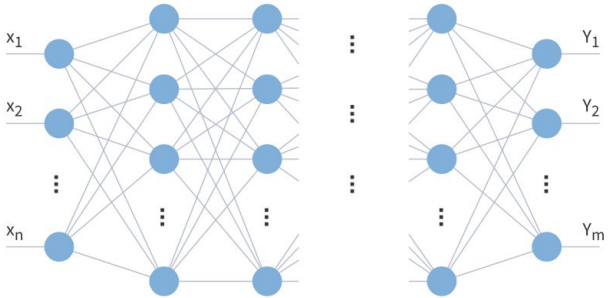
1.



Здесь должно быть
10 нейронов!

2.

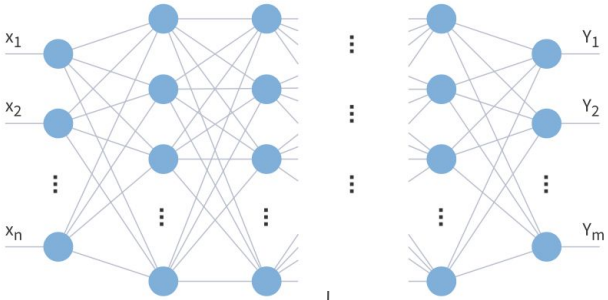




Дообучаем на нашем
датасете диких животных

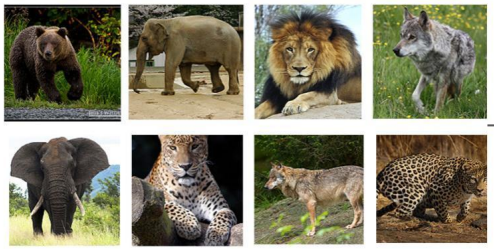
Обучаем на ImageNet

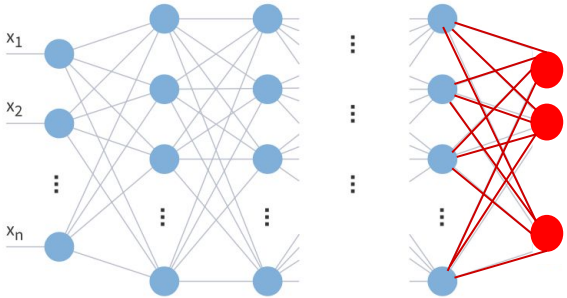
1.



Выкидываем последний слой и заменяем на новый из 10 нейронов

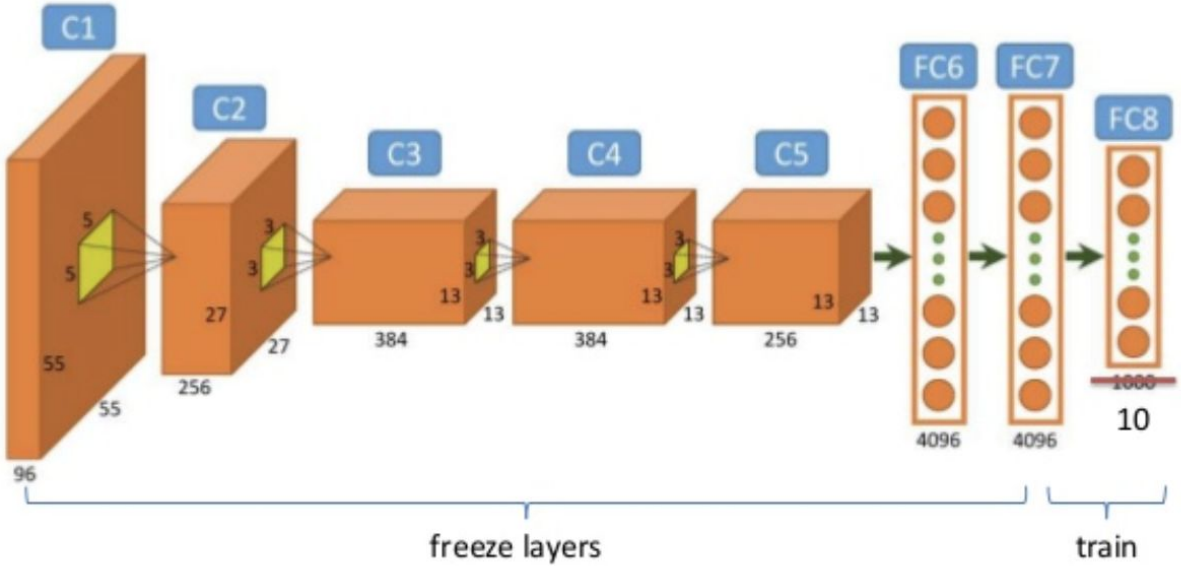
2.





