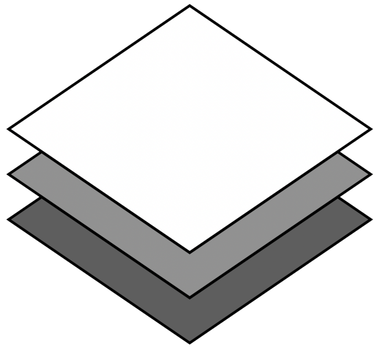


Lecture 2.B
Computer Vision Applications

Goal for this lecture

1. Review notable applications of DL in computer vision
2. Build a toolbox of model architectures and tricks

Agenda

1. Tour of Convnet Architectures
2. Localization, Detection, Segmentation
3. Misc

Agenda

1. **Tour of Convnet Architectures**
2. Localization, Detection, Segmentation
3. Misc

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

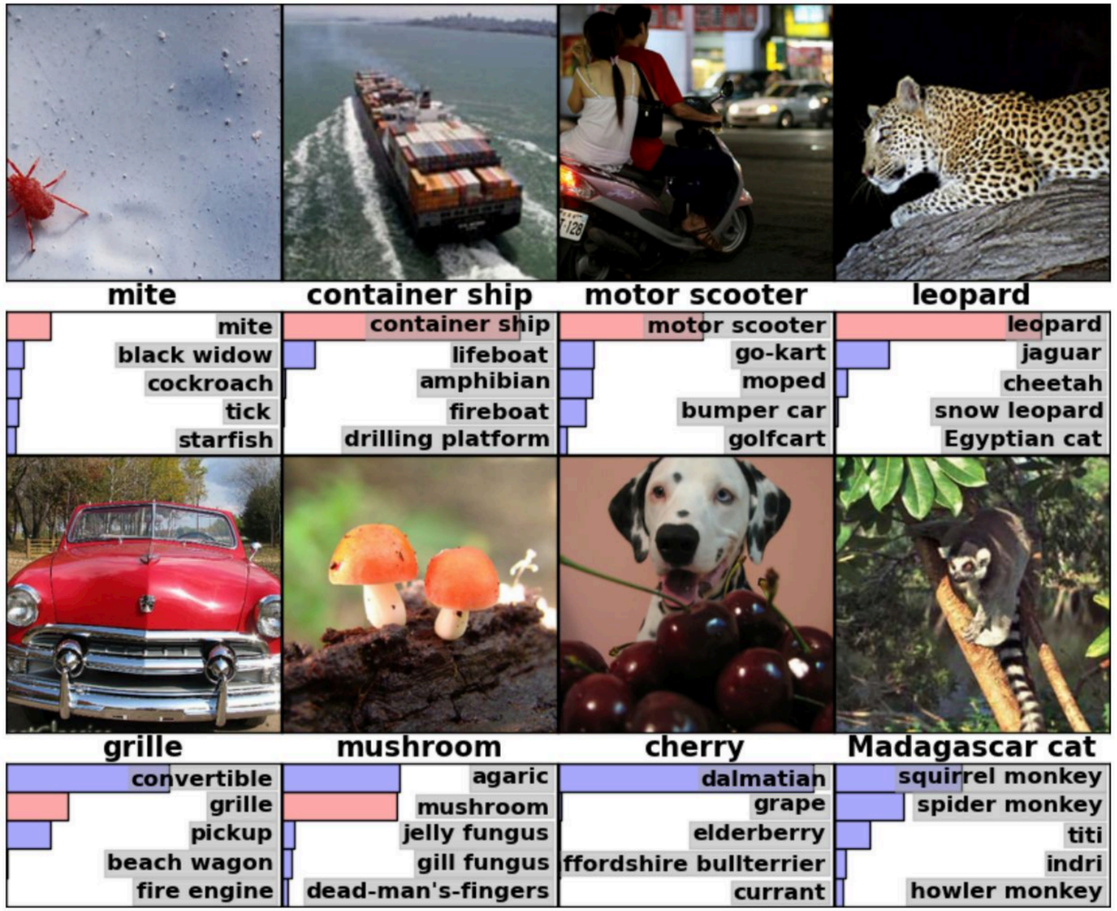
The models subpackage contains definitions for the following model architectures for image classification:

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

Progress in Convnet Architectures

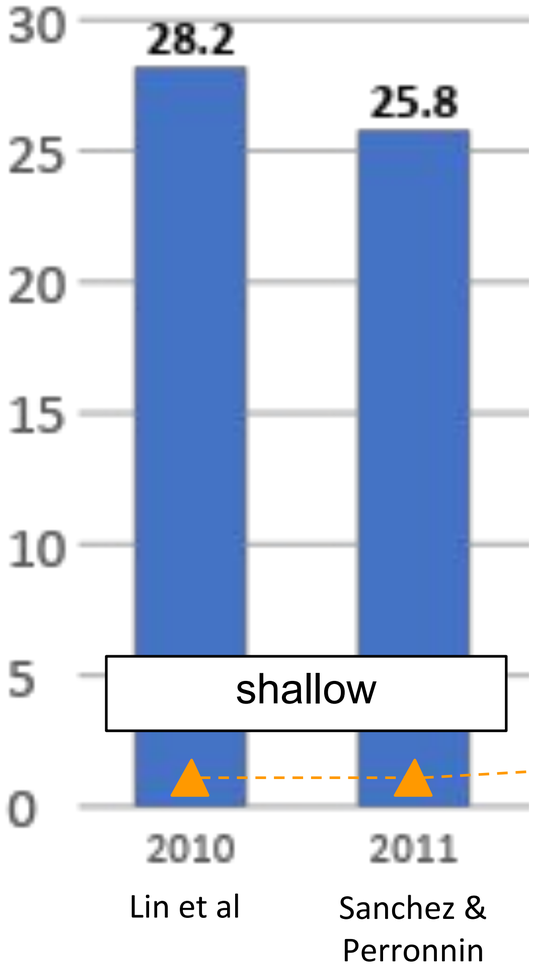
- AlexNet's outstanding performance on ImageNet in 2012 launched the convnet revolution
- Progress since then is well tracked by ImageNet winners

ImageNet



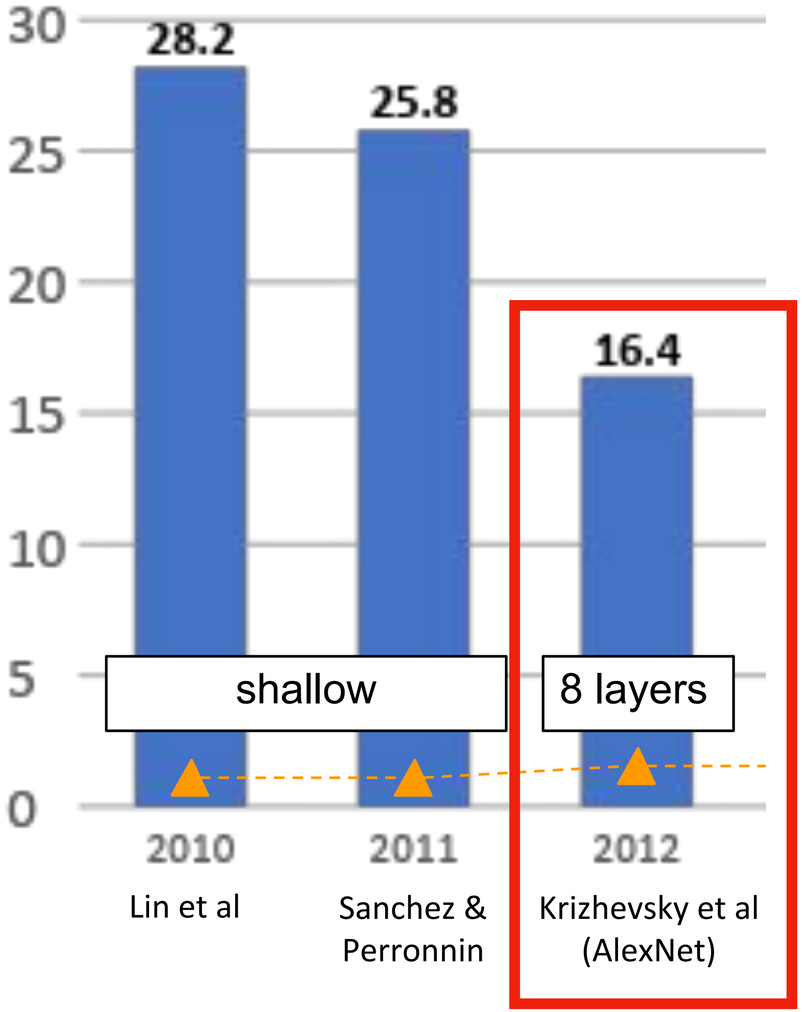
- ImageNet Large Scale Visual Recognition Challenge (ILSVRC): running since 2010
- 1.2M images from 1000 object categories (test on 150K)
- Classification, Localization, and Detection tracks. Classification has gotten the most attention.

ILSVRC Winners



http://cs231n.stanford.edu/

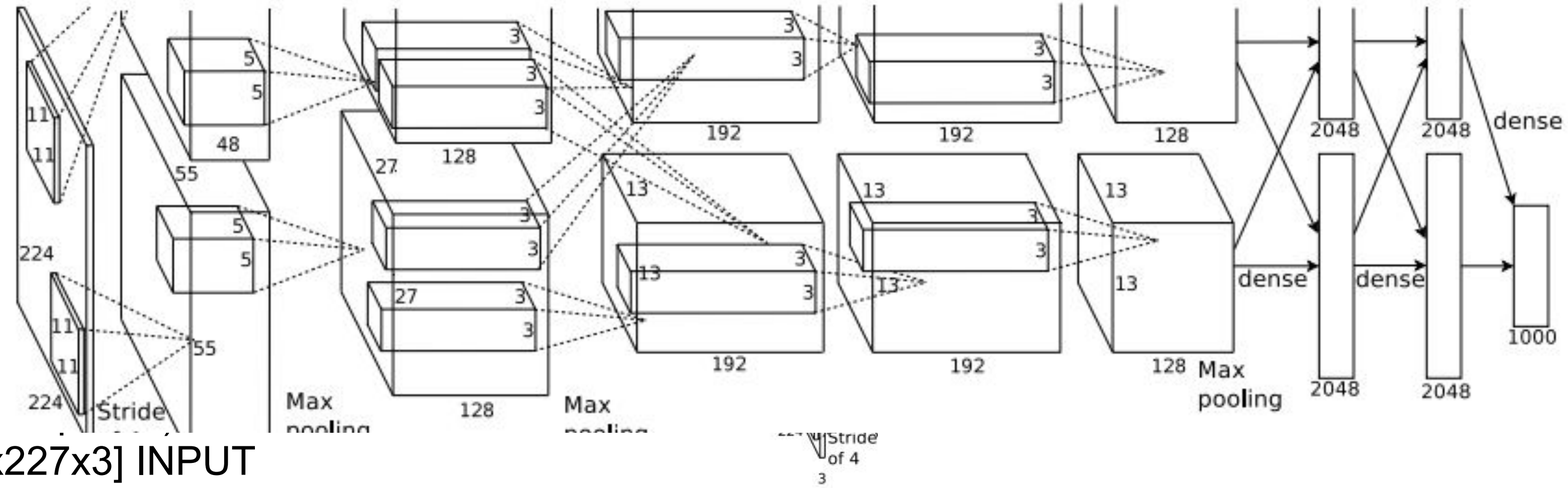
ILSVRC Winners



http://cs231n.stanford.edu/

[Krizhevsky et al. 2012]

AlexNet

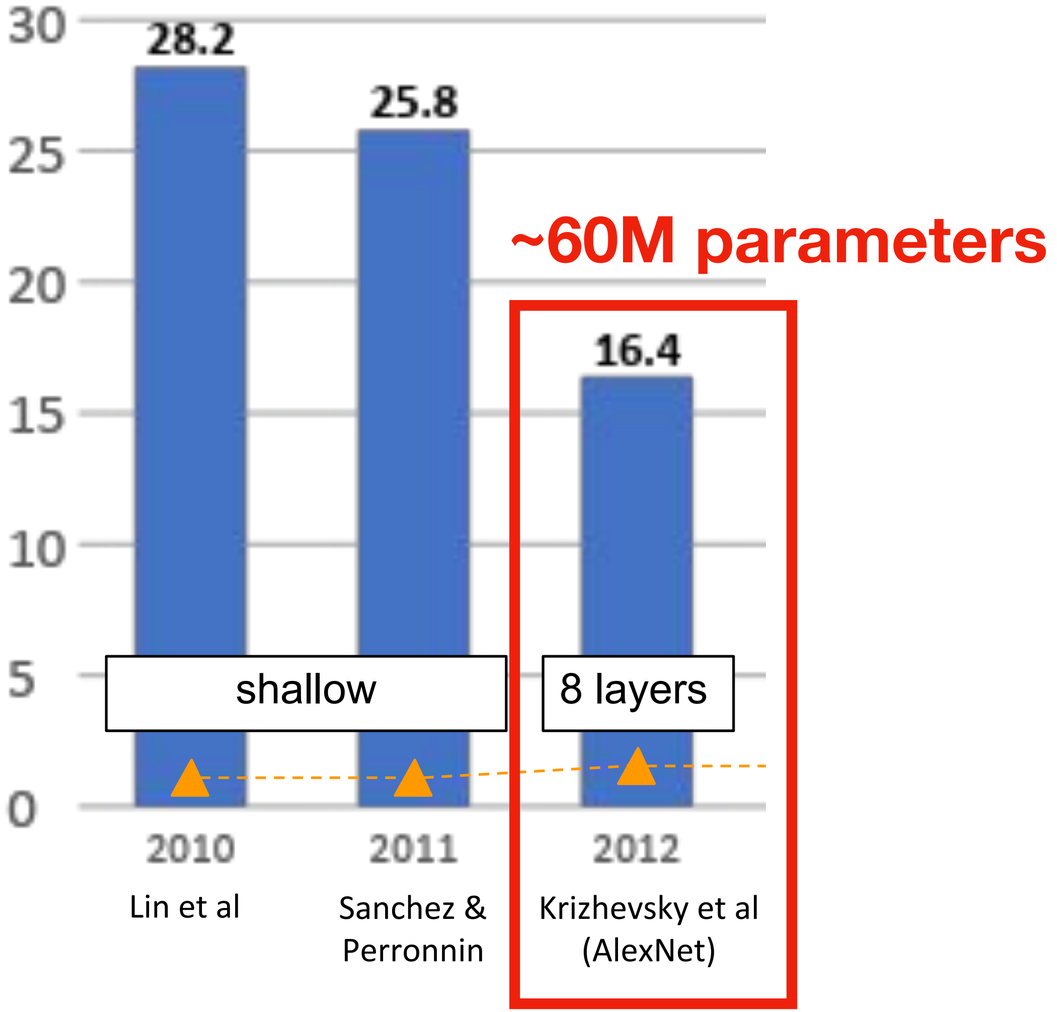


[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

- First convnet winner
- Innovated with ReLU and Dropout
- Heavy data augmentation (flip, scale, etc)
- Fun fact: usually drawn as two parts, because GPU only had 3GB memory

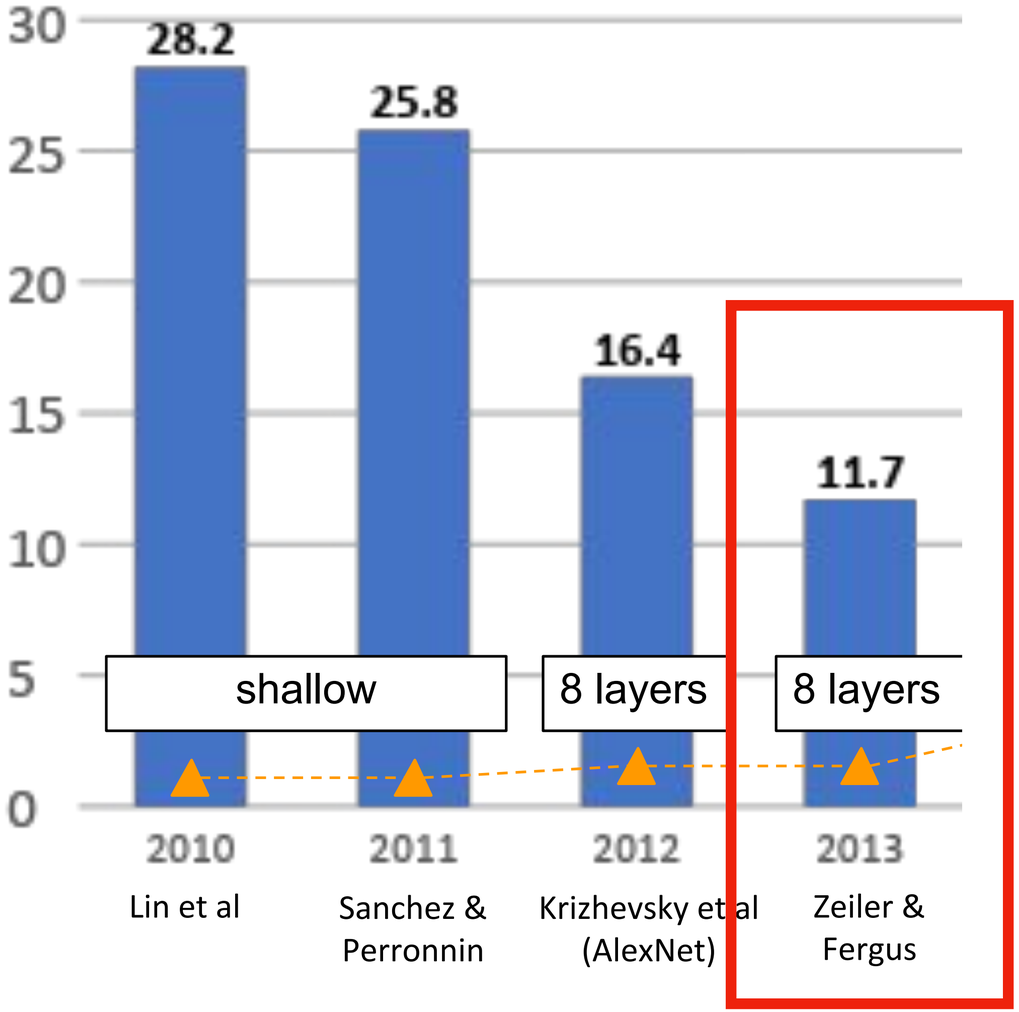
http://cs231n.stanford.edu/

ILSVRC Winners



http://cs231n.stanford.edu/

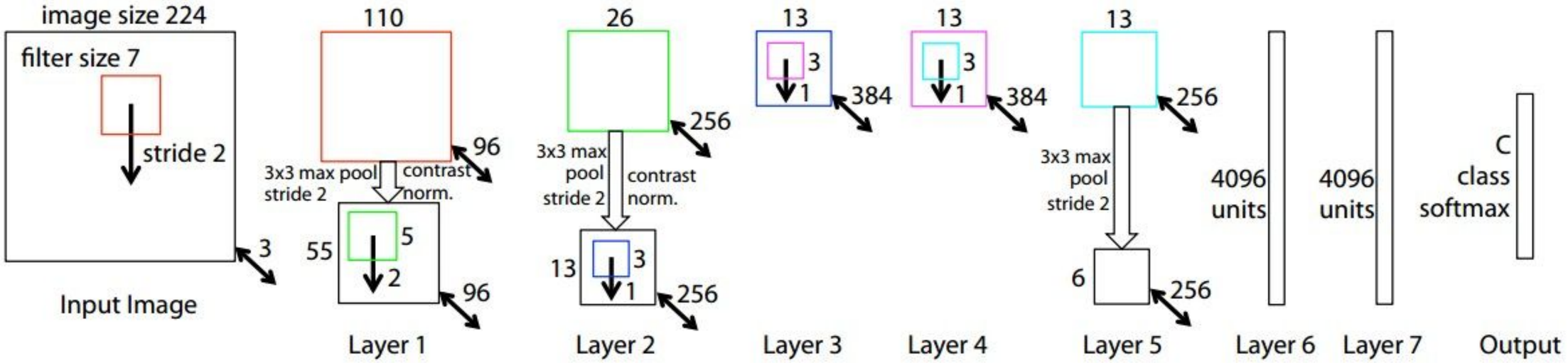
ILSVRC Winners



http://cs231n.stanford.edu/

ZFNet

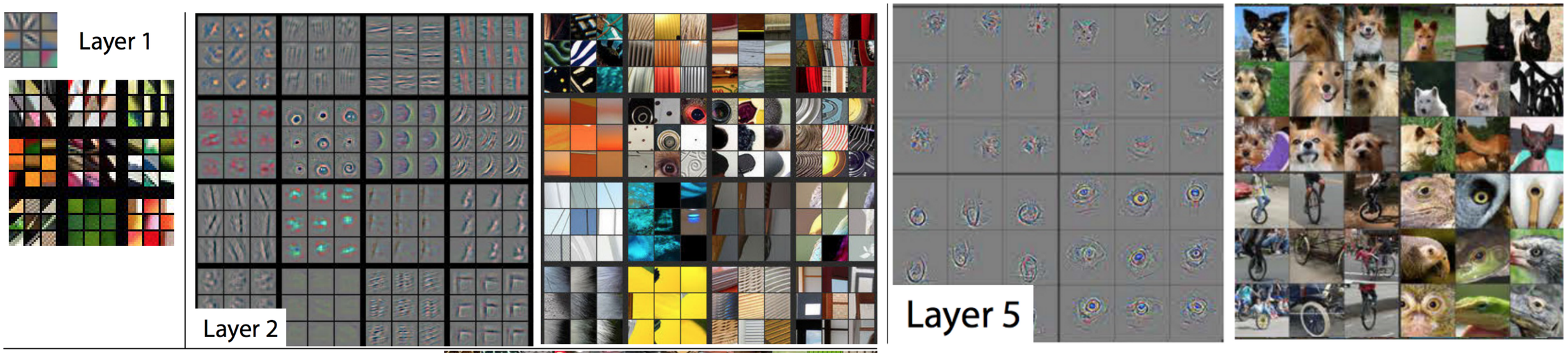
[Zeiler and Fergus, 2013]



- Essentially AlexNet, but with a smaller CONV1 and higher-channel CONV3,4,5
- Reduced error 5%!