Дообучаем на нашем датасете диких животных

Fine-tuning Pretrained Network

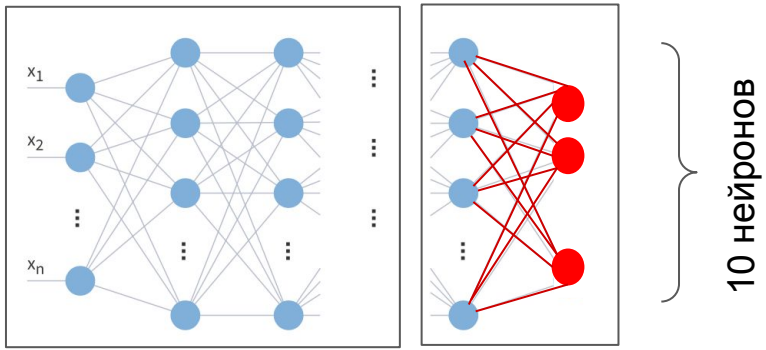


Заморозка слоев

Заморозим первые слои:
их веса не будут обновляться

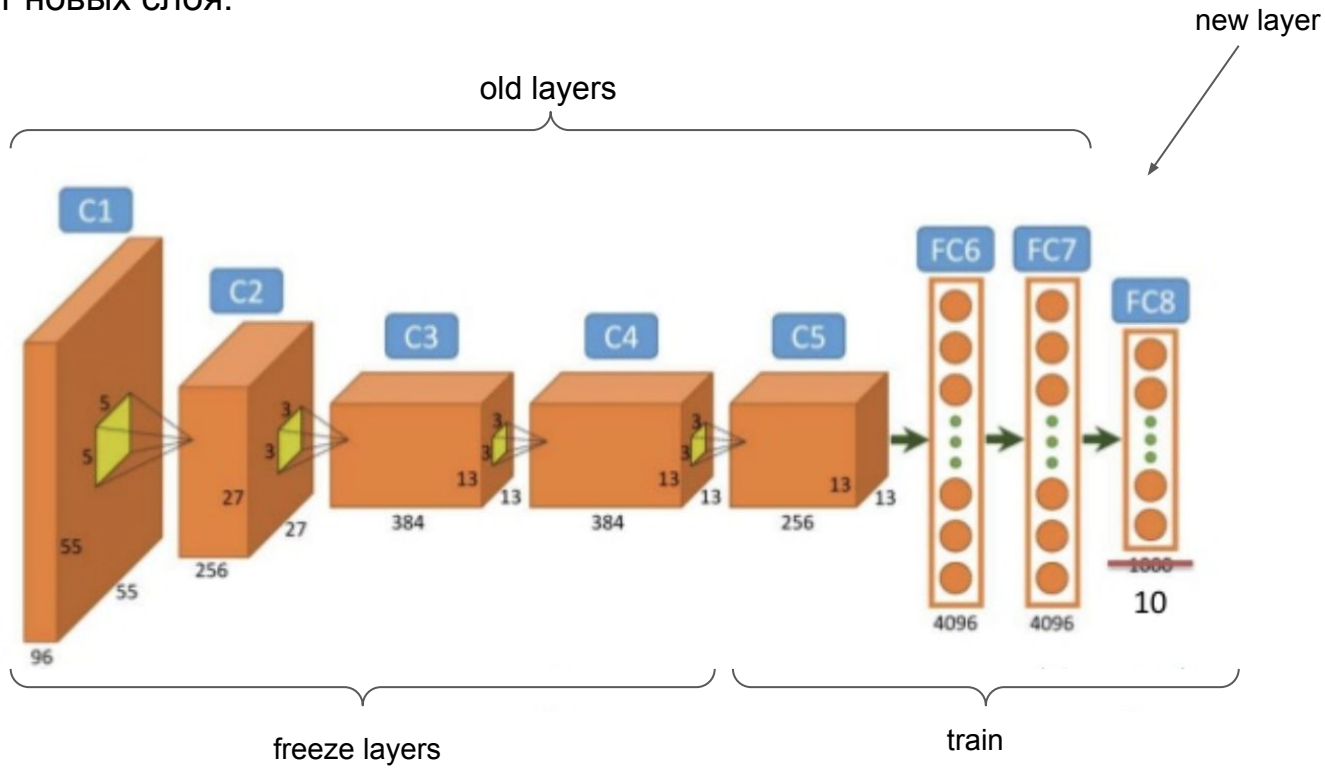
Эти веса обновляться будут

10 классов



10 нейронов

Мы можем выбирать, какие слои замораживать и какие слои заменять
Можно заморозить 10 первых слоев, тренировать 5 оставшихся, среди которых 2 будут новых слоя.



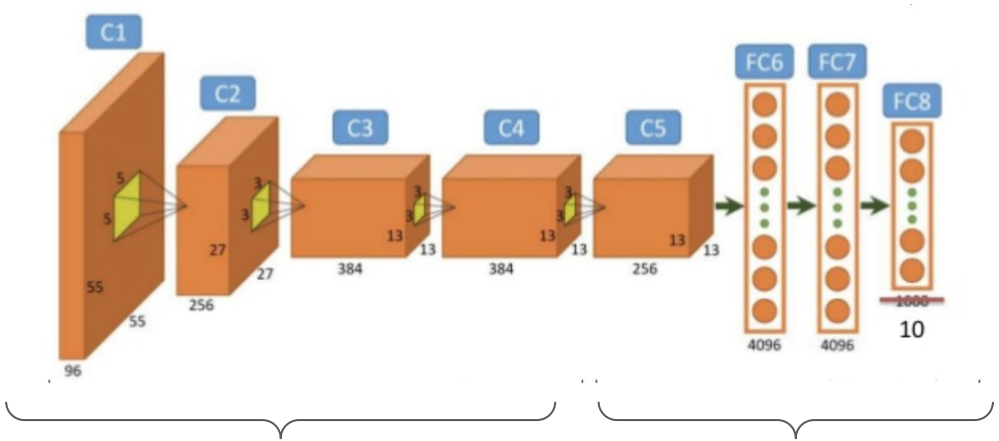
Fine-tuning