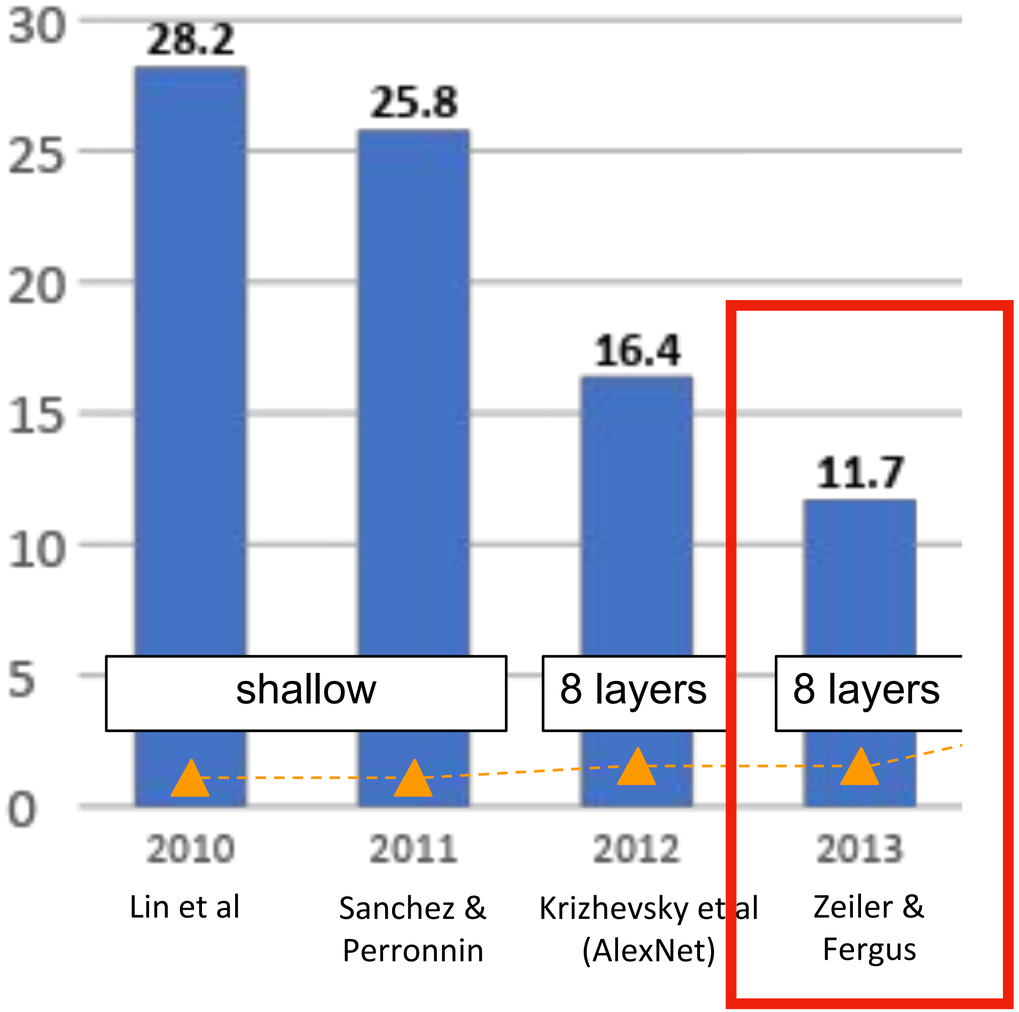
ZFNet

[Zeiler and Fergus, 2013]



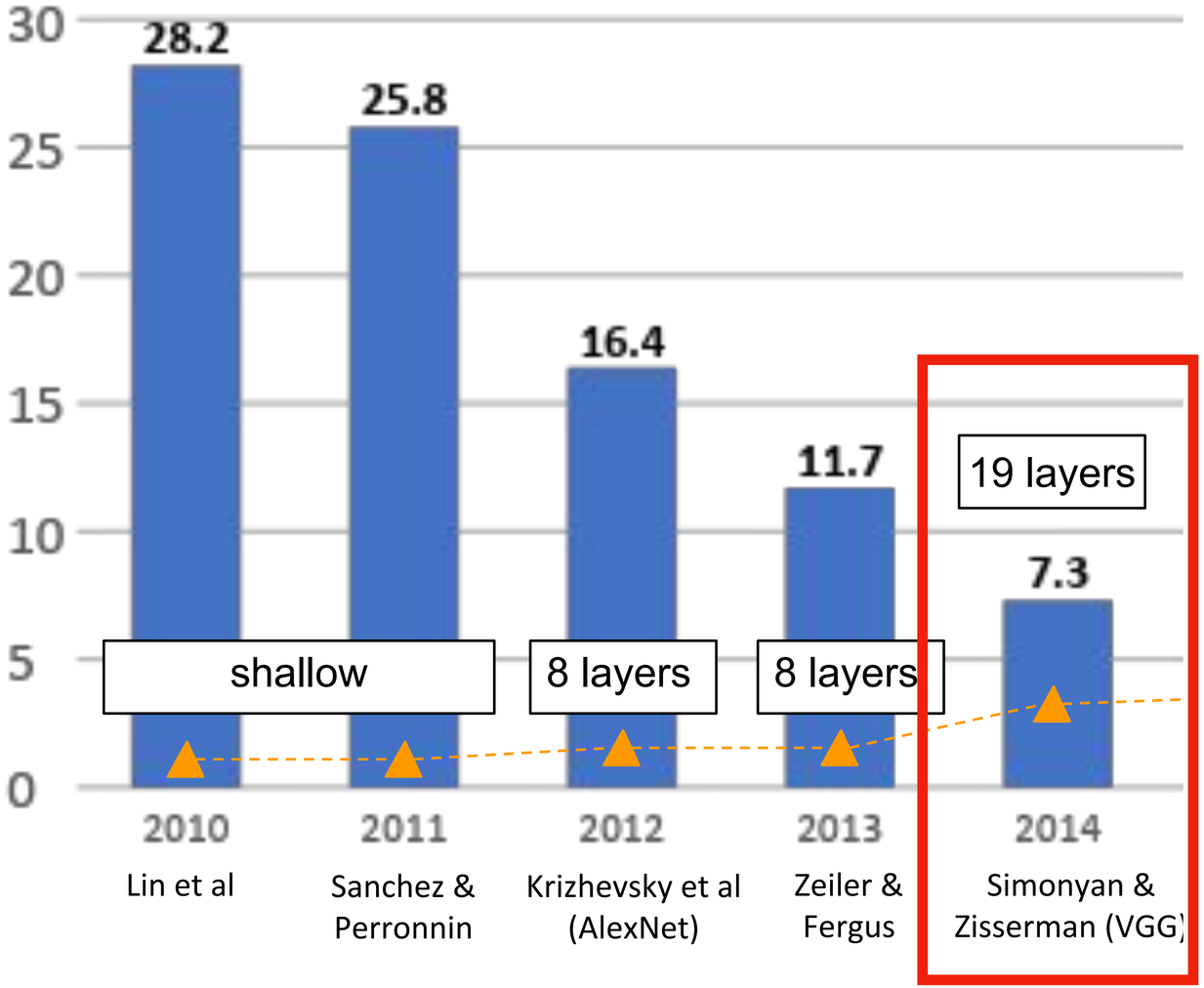
- Essentially AlexNet, but with a smaller CONV1 and higher-channel CONV3,4,5
- Reduced error 5%!
- Better known for cool “deconvolutional” visualizations

ILSVRC Winners



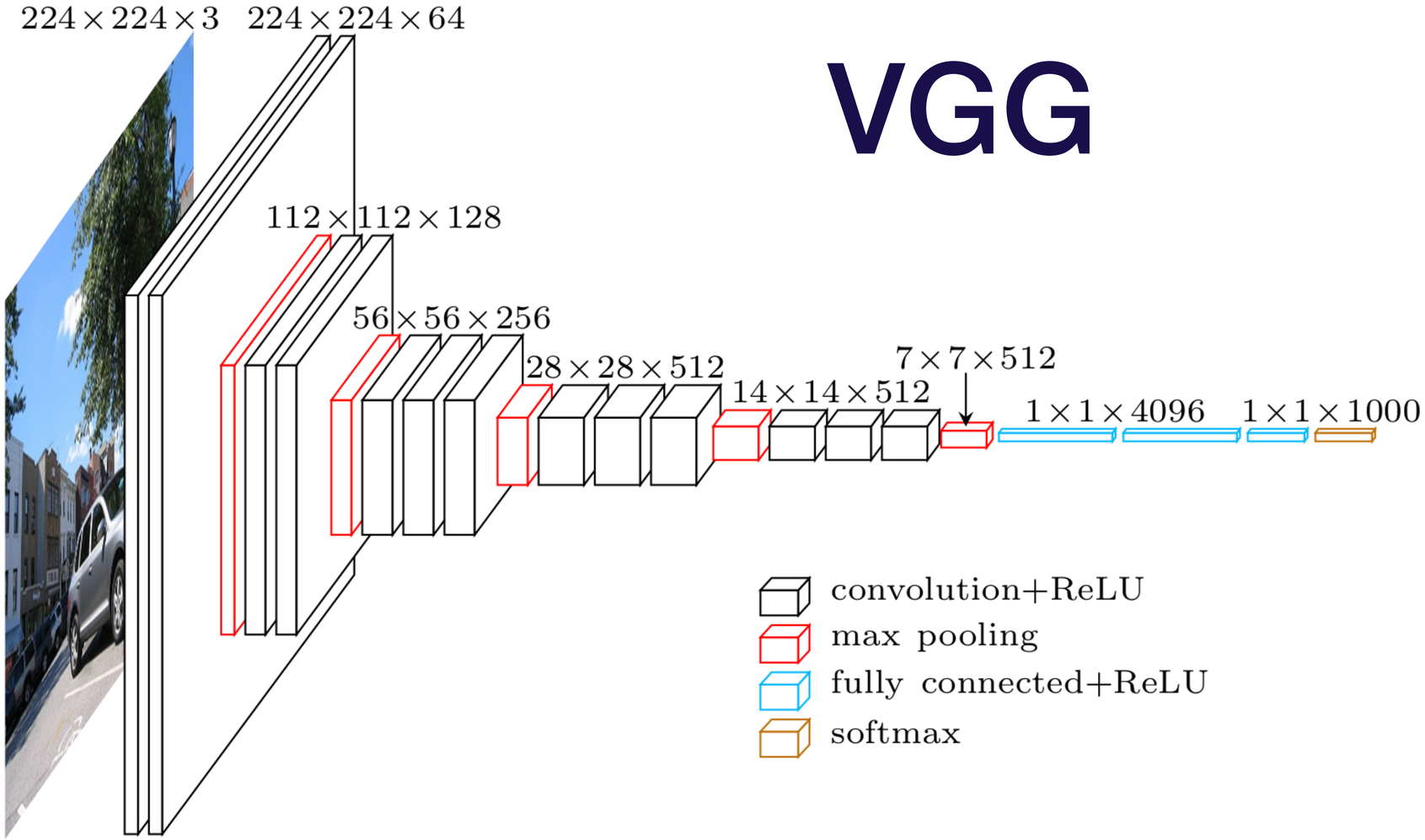
http://cs231n.stanford.edu/

ILSVRC Winners



http://cs231n.stanford.edu/

VGG



4 stacked 3x3 convs get the same receptive field as a 9x9 conv, but use fewer parameters

- Simple but deep architecture of only 3x3 stride-1 convs, 2x2 stride-2 pools.
- Channel dimension increases with each layer.
- From 11.7% (ZFNet) down to 7.3% top-5 error

https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/

VGG16

(not counting biases)

INPUT: [224x224x3] memory: 224*224*3=150K params: 0
CONV3-64: [224x224x64] memory: **224*224*64=3.2M** params: (3*3*3)*64 = 1,728
CONV3-64: [224x224x64] memory: **224*224*64=3.2M** params: (3*3*64)*64 = 36,864
POOL2: [112x112x64] memory: 112*112*64=800K params: 0
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*64)*128 = 73,728
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*128)*128 = 147,456
POOL2: [56x56x128] memory: 56*56*128=400K params: 0
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*128)*256 = 294,912
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
POOL2: [28x28x256] memory: 28*28*256=200K params: 0
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*256)*512 = 1,179,648
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
POOL2: [14x14x512] memory: 14*14*512=100K params: 0
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
POOL2: [7x7x512] memory: 7*7*512=25K params: 0
FC: [1x1x4096] memory: 4096 params: 7*7*512*4096 = **102,760,448**
FC: [1x1x4096] memory: 4096 params: 4096*4096 = 16,777,216
FC: [1x1x1000] memory: 1000 params: 4096*1000 = 4,096,000

TOTAL memory: 24M * 4 bytes ~= 96MB / image (only forward! ~*2 for bwd)
TOTAL params: 138M parameters

Note:

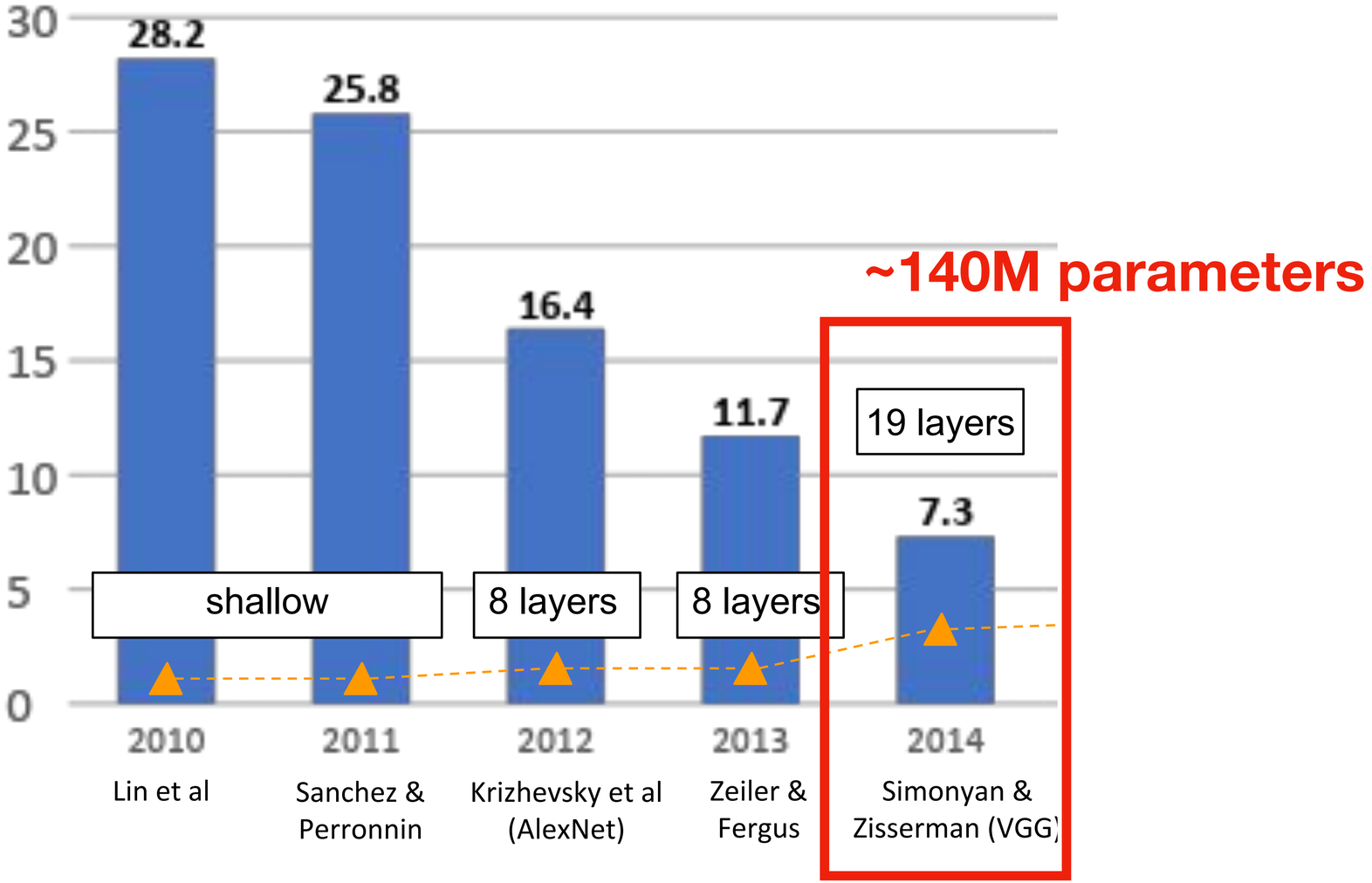
Most memory is in early CONV

Most params are in late FC

VGG19

http://cs231n.stanford.edu/

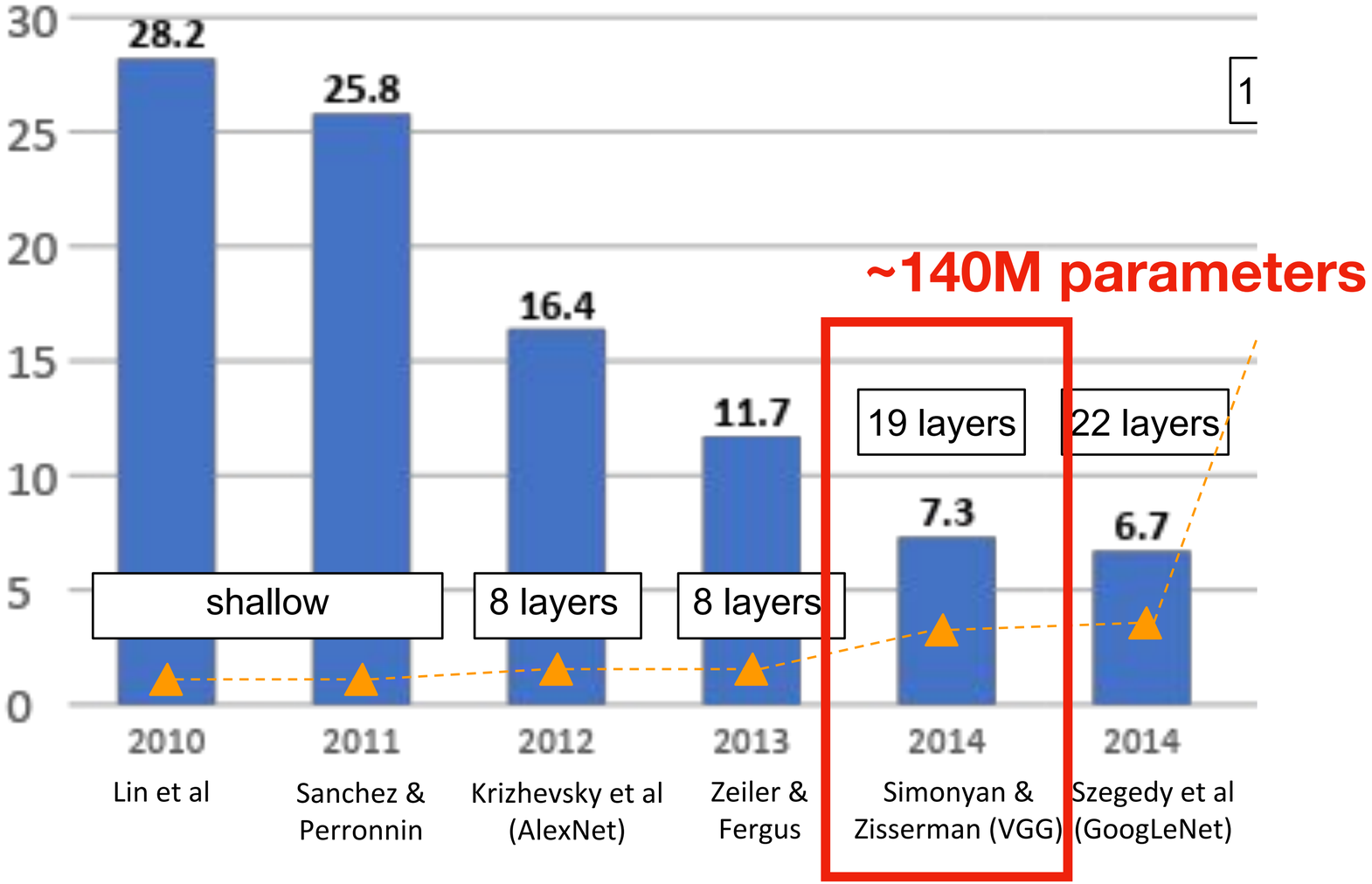
ILSVRC Winners



~140M parameters

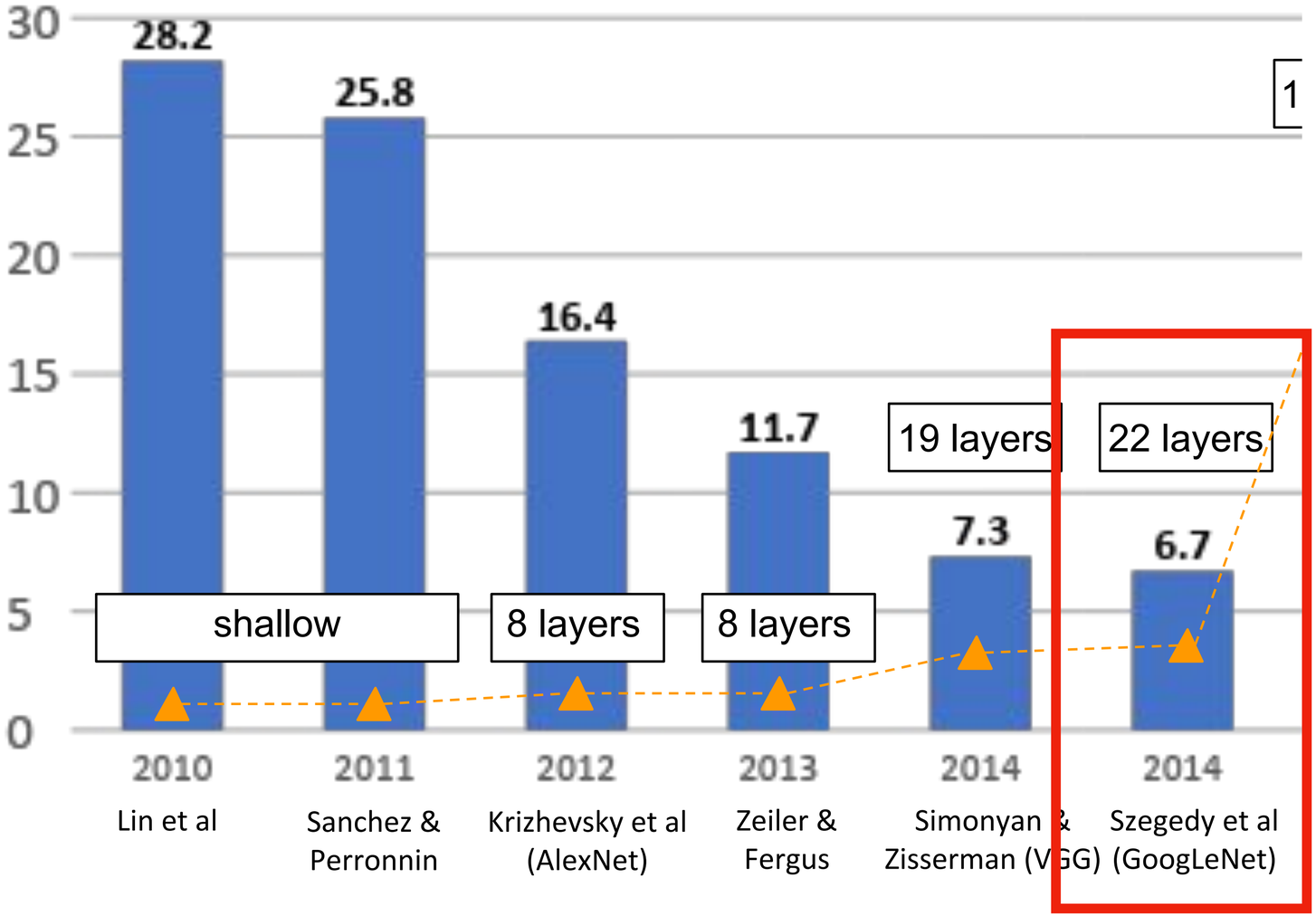
http://cs231n.stanford.edu/

ILSVRC Winners



http://cs231n.stanford.edu/

ILSVRC Winners

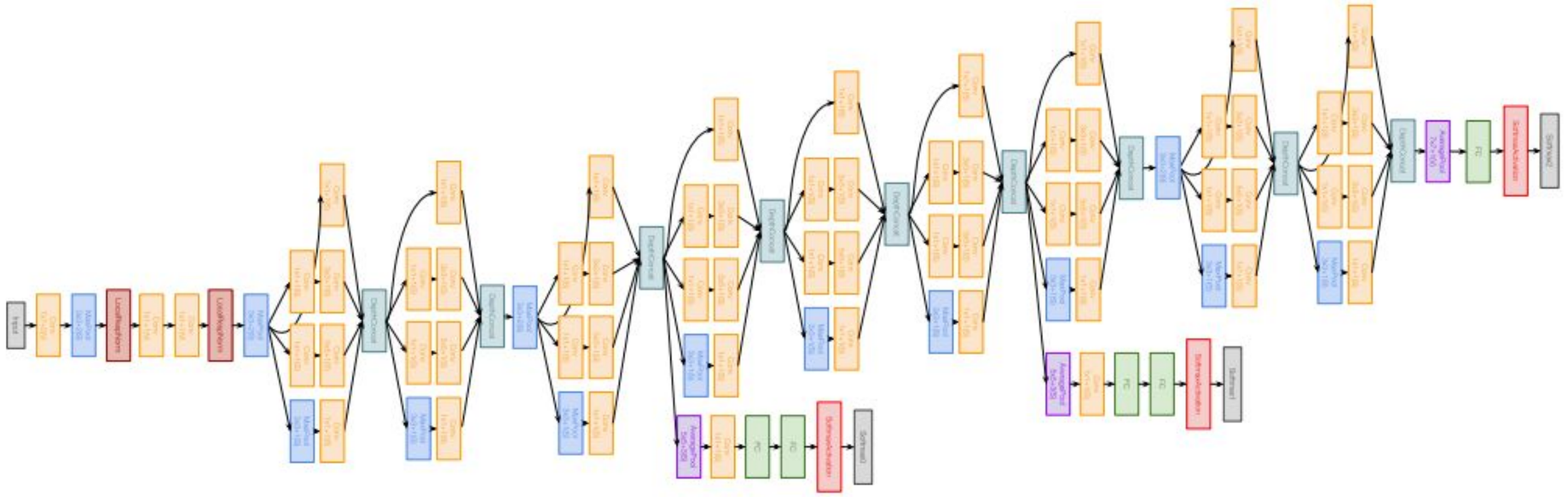


http://cs231n.stanford.edu/

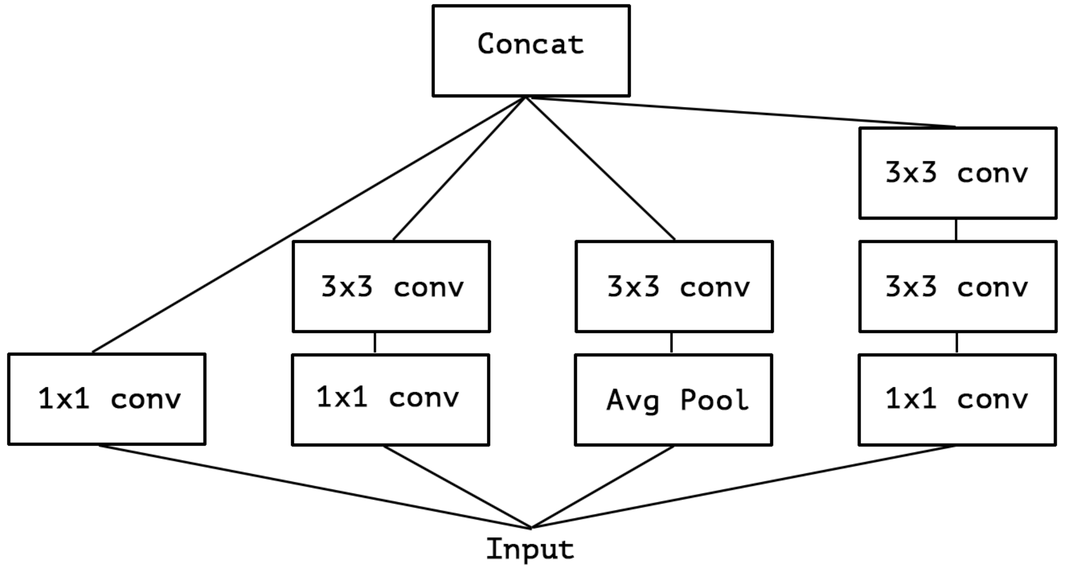
[Szegedy et al., 2014]

GoogLeNet

- Just as deep as VGG, but has only 3% of the parameters!
- No fully-connected layers
- Is just a stack of Inception modules, which is a network module more creative than the standard (conv->relu->conv->relu->pool)



GoogLeNet: Inception Module

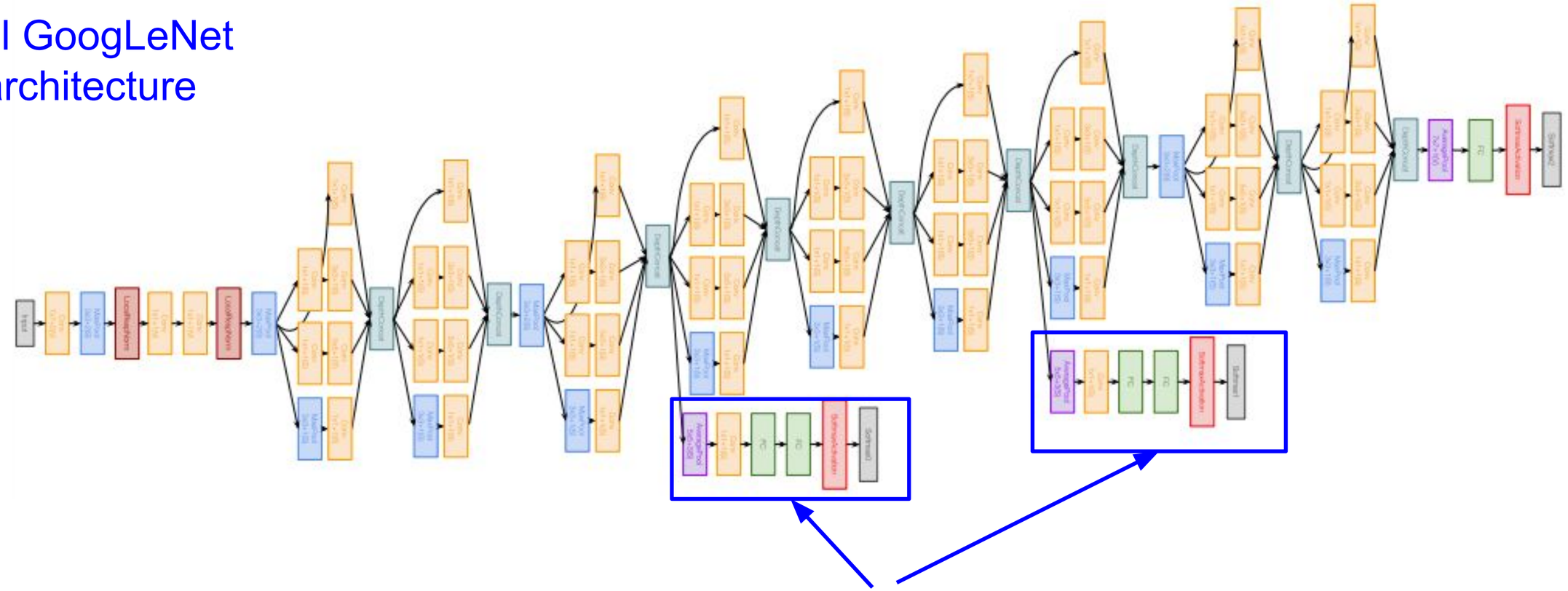


- Instead of picking 1x1, 3x3, 5x5, or pooling, send each input through all of them, and concatenate the results depthwise.
- Reduce with 1x1 conv first.
  - This reduces the amount of computation and follows the “Inception hypothesis”: cross-channel correlations and spatial correlations are decoupled and can be mapped separately.

Xception: Deep Learning with Depthwise Separable Convolutions François Chollet 2017