- Замораживаются верхние (первые) слои сети, нижние дообучаются. Верхние слои выделяют низкоуровневую информацию, и они научились хорошо это делать при предобучении. Нижние слои выделяют из информации, полученной из верхних слоев, спецефическую для задачи информацию, поэтому их нужно дообучать.

- Сколько слоев замораживать, заивисит от различия между датасетами, сложности задачи и объема датасета для дообучения.

Fine-tuning

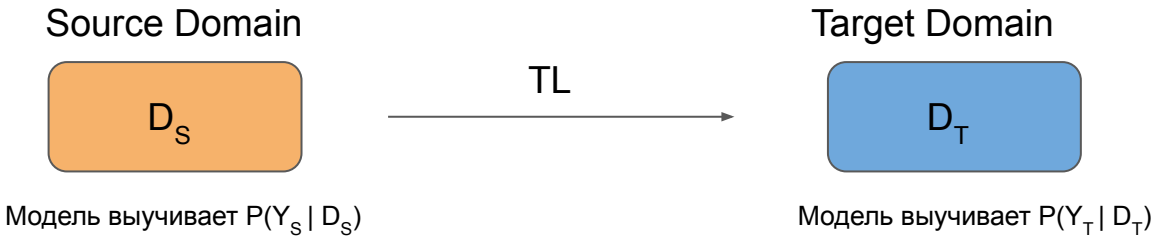
- Чем больше в target задаче тренировочных данных и чем сильнее source и target задачи отличаются между собой, тем больше слоев нужно дообучать.



Верхние слои содержат более общую инфрмацию, которая пригодится при дообучении на новой задаче

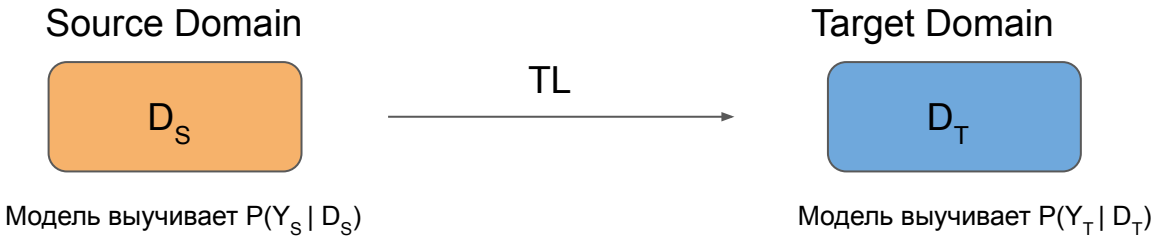
Нижние слои содержат специфическую для задачи информацию. При переходе на новую задачу нужно научить сеть выделять другую инфу, нужную для решения новой задачи

Transfer Learning



Понятия Transfer Learning, Domain Adaptation и Multi-task Learning часто используются в одном контексте

Transfer Learning

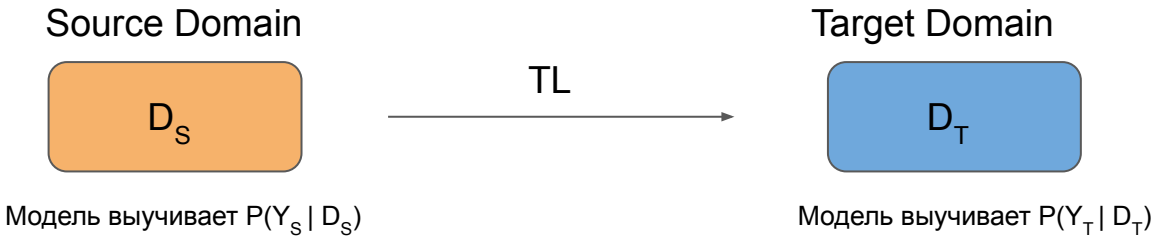


1. $D_S \neq D_T$

Пространства признаков двух доменов —разные.

Пример:
В NLP два датасета текстов на двух разных языках.

Transfer Learning

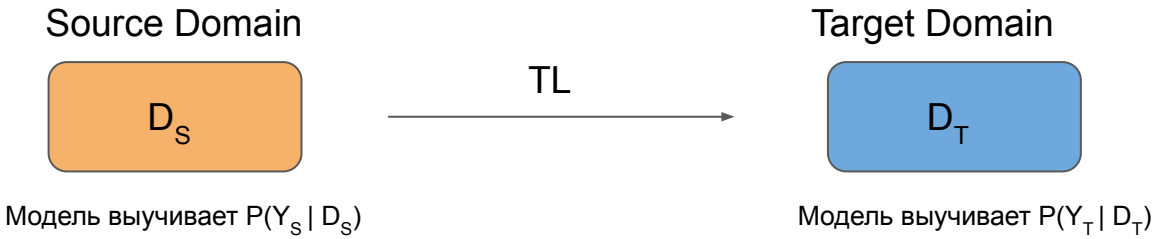


2. $P(D_S) \neq P(D_T)$

Распределения признаков в двух доменах разные. Такой TL называется Domain Adaptation.