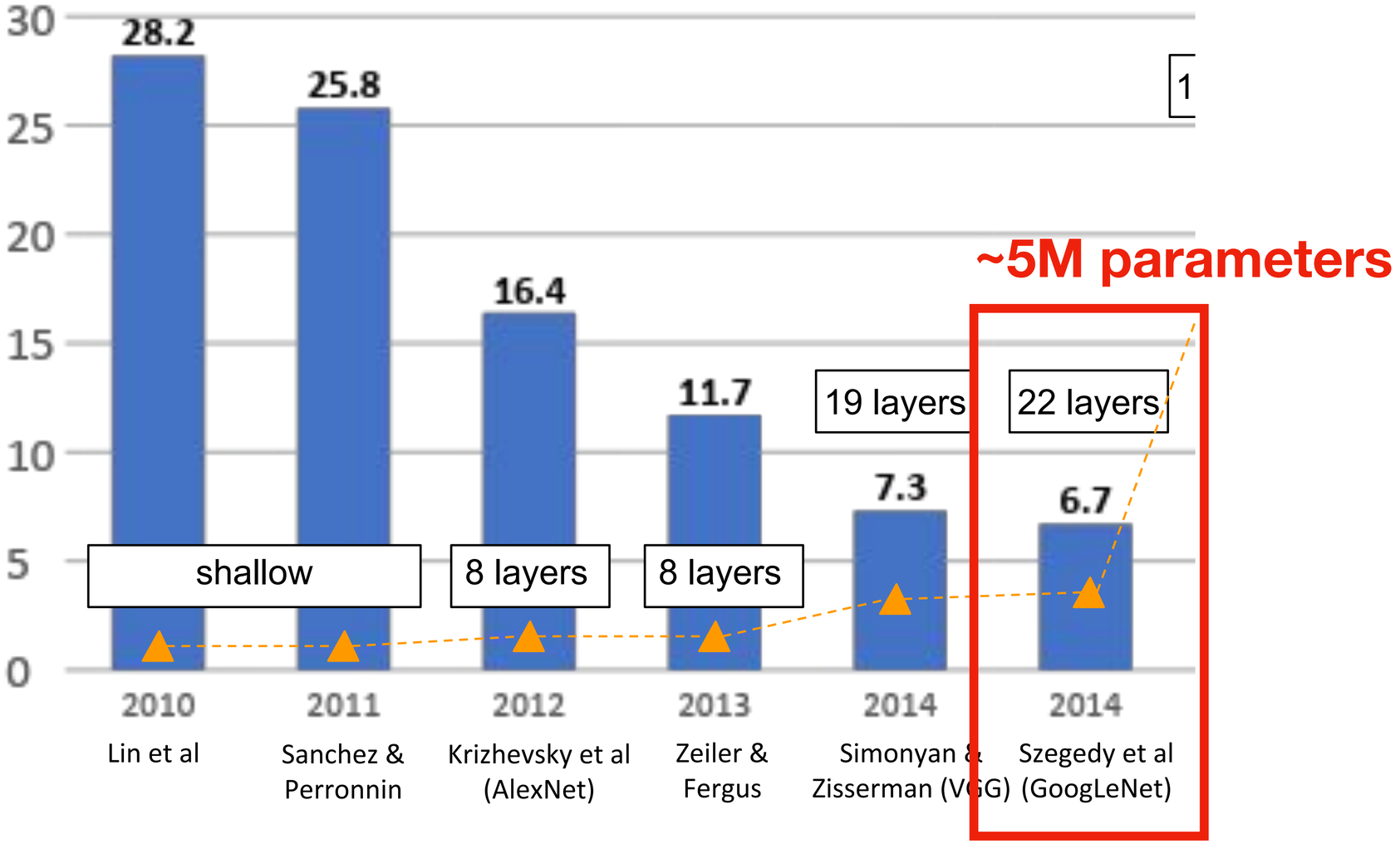
GoogLeNet

Full GoogLeNet architecture



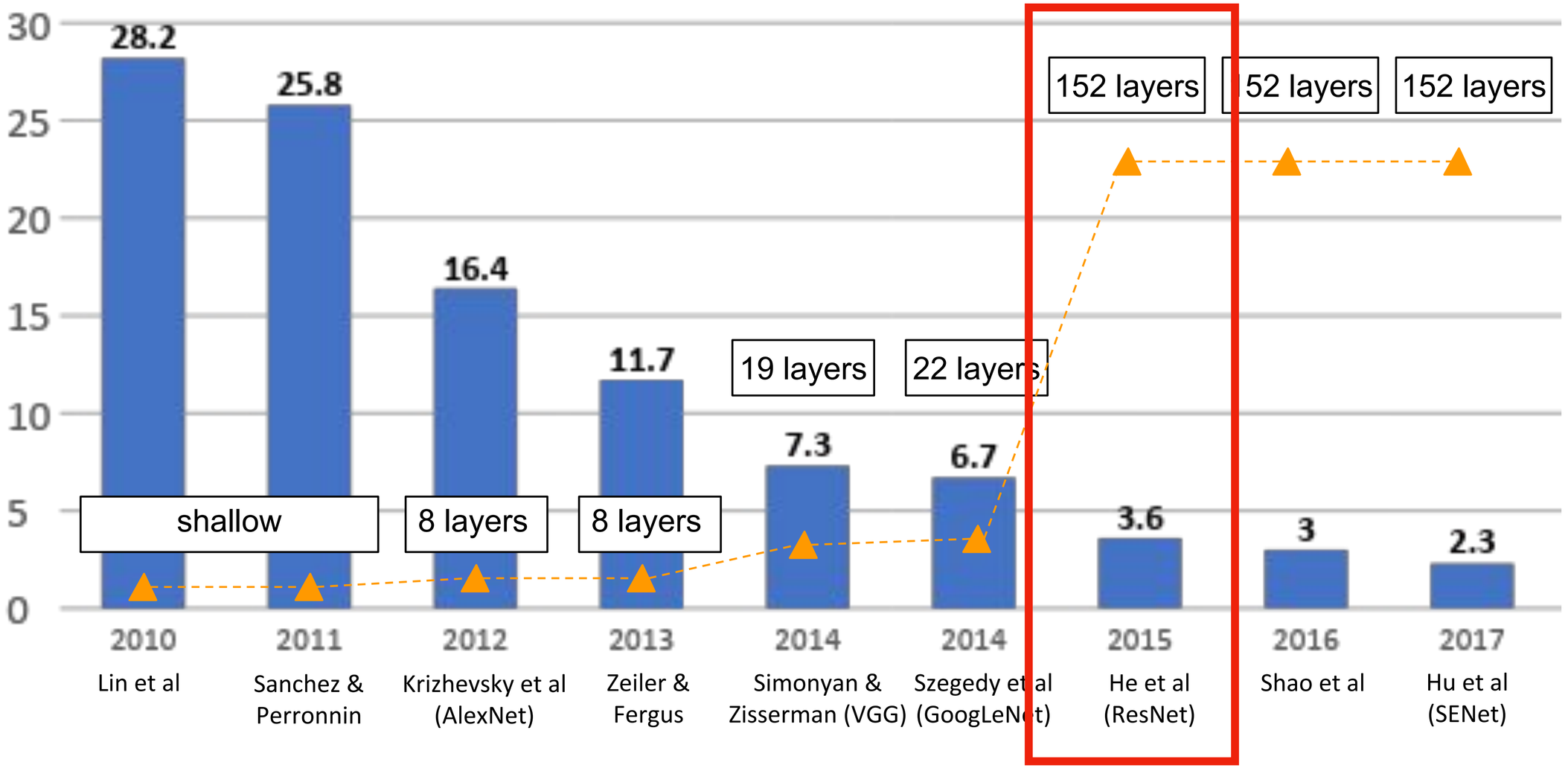
- Another cool trick: additional inject classifier gradient earlier in the network
- And, remove FC layers for speed

ILSVRC Winners



http://cs231n.stanford.edu/

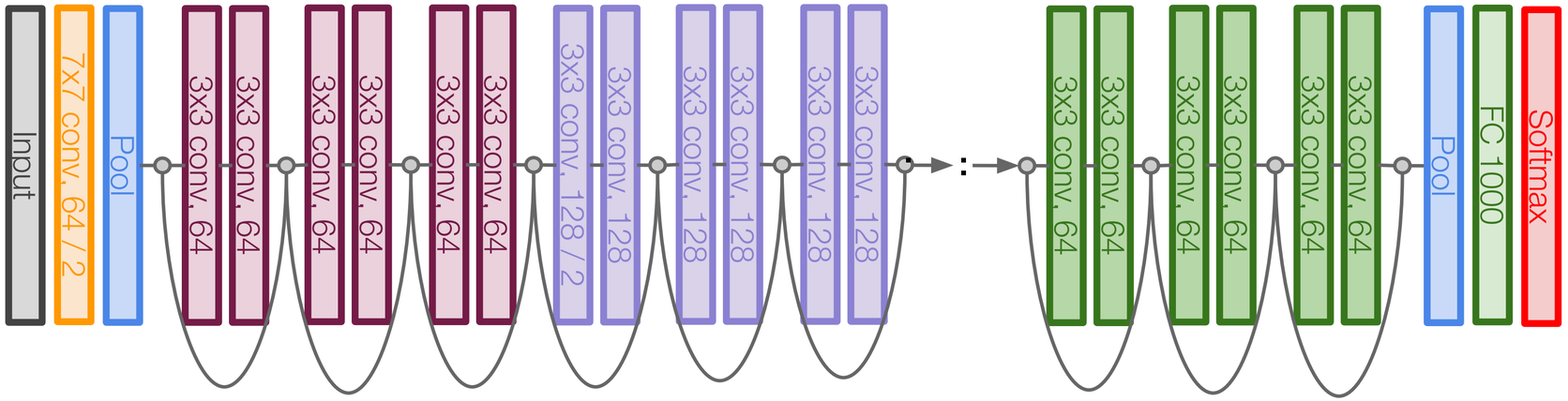
ILSVRC Winners

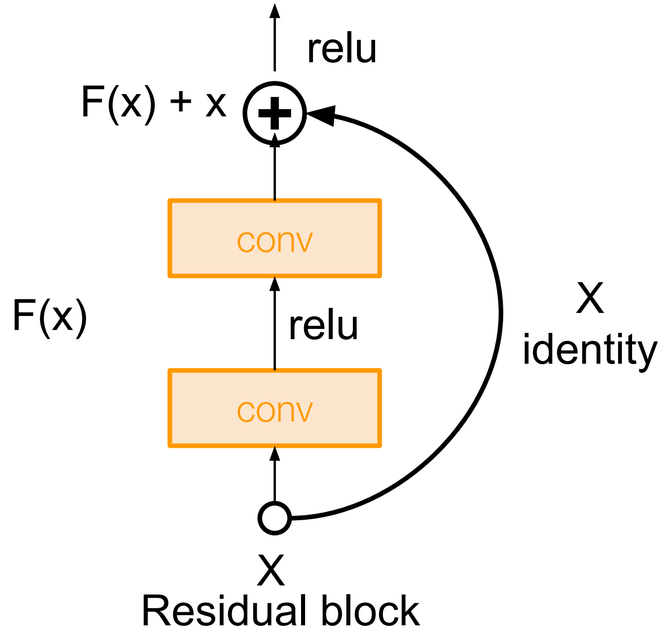


http://cs231n.stanford.edu/

ResNet

[He et al., 2015]

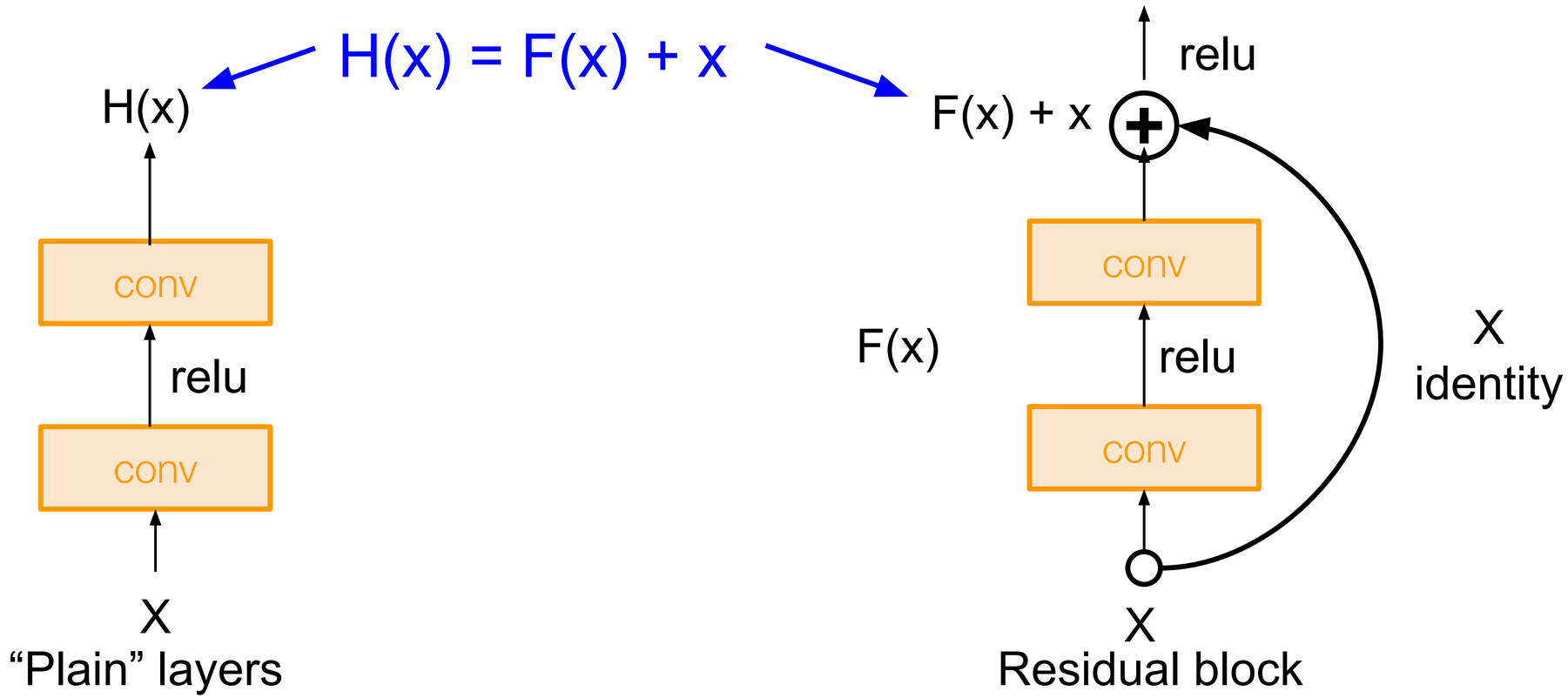




- Down to 3.57% top-5 error (lower than human performance of 5%)
- Problem: deeper models should be able to perform **at least** as well as shallower networks, but don’t due to vanishing gradient
- Solution: add an option to simply skip layers, such that if the gradient vanishes, at least no harm is done!

ResNet

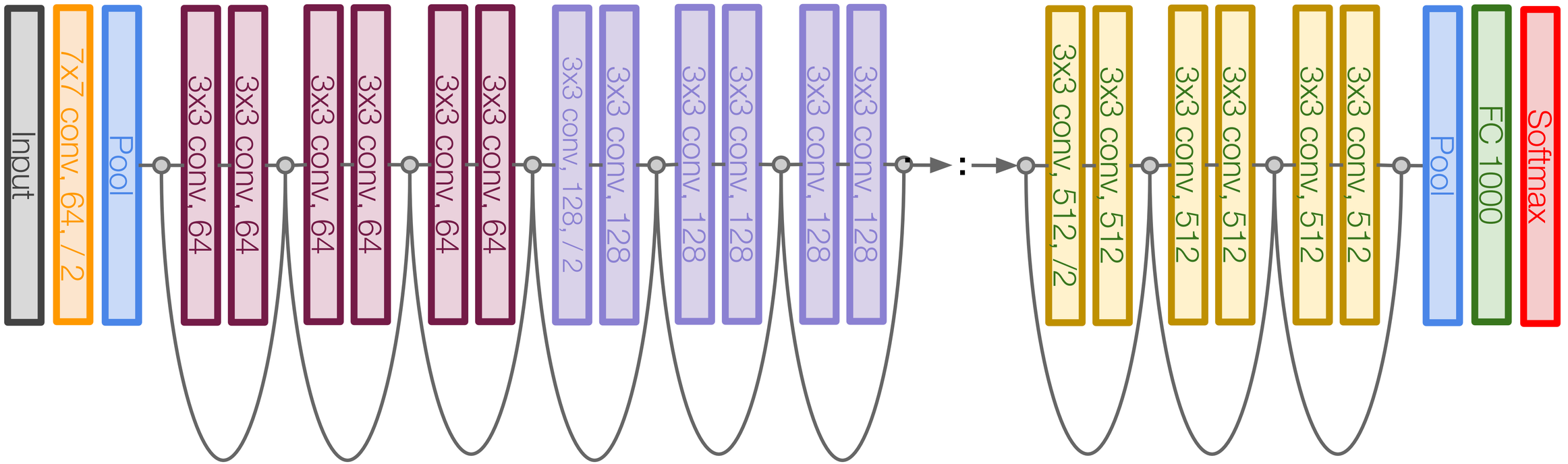
[He et al., 2015]



- Why "residual"? Because fitting F(x) = H(x) - x

ResNet

[He et al., 2015]

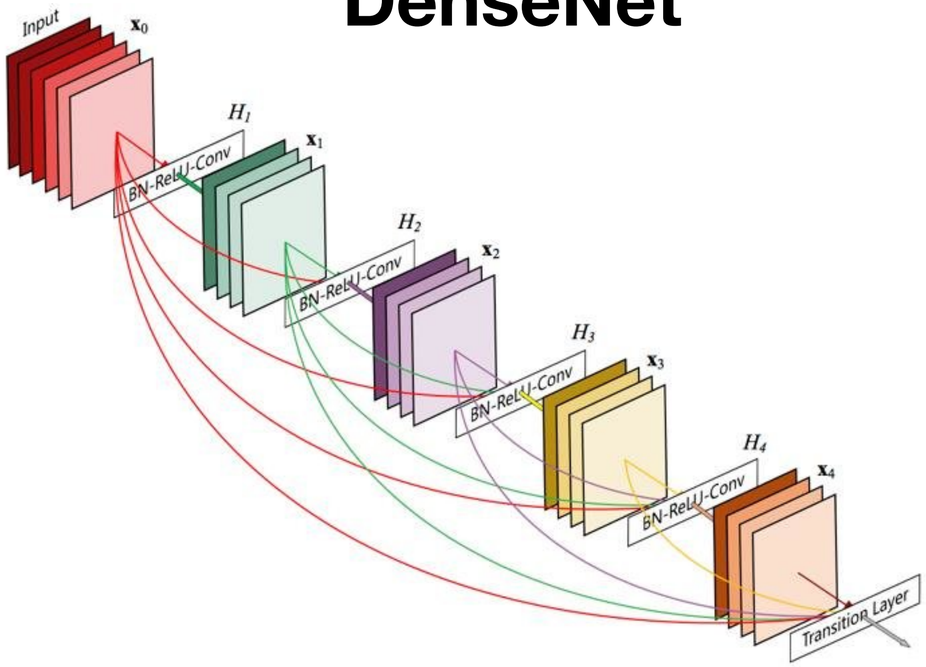


- Increase number of filters with depth; downsample spatially using strides.

http://cs231n.stanford.edu/

Residual Net Variants

DenseNet



- Add more skip connections!

ResNeXt

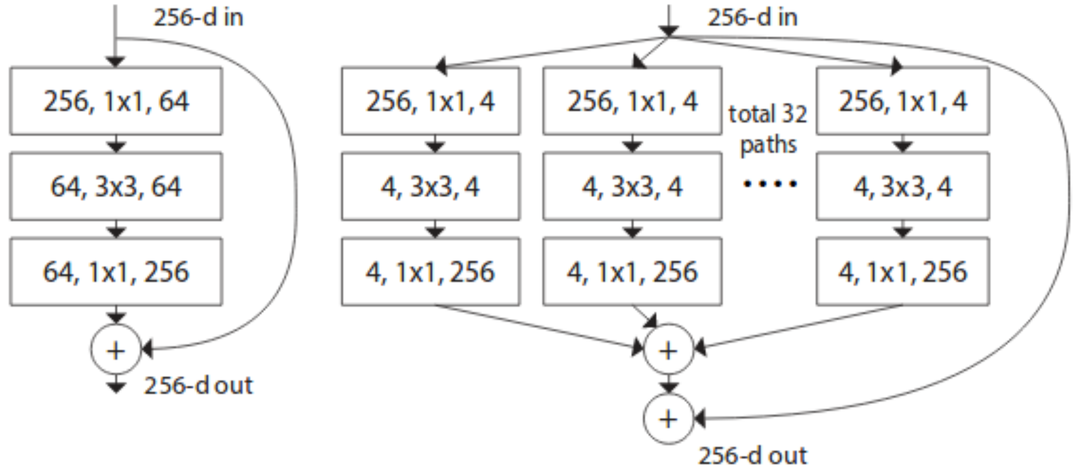
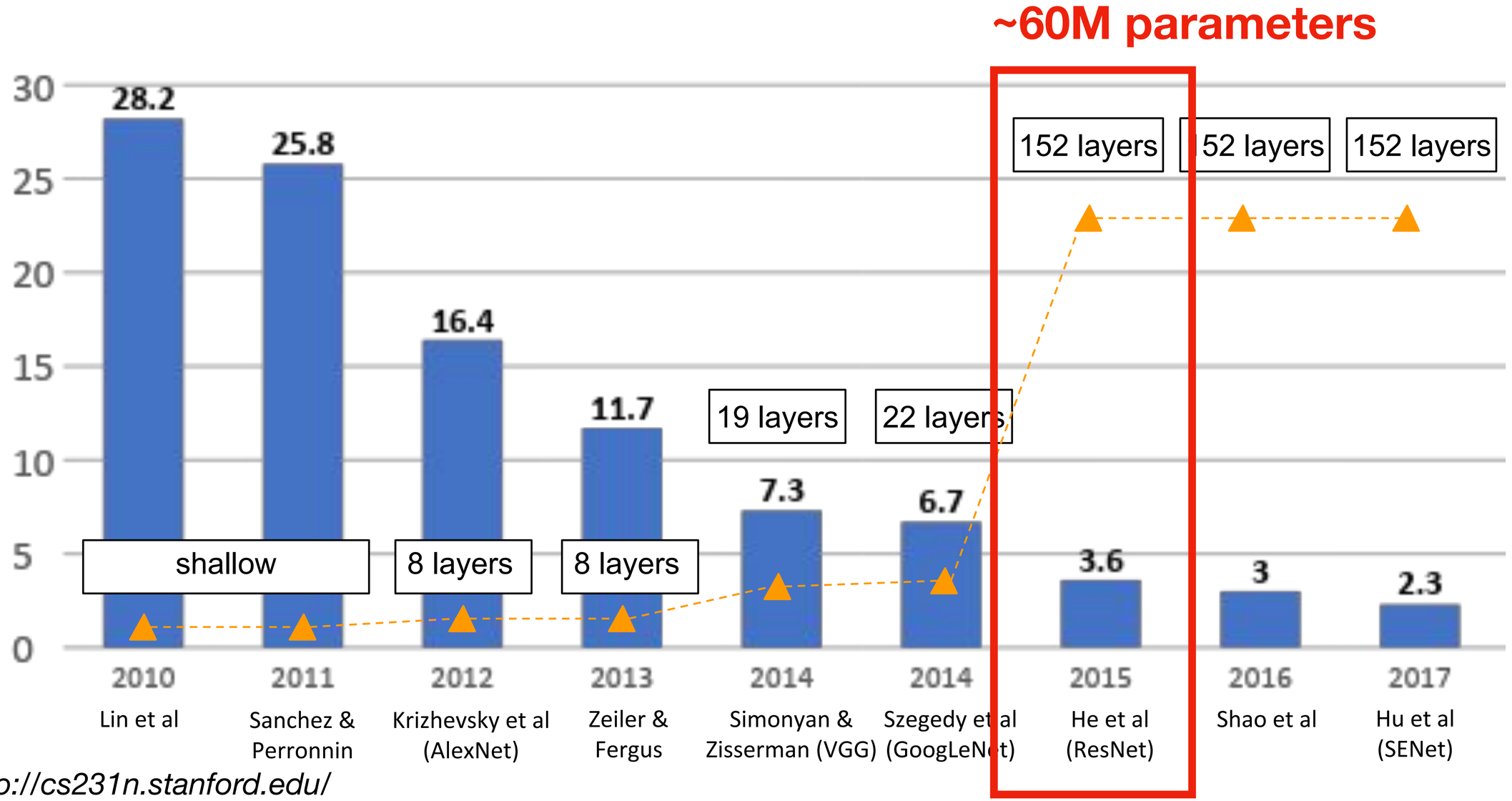


Figure 1. **Left**: A block of ResNet [14]. **Right**: A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).

- C*ardinality* of a module: the amount of paths through split/ transform/aggregate

http://cs231n.stanford.edu/

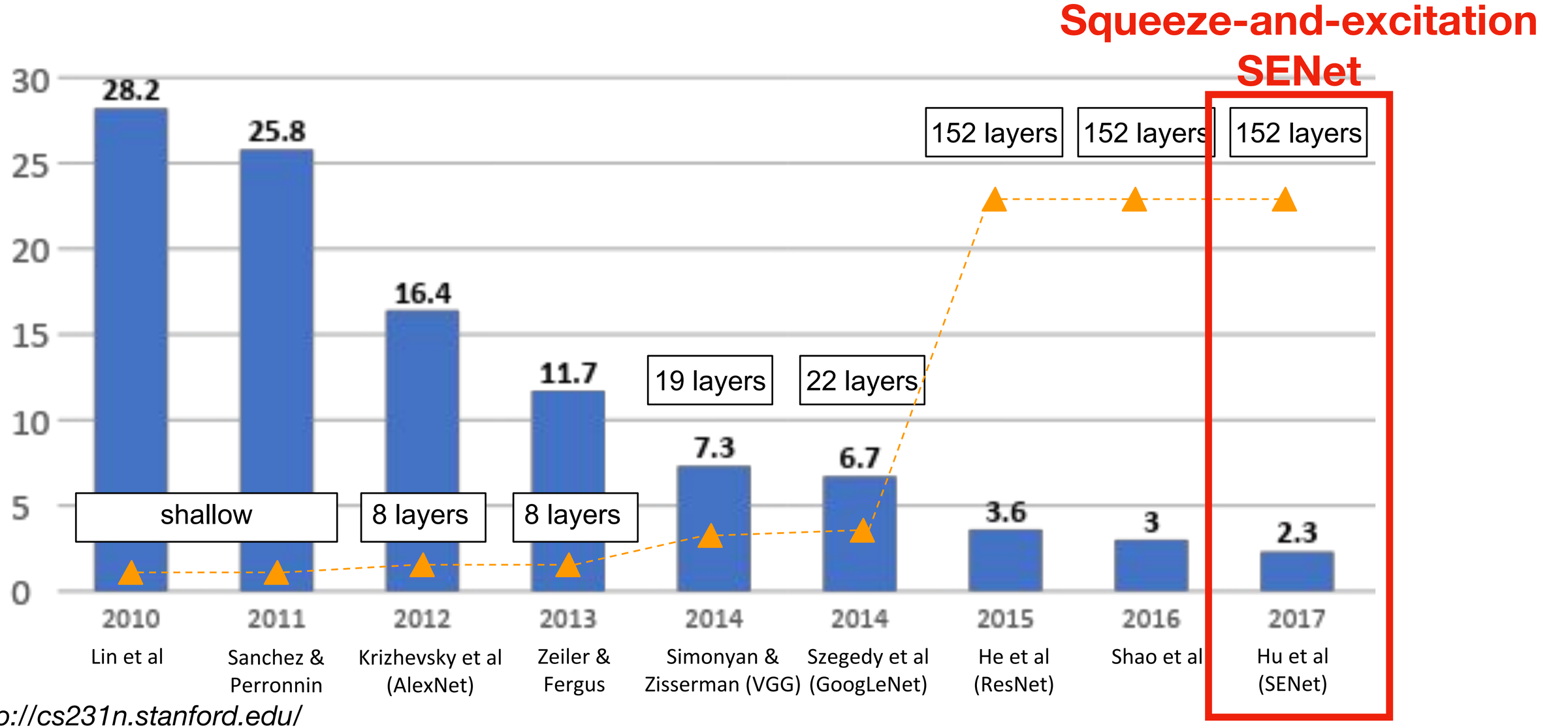
ILSVRC Winners



http://cs231n.stanford.edu/

https://resources.wolframcloud.com/NeuralNetRepository/resources/ResNet-152-Trained-on-ImageNet-Competition-Data

ILSVRC Winners

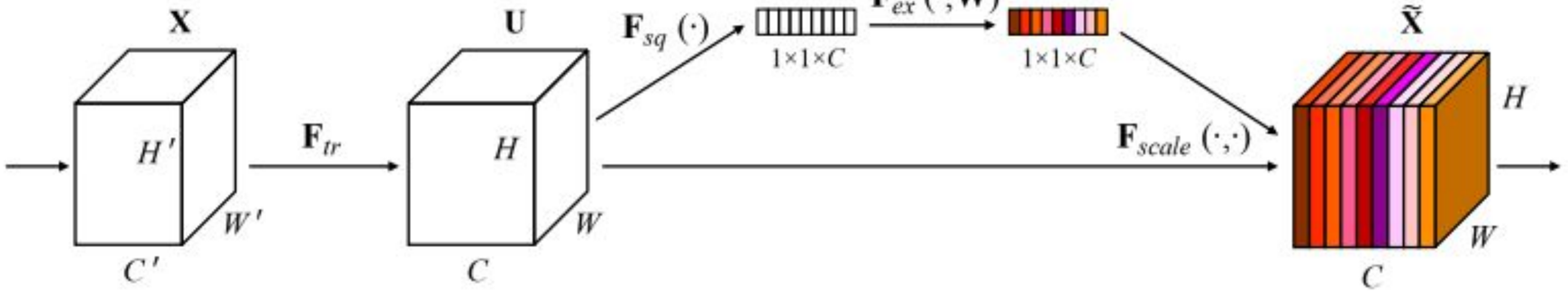


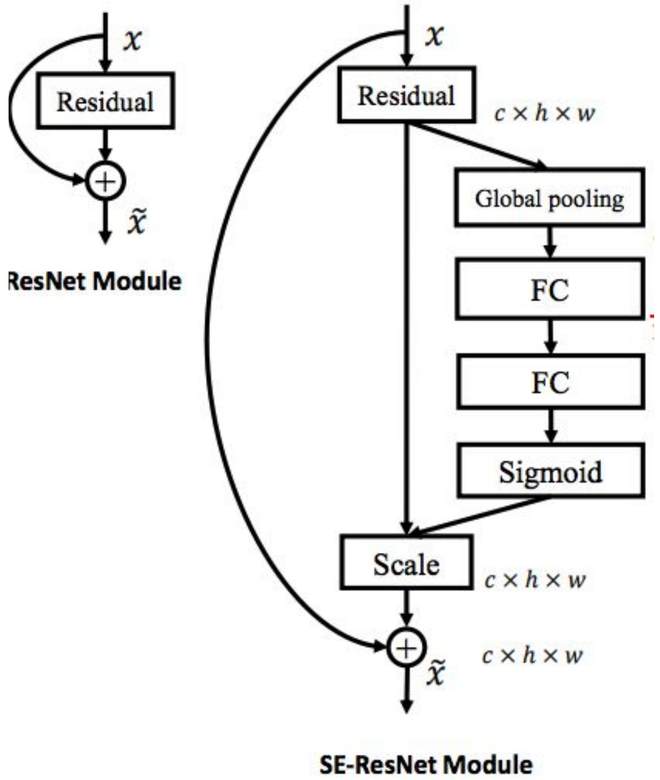
http://cs231n.stanford.edu/

https://resources.wolframcloud.com/NeuralNetRepository/resources/ResNet-152-Trained-on-ImageNet-Competition-Data

SENet

[Hu et al. 2017]

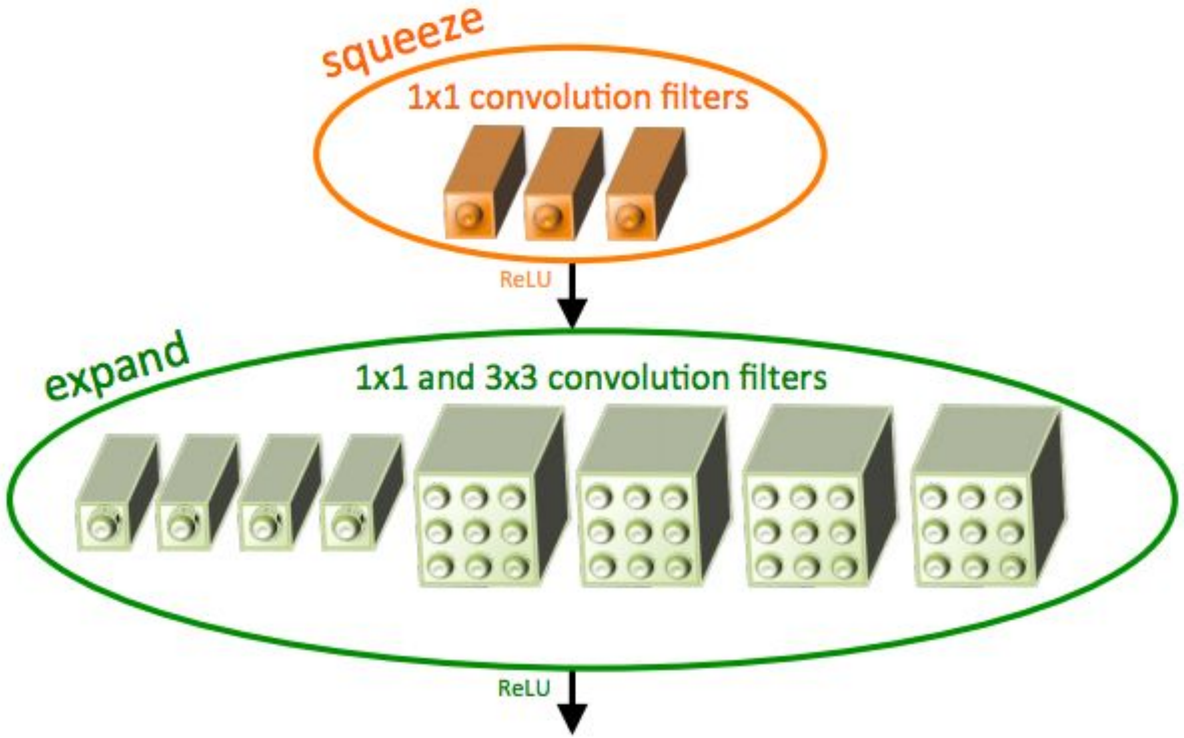


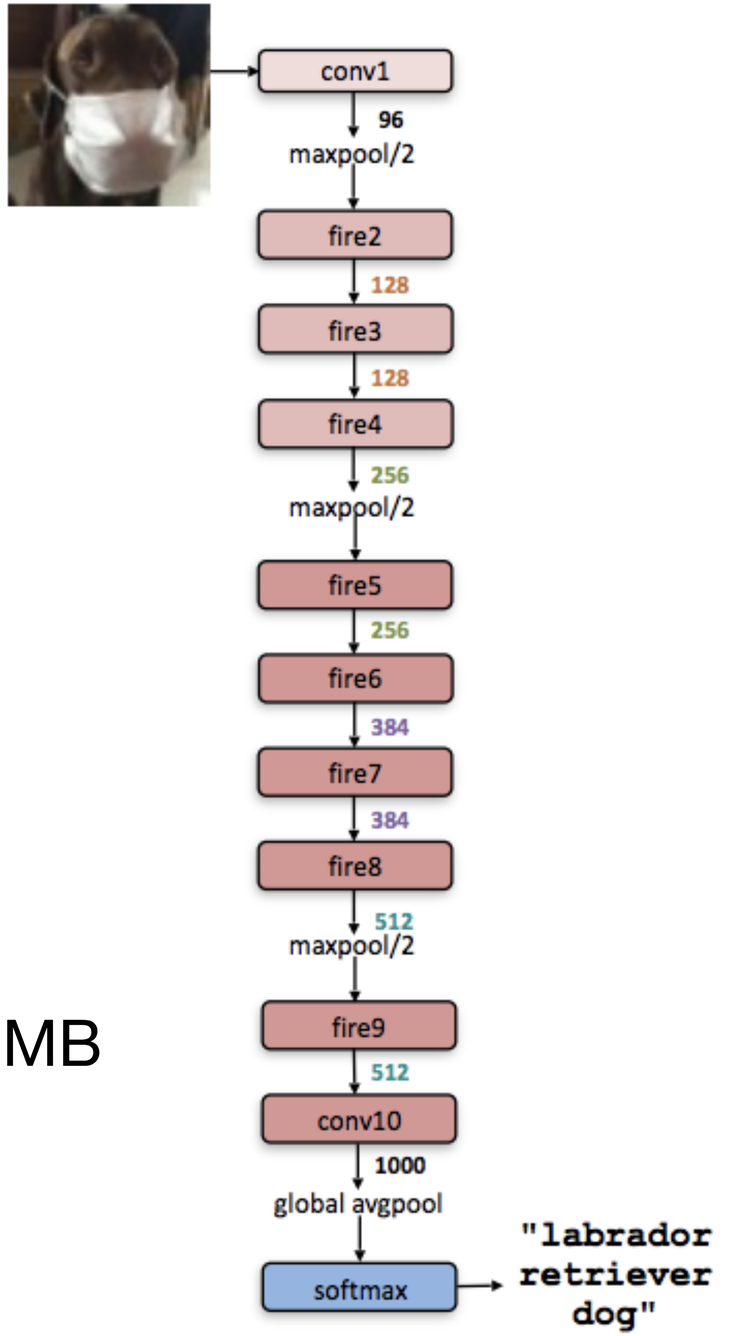


- Add a module of global pooling + FC to adaptively reweight feature output maps
- Kind of like attention: the network gets to choose what to use

[Iandola et al. 2017]

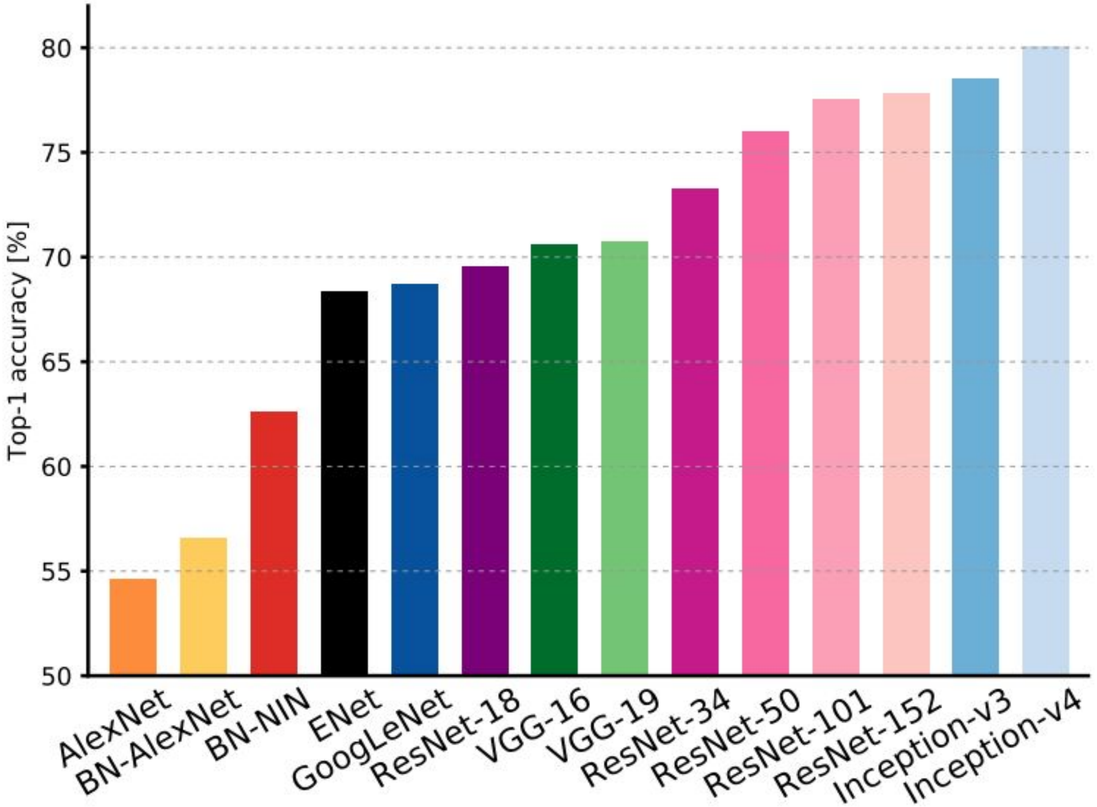
SqueezeNet

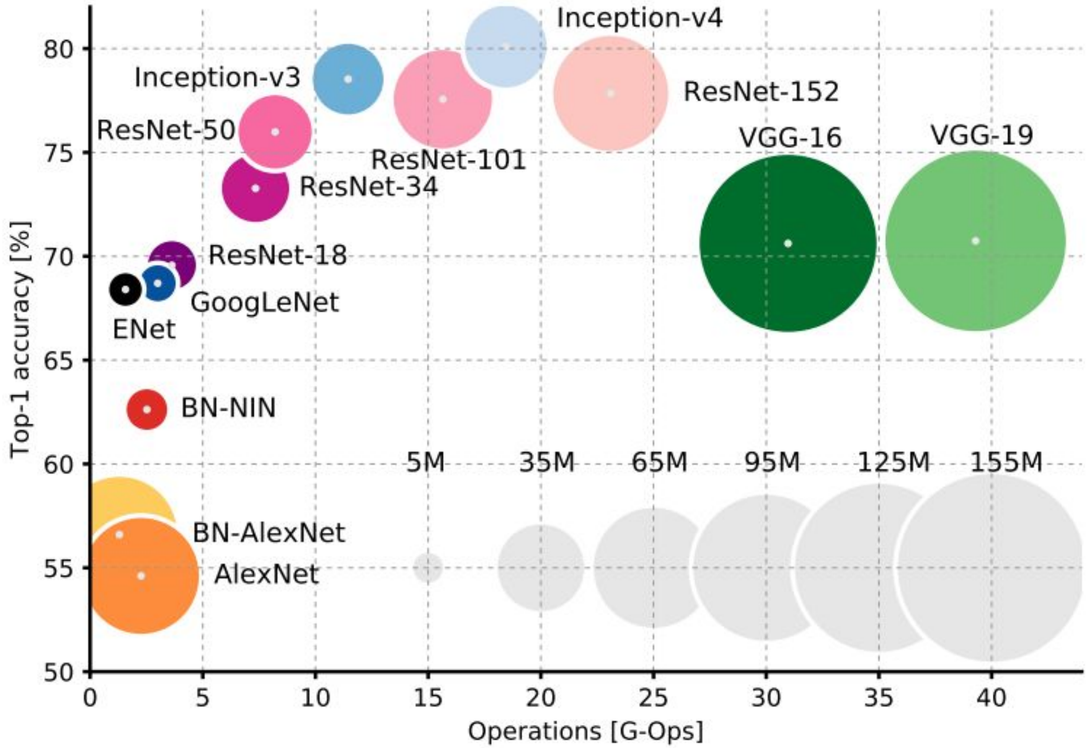




- AlexNet-level accuracy with 50x fewer params and <0.5MB model, through constant 1x1 bottlenecking
- Interesting for hardware/mobile deployment