Пример:
Датасет ревью на фильмы на Кинопоиске и датасет ревью на мобильные приложения в Google Play. И там, и там задача — предсказать оценку ревью (от 0 до 5)

Transfer Learning



3. $Y_S \neq Y_T : P(Y_S \mid D_S) \neq P(Y_T \mid D_T)$

Пример:
Датасет лиц людей.
Задача 1: классификация лиц по половому признаку,
Задача 2: классификация лиц по расовому признаку

Transfer Learning

Разные домены: задача одна и та же: классификация изображений цифр на 10 классов
Домены (распределения D_S и D_T) — разные

MNIST

SVHN

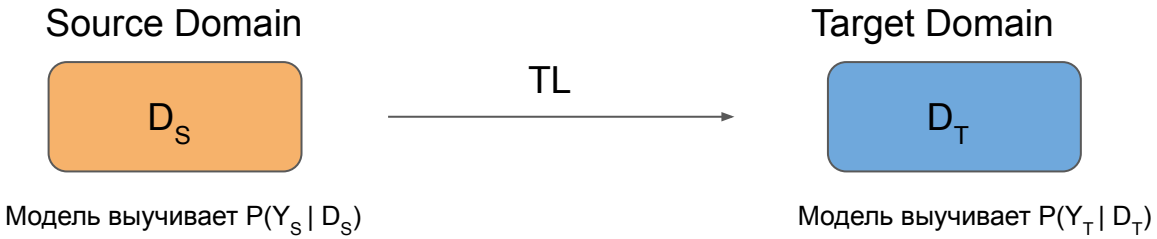
Transfer Learning

Разные задачи: есть различия в лейблах: либо они вообще разные, либо различны условные распределения на лейблы: $P(Y_S \mid D_S) \neq P(Y_T \mid D_T)$

Задача: определение расы

Задача: определение эмоции

Transfer Learning



4. Supervised / Unsupervised

Есть ли лейблы у D_T

Пример:

Датасет лиц людей.

D_S: размеченные лица по id

D_T: неразмеченные лица людей из другой части мира

Transfer Learning Ideas

Когда нейронная сеть обучается, она учится выделять из объектов inter-domain и intra-domain информацию.

inter-domain: расположение элементов на лице, выражения лиц

intra-domain: цвет лица, широта глаз

Transfer Learning Ideas

Когда нейронная сеть обучается, она учится выделять из объектов inter-domain и intra-domain информацию.

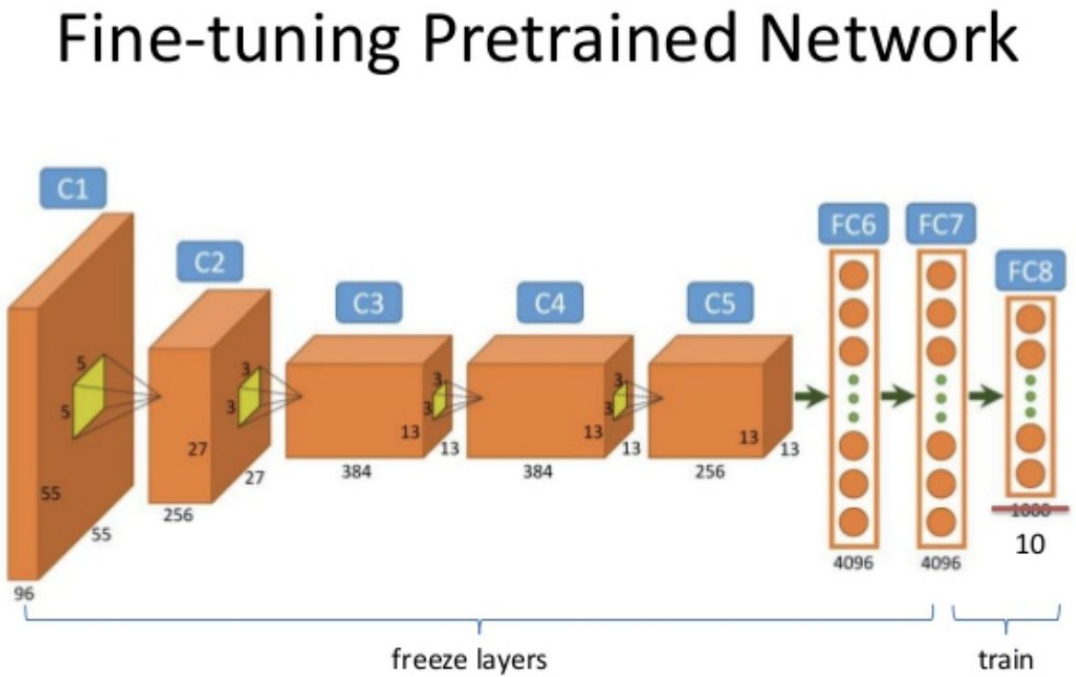
inter-domain: расположение элементов на лице, выражения лиц

intra-domain: цвет лица, широта глаз

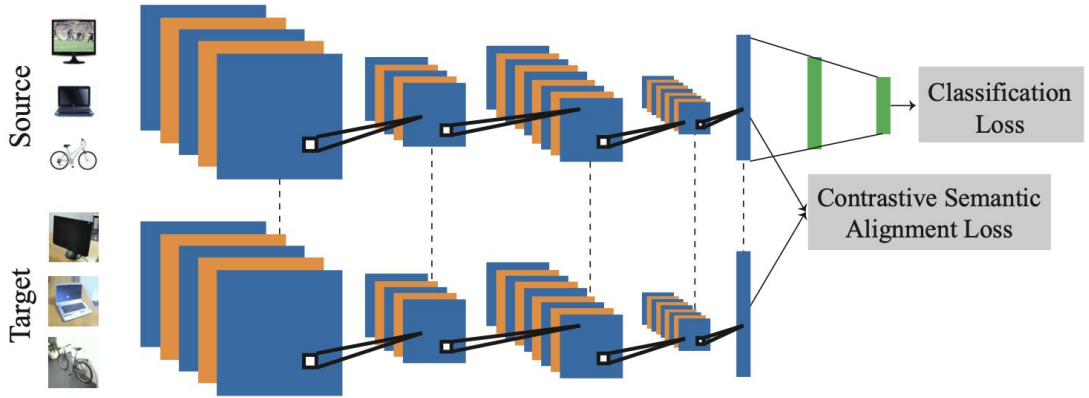
Как перенести только нужные знания сети (inter-domain) на новый домен?