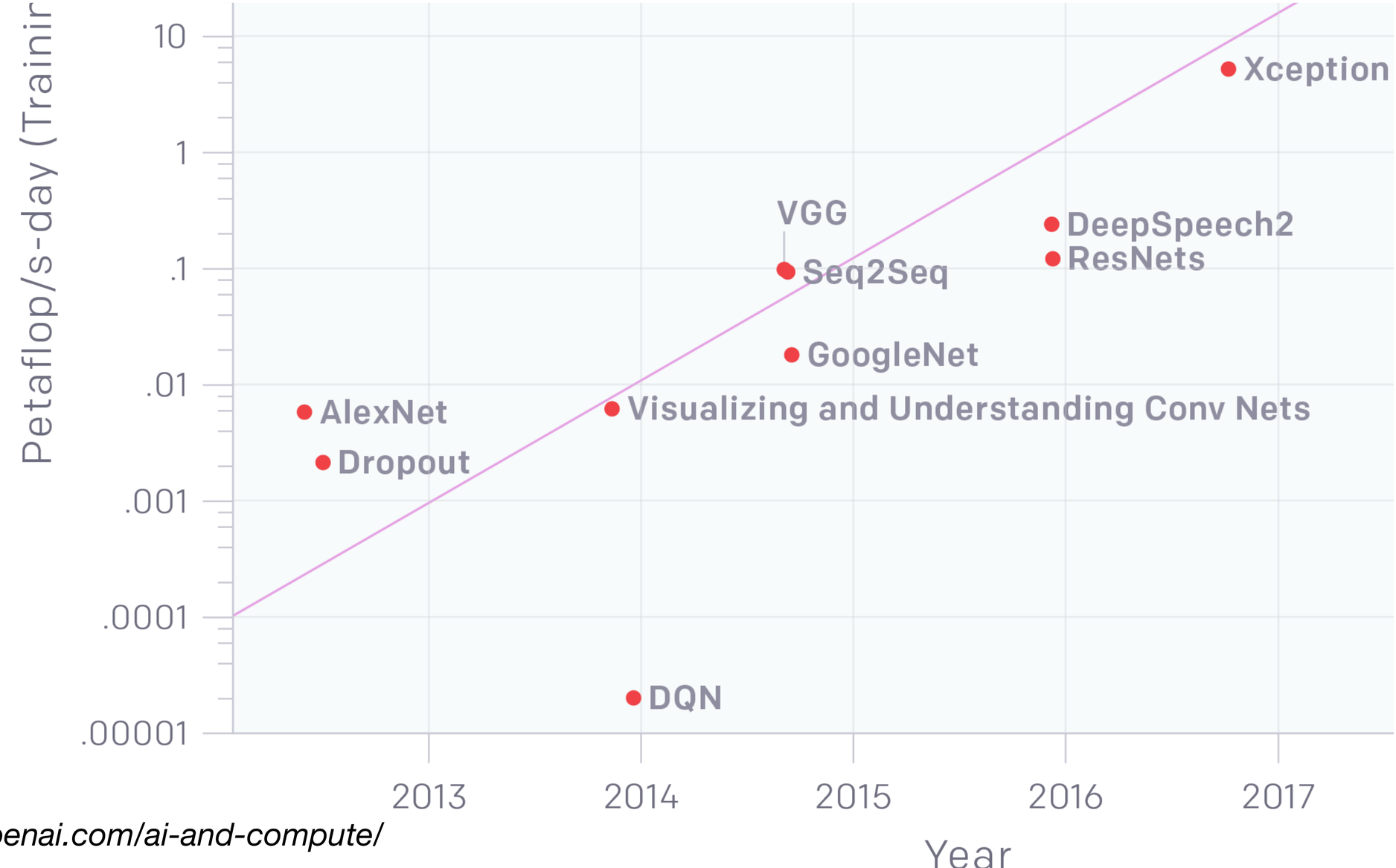
Overall Comparison





- Some ResNets, or Inception-v4 (Inception + ResNet) are probably in the sweet spot of accuracy to operations / memory.

Compute Power



https://blog.openai.com/ai-and-compute/

Fast Training

Extremely Large Minibatch SGD: Training ResNet-50 on ImageNet in 15 Minutes

Takuya Akiba, Shuji Suzuki, Keisuke Fukuda

(Submitted on 12 Nov 2017)

We demonstrate that training ResNet-50 on ImageNet for 90 epochs can be achieved in 15 minutes with 1024 Tesla P100 GPUs. This was made possible by using a large minibatch size of 32k. To maintain accuracy with this large minibatch size, we employed several techniques such as RMSprop warm-up, batch normalization without moving averages, and a slow-start learning rate schedule. This paper also describes the details of the hardware and software of the system used to achieve the above performance.

https://arxiv.org/abs/1711.04325

Fast Training

DAWNBench About Submit Stanford DAWN

- Image Classification (ImageNet)
  - Training Time
  - Training Cost
  - Inference Latency
  - Inference Cost
- Image Classification (CIFAR10)
- Question Answering (SQuAD)

Image Classification on ImageNet

Training Time

All Submissions

Objective: Time taken to train an image classification model to a top-5 validation accuracy of 93% or greater on ImageNet.

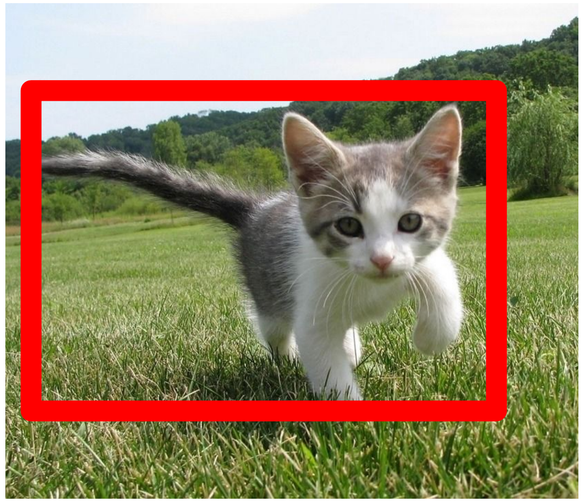
Rank	Time to 93% Accuracy	Model	Hardware	Framework
1 Mar 2020	0:02:38	ResNet50-v1.5 *Apsara AI Acceleration(AIACC) team in Alibaba Cloud* source	16 ecs.gn6e-c12g1.24xlarge (AlibabaCloud)	AIACC-Training 1.3 + Tensorflow 2.1
2 May 2019	0:02:43	ResNet-50 *ModelArts Service of Huawei Cloud* source	16 nodes with InfiniBand (8*V100 with NVLink for each node)	Moxing v1.13.0 + TensorFlow v1.13.1
3 Dec 2018	0:09:22	ResNet-50 *ModelArts Service of Huawei Cloud* source	16 * 8 * Tesla-V100(ModelArts Service)	Huawei Optimized MXNet
4 Sep 2018	0:18:06	ResNet-50 *fast.ai/DIUx (Yaroslav Bulatov, Andrew Shaw, Jeremy Howard)* source	16 p3.16xlarge (AWS)	PyTorch 0.4.1
5 Sep 2018	0:18:53	Resnet 50 *Andrew Shaw, Yaroslav Bulatov, Jeremy Howard* source	64 * V100 (8 machines - AWS p3.16xlarge)	ncluster / Pytorch 0.5.0a0+0e8088d

https://dawn.cs.stanford.edu/benchmark/#imagenet-train-time

Agenda

1. Tour of Convnet Architectures
2. **Localization, Detection, Segmentation**
3. Misc

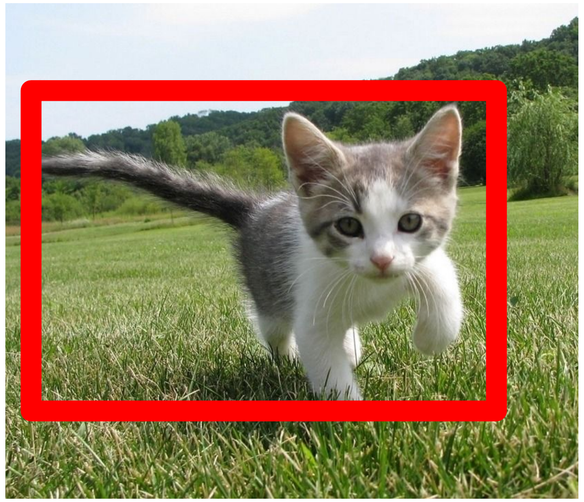
Localization, Detection, Segmentation

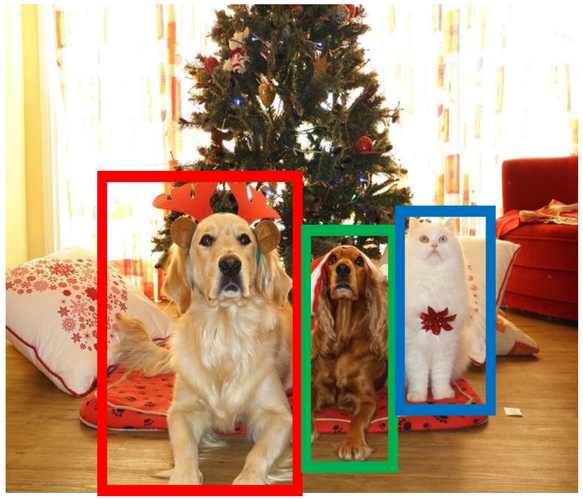


- **Classification:** output the single object's class
- **Localization:** output the single object's class **and its location**

http://cs231n.stanford.edu/

Localization, Detection, Segmentation

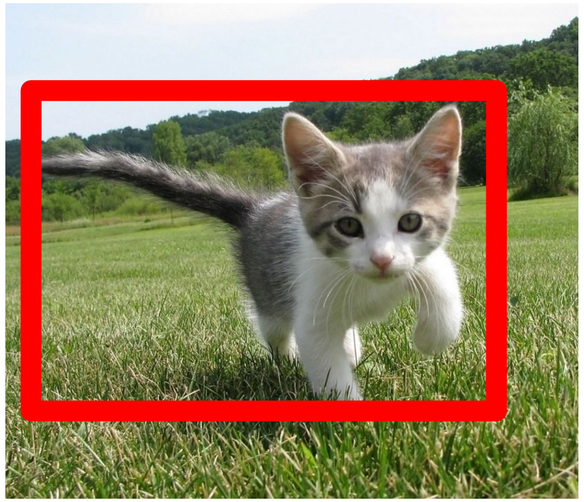
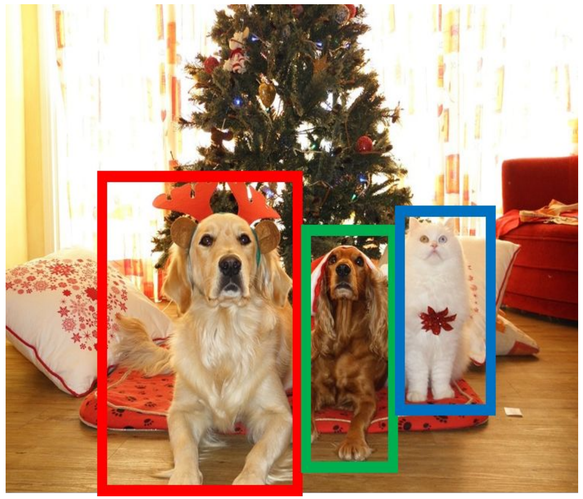
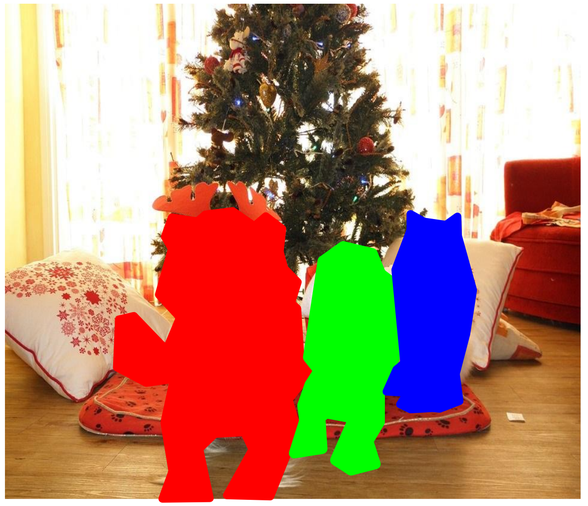




- **Classification:** output the single object's class
- **Localization:** output the single object's class **and its location**
- **Detection:** output **every** object's class and location

http://cs231n.stanford.edu/

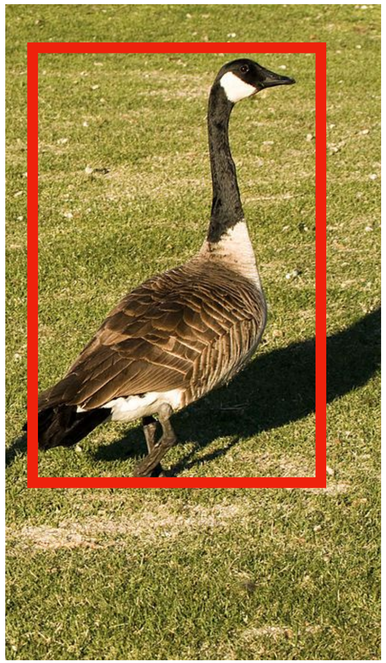
Localization, Detection, Segmentation

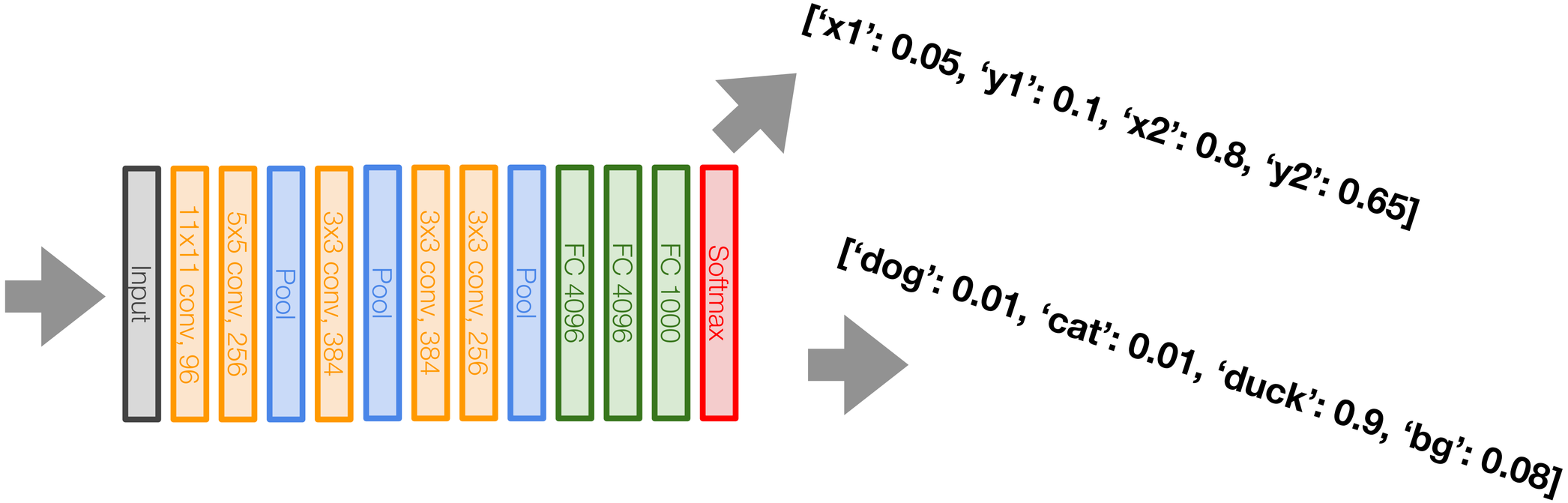




- **Classification:** output the single object's class
- **Localization:** output the single object's class **and its location**
- **Detection:** output **every** object's class and location
- **Segmentation: label every pixel** in the image as belonging to an object or to background
  - **Instance Segmentation** additionally differentiates between different objects of the same class