https://arxiv.org/pdf/1802.03601.pdf

Idea #1 : Fine-tuning

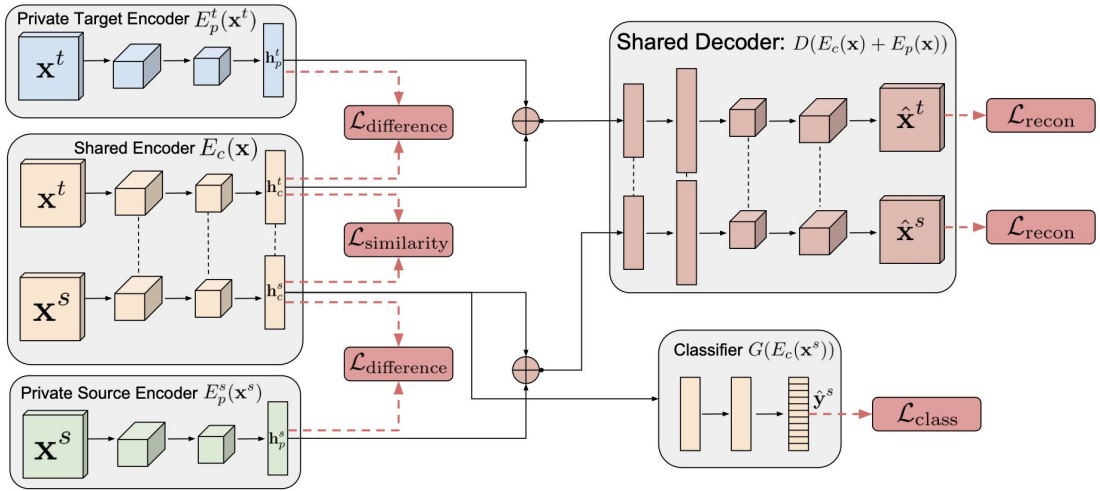


Idea #2: Learn two tasks simultaneously



https://arxiv.org/pdf/1709.10190.pdf

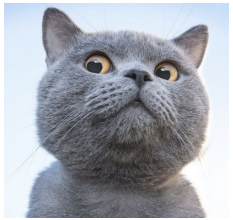
Idea #2: Learn two tasks simultaneously

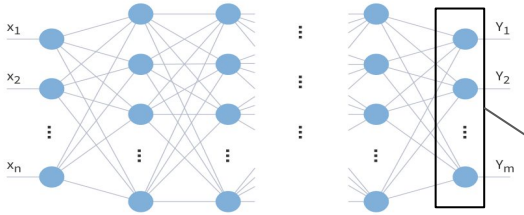


https://arxiv.org/pdf/1608.06019.pdf

Transfer Learning: extreme cases

Сеть ResNet, (пред)обученная на ImageNet

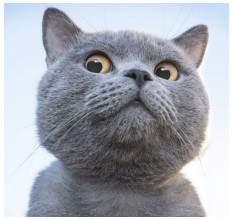


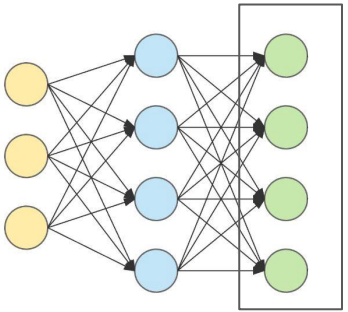


Сеть-teacher

нет доступа к ее коду, мы можем только получать эмбеддинги

Loss (e.g MSE), который заставляет оба эмбеддинга быть похожими

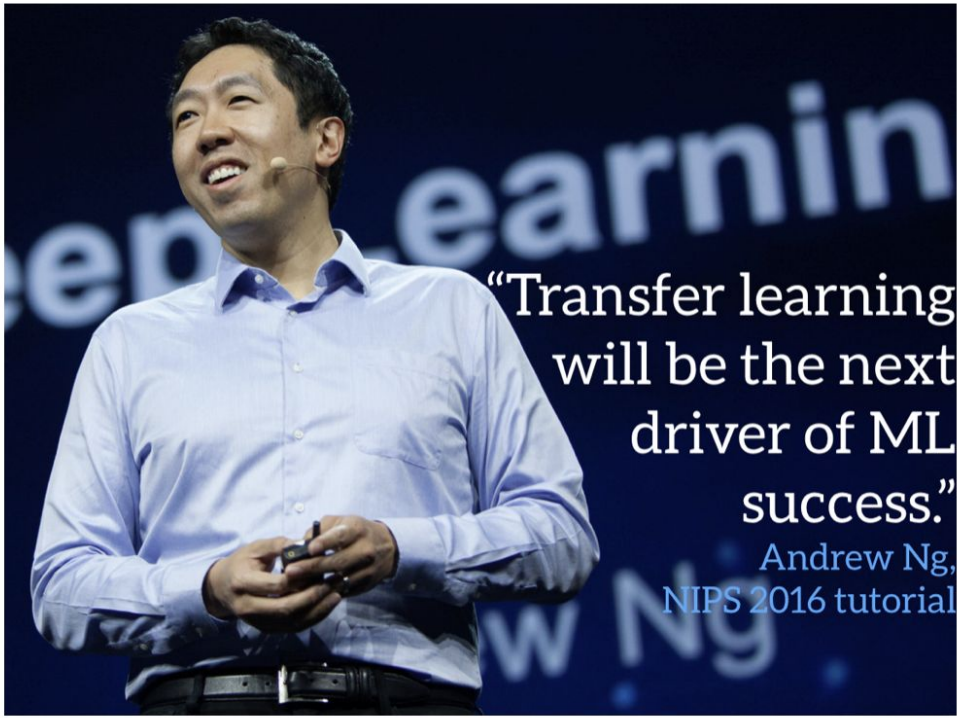




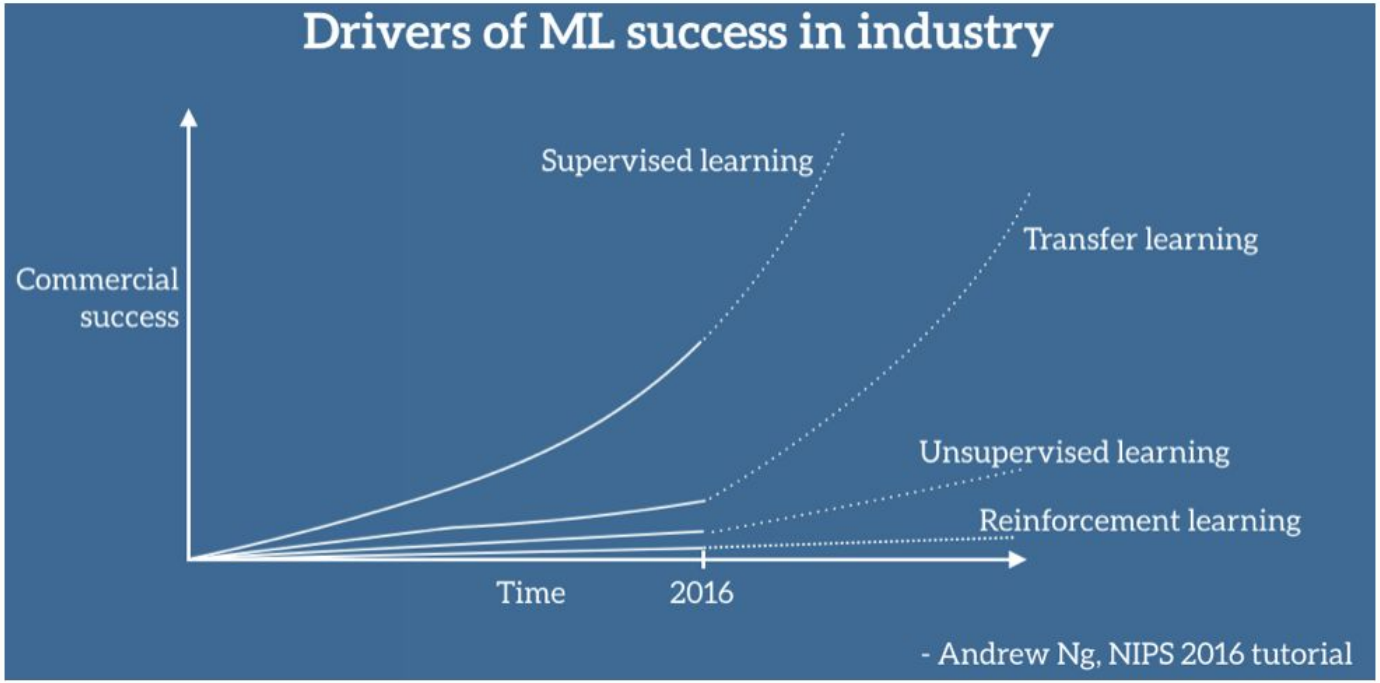
Сеть-student

Transfer Learning is a new black

“Transfer learning will be —
after supervised learning —
the next driver of ML commercial
success”



Transfer Learning is a new black



Transfer Learning is a new black

Почему Transfer Learning становится все более популярным?

- все больше задач в индустрии, которые хочется решать с помощью deep learning, но данных в принципе мало

- зато симулировать мы можем оочень много данных!
  но у симулированных данных $P(D_S) \neq P(D_T)$, поэтому это случай, когда нужен transfer learning

The End