http://cs231n.stanford.edu/

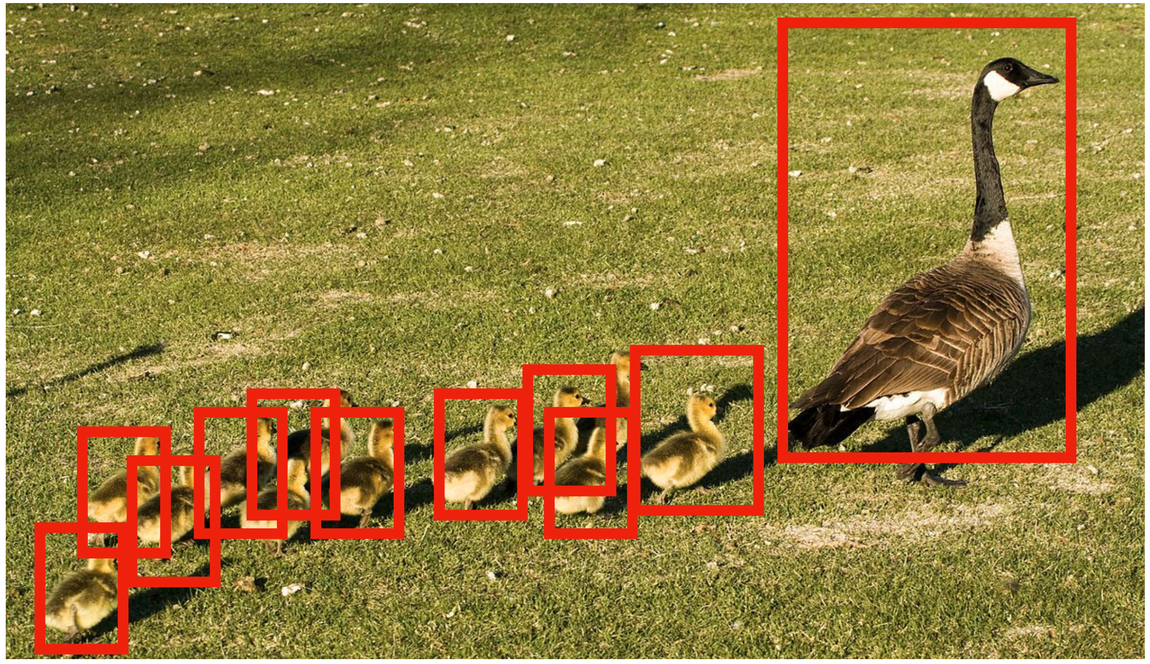
Localization





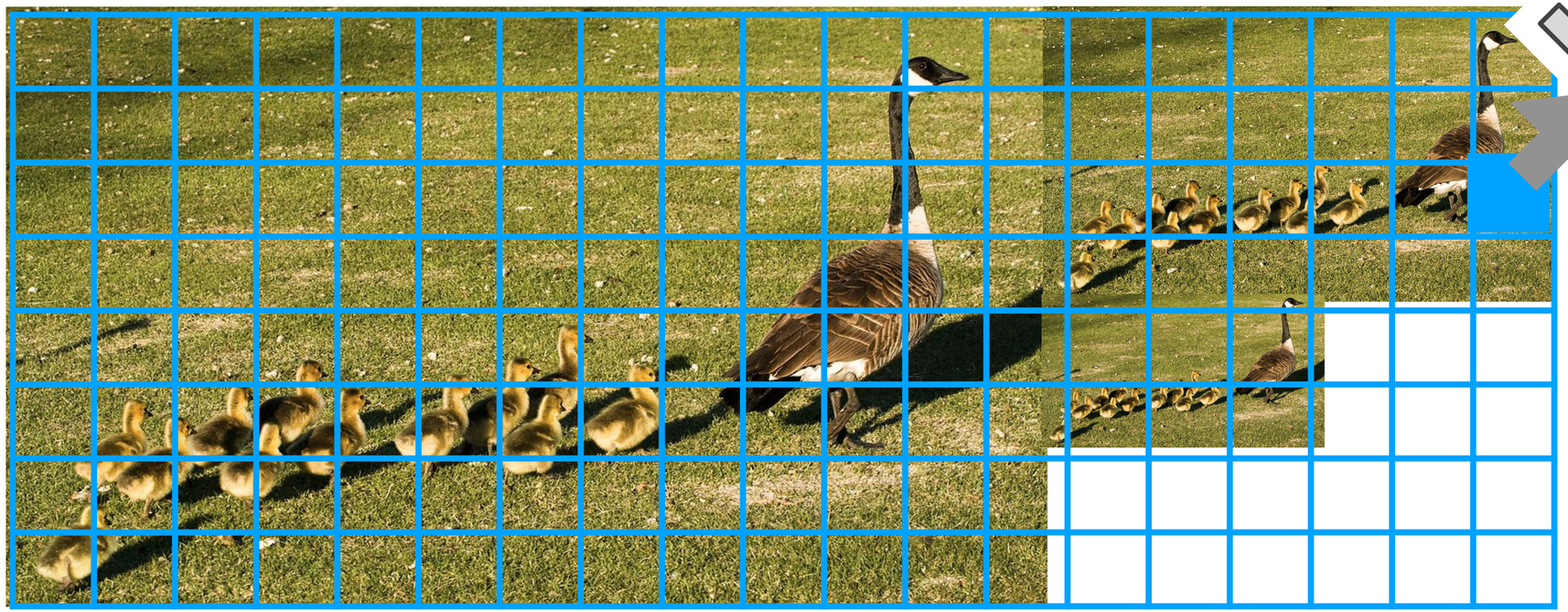
- Simply predict the bounding box coordinates (x1, y1, x2, y2) as well as the class, using the same network.

Detection



- Localization does not scale when there are multiple objects.

Detection

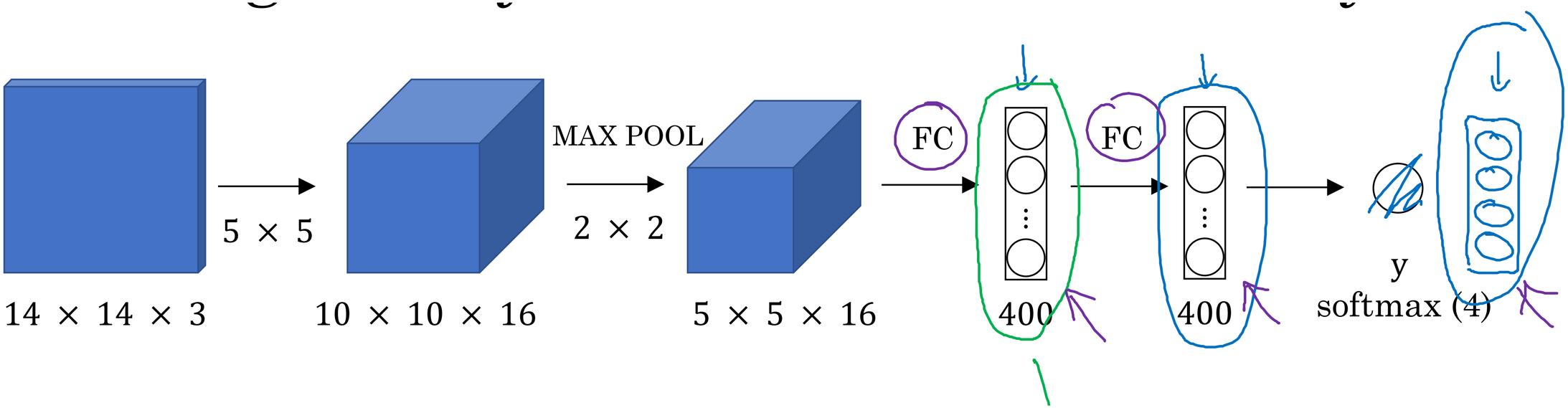


- One solution: slide a classifier over the image (at multiple scales)

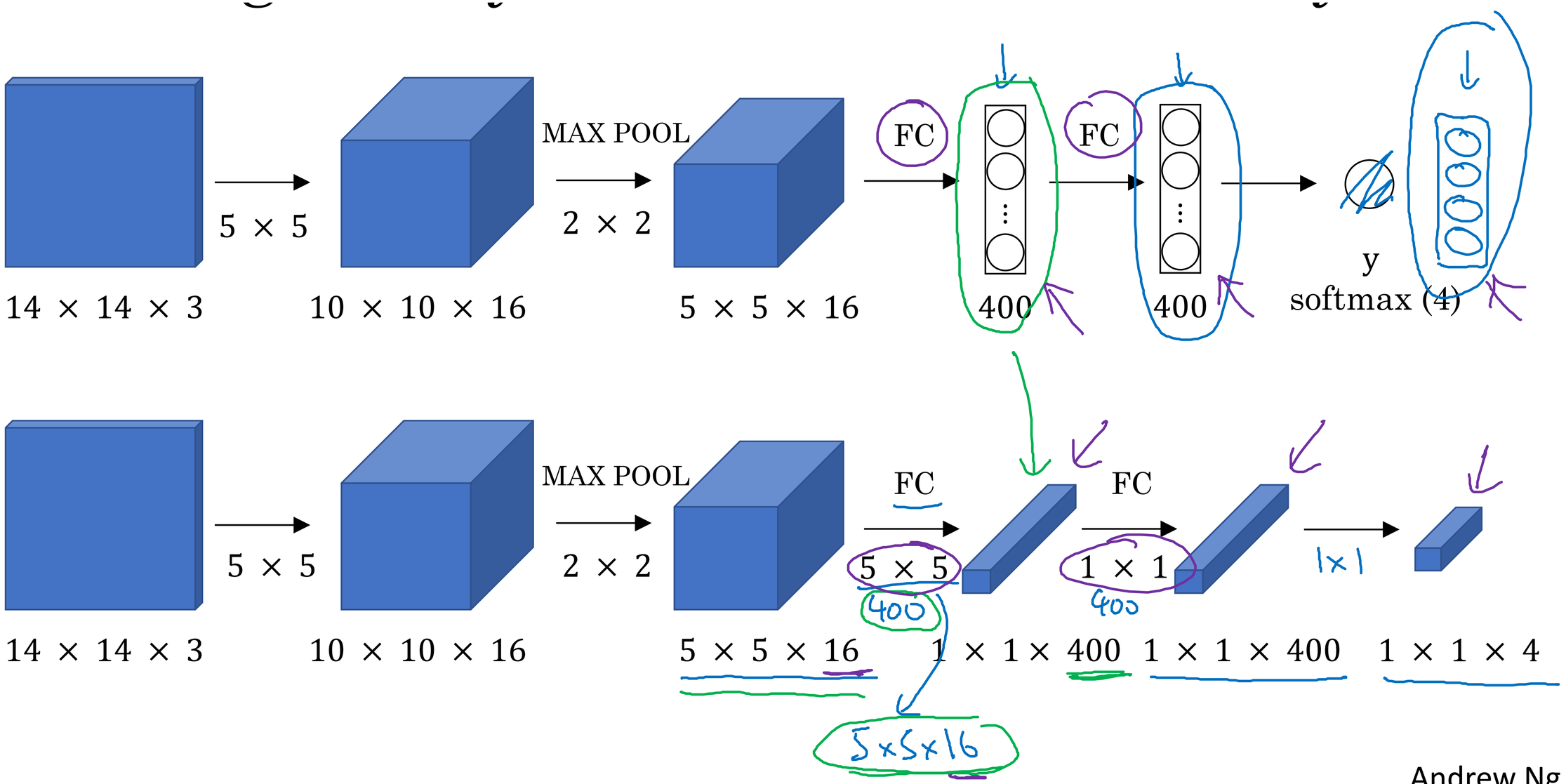
Very computationally expensive

But there is a way to ameliorate that

Turning FC layers into conv layers



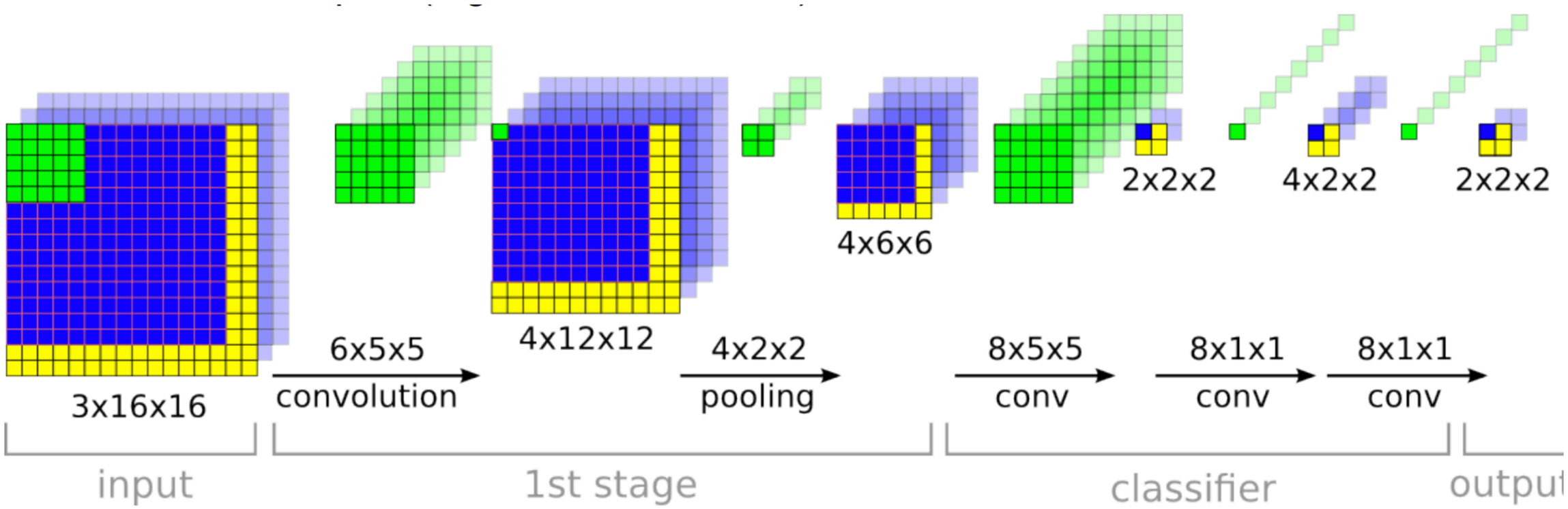
Turning FC layers into conv layers



Andrew Ng

Overfeat

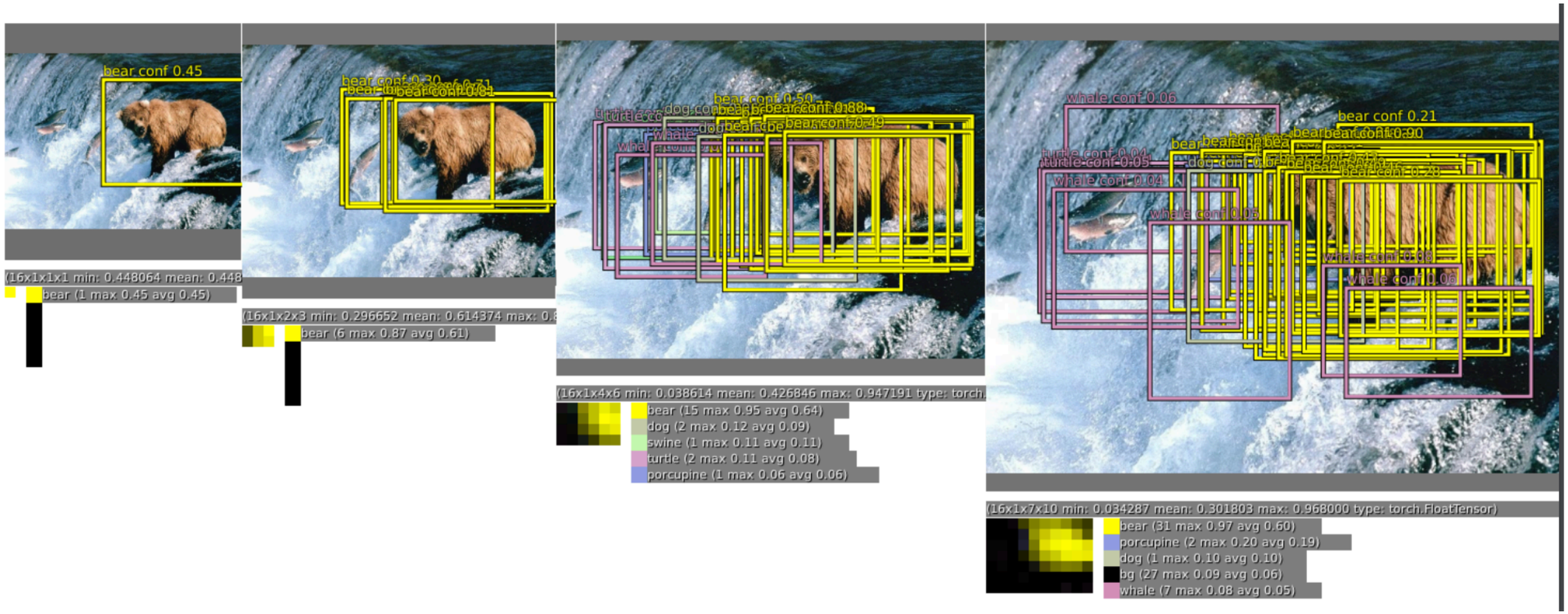
[Sermanet, 2013]



http://www.image-net.org/challenges/LSVRC/2013/slides/overfeat_ilsvrc2013.pdf

Overfeat

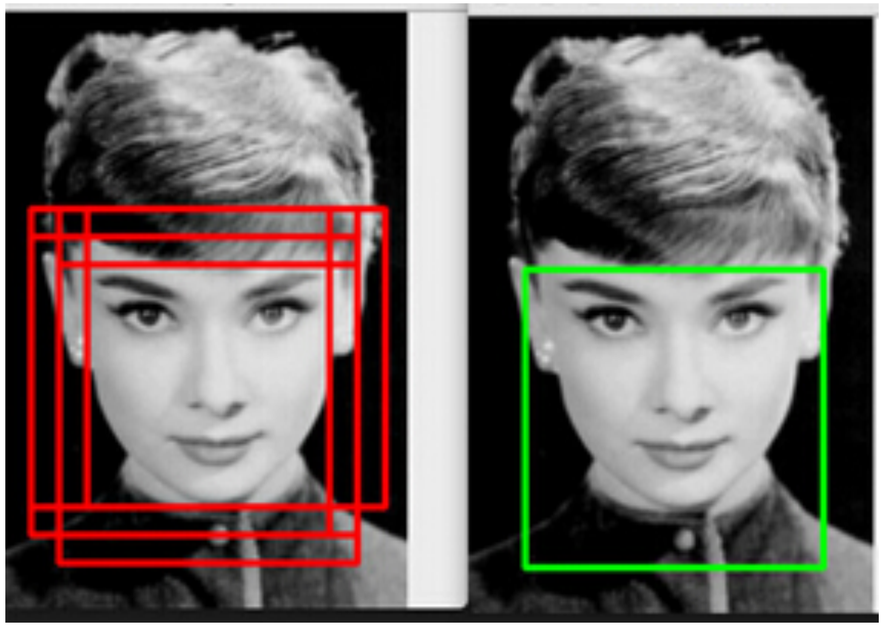
[Sermanet, 2013]

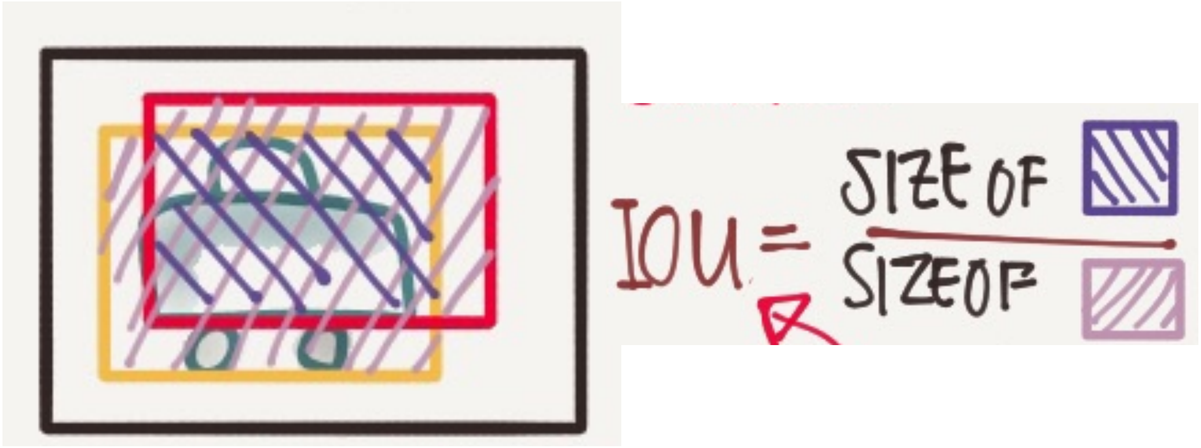


- For each location considered, predict class and bounding box
- Then count votes and unify via non-maximum suppression

http://www.image-net.org/challenges/LSVRC/2013/slides/overfeat_ilsvrc2013.pdf

NMS and IOU

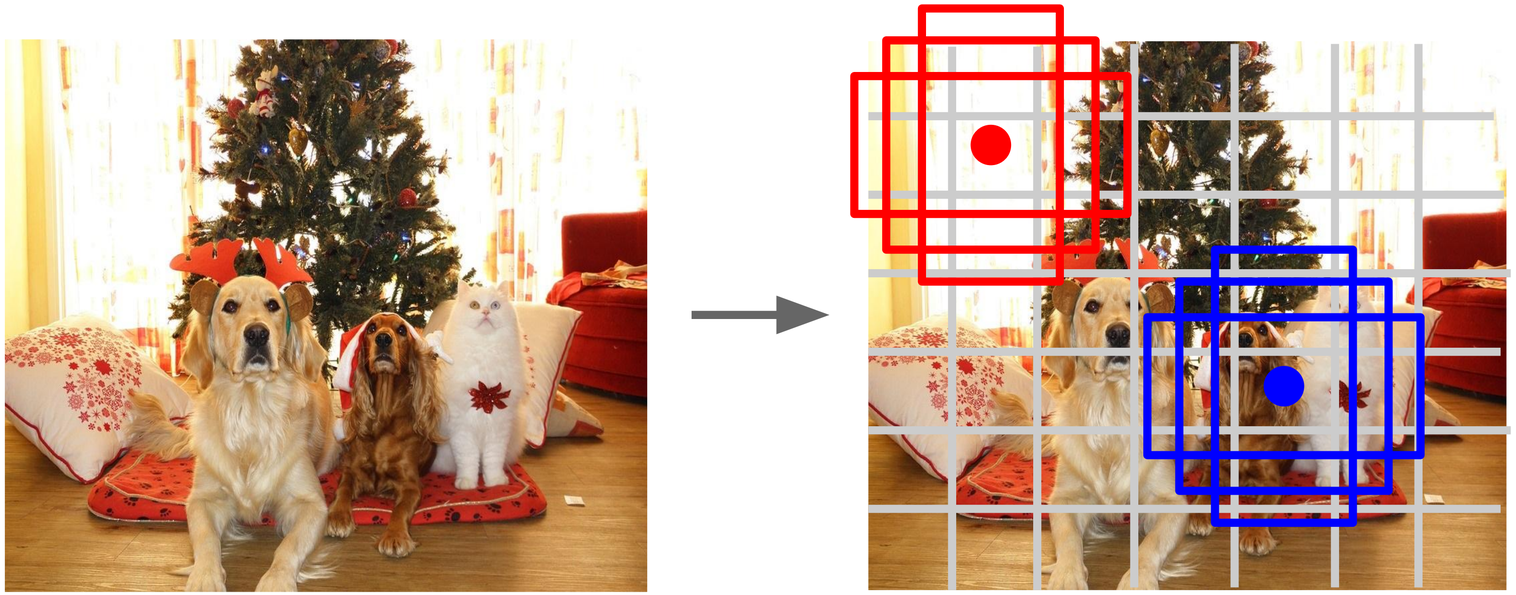




- Non-maximum Supression (NMS) means that when bounding boxes overlap significantly, only the one with the highest score should remain.
- Intersection over Union (IOU) is the most common metric for localization quality.

https://www.slideshare.net/TessFerrandez/notes-from-coursera-deep-learning-courses-by-andrew-ng
https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/single-shot-detectors/yolo.html

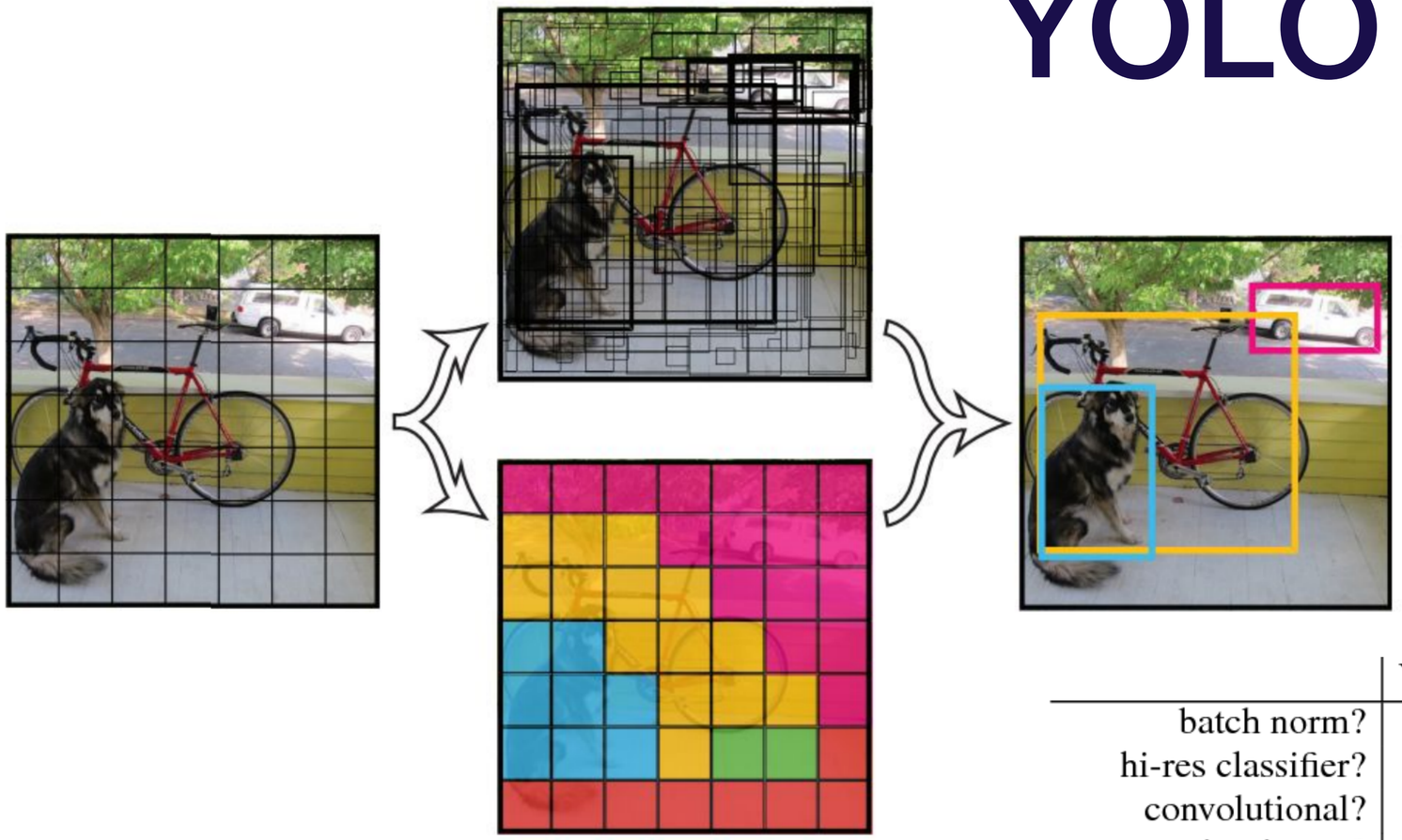
YOLO/SSD



- YOLO (You Only Look Once) and SSD (Single Shot Detector) scaled this approach up
- Each cell in a fixed grid predicts the presence of an object centerpoint, and has several “anchor boxes” of different aspect ratios
- Predict class and bounding box for each anchor box and cell, NMS predictions afterward

http://cs231n.stanford.edu/

YOLO



- YOLO is nice and fast, and is in active development
- Good off-the-shelf solution for many detection tasks

	YOLO								YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓	✓
anchor boxes?				✓	✓				
new network?					✓	✓	✓	✓	✓
dimension priors?						✓	✓	✓	✓
location prediction?						✓	✓	✓	✓
passthrough?							✓	✓	✓
multi-scale?								✓	✓
hi-res detector?									✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/single-shot-detectors/yolo.html

Dataset

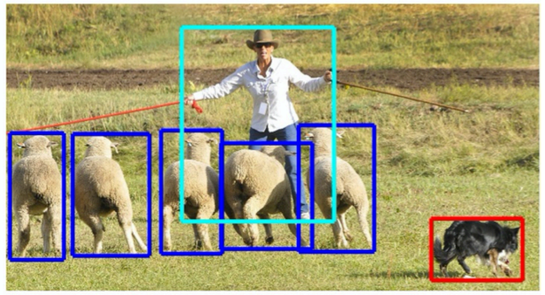
Microsoft COCO: Common Objects in Context

Tsung-Yi Lin Michael Maire Serge Belongie Lubomir Bourdev Ross Girshick
James Hays Pietro Perona Deva Ramanan C. Lawrence Zitnick Piotr Dollár

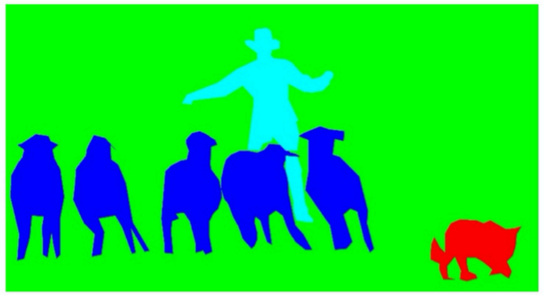
COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- **Object segmentation**
- **Recognition in context**
- **Superpixel stuff segmentation**
- **330K images (>200K labeled)**
- **1.5 million object instances**
- **80 object categories**
- **91 stuff categories**
- **5 captions per image**
- **250,000 people with keypoints**

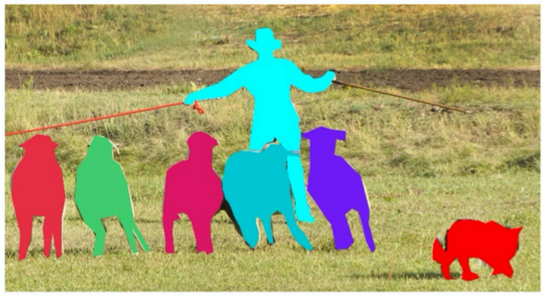
(a) Image classification



(b) Object localization



(c) Semantic segmentation



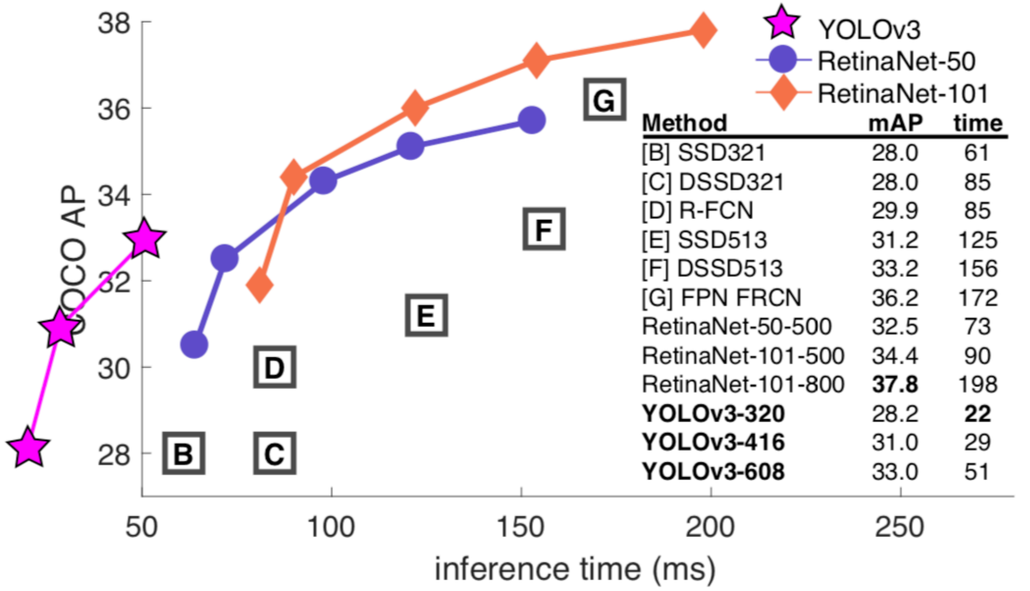
(d) This work

YOLOv3: An Incremental Improvement

Joseph Redmon, Ali Farhadi
University of Washington

Abstract

We present some updates to YOLO! We made a bunch of little design changes to make it better. We also trained this new network that's pretty swell. It's a little bigger than last time but more accurate. It's still fast though, don't worry. At 320 × 320 YOLOv3 runs in 22 ms at 28.2 mAP, as accurate as SSD but three times faster. When we look at the old .5 IOU mAP detection metric YOLOv3 is quite good. It achieves 57.9 AP_{50} in 51 ms on a Titan X, compared to 57.5 AP_{50} in 198 ms by RetinaNet, similar performance but 3.8× faster. As always, all the code is online at https://pjreddie.com/yolo/.



Method	mAP	time
[B] SSD321	28.0	61
[C] DSSD321	28.0	85
[D] R-FCN	29.9	85
[E] SSD513	31.2	125
[F] DSSD513	33.2	156
[G] FPN FRCN	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	**37.8**	198
YOLOv3-320	28.2	**22**
YOLOv3-416	31.0	29
YOLOv3-608	33.0	51

Figure 1. We adapt this figure from the Focal Loss paper [9]. YOLOv3 runs significantly faster than other detection methods with comparable performance. Times from either an M40 or Titan X, they are basically the same GPU.

1. Introduction

https://pjreddie.com/media/files/papers/YOLOv3.pdf

YOLOv4: Optimal Speed and Accuracy of Object Detection

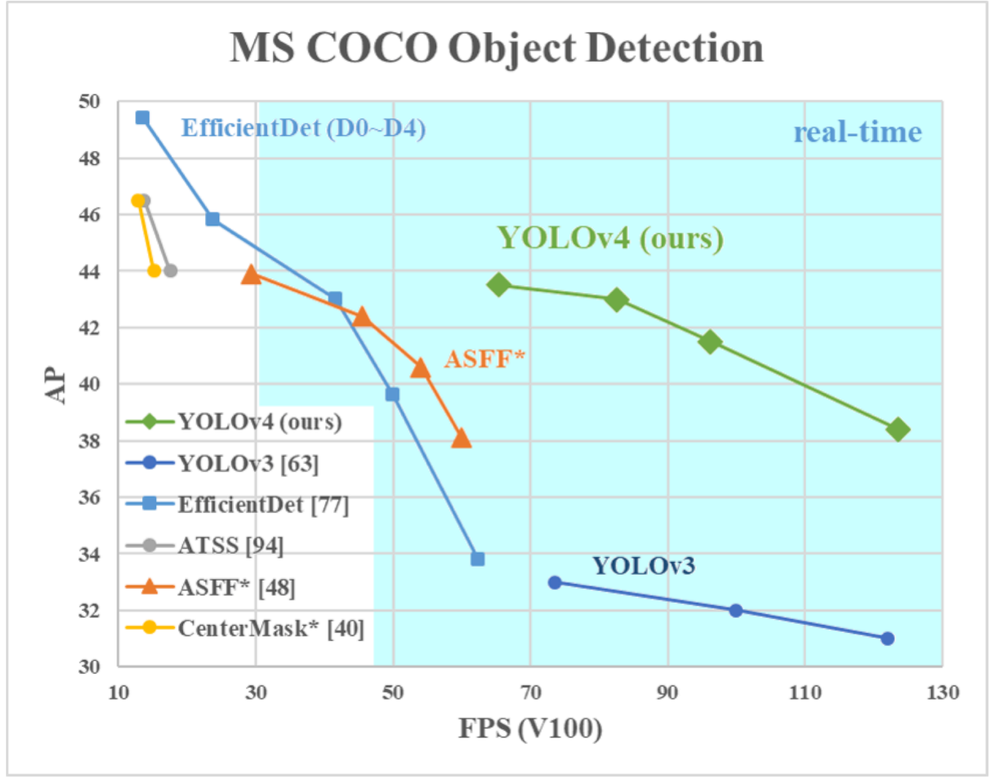
Alexey Bochkovskiy*
alexeyab84@gmail.com

Chien-Yao Wang*
Institute of Information Science
Academia Sinica, Taiwan
kinyiu@iis.sinica.edu.tw

Hong-Yuan Mark Liao
Institute of Information Science
Academia Sinica, Taiwan
liao@iis.sinica.edu.tw

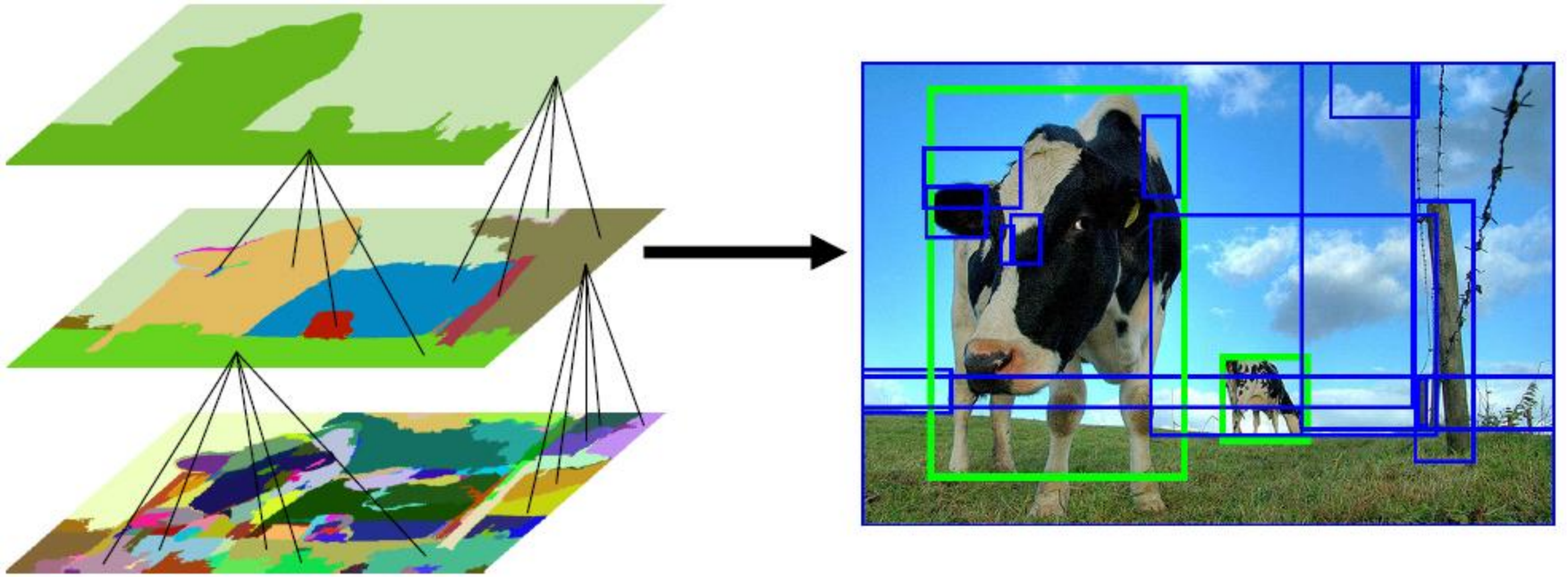
Abstract

There are a huge number of features which are said to improve Convolutional Neural Network (CNN) accuracy. Practical testing of combinations of such features on large datasets, and theoretical justification of the result, is required. Some features operate on certain models exclusively and for certain problems exclusively, or only for small-scale datasets; while some features, such as batch-normalization and residual-connections, are applicable to the majority of models, tasks, and datasets. We assume that such universal features include Weighted-Residual-Connections (WRC), Cross-Stage-Partial-connections (CSP), Cross mini-Batch Normalization (CmBN), Self-adversarial-training (SAT) and Mish-activation. We use new features: WRC, CSP, CmBN, SAT, Mish activation, Mosaic data augmentation



https://arxiv.org/pdf/2004.10934.pdf

Region proposal methods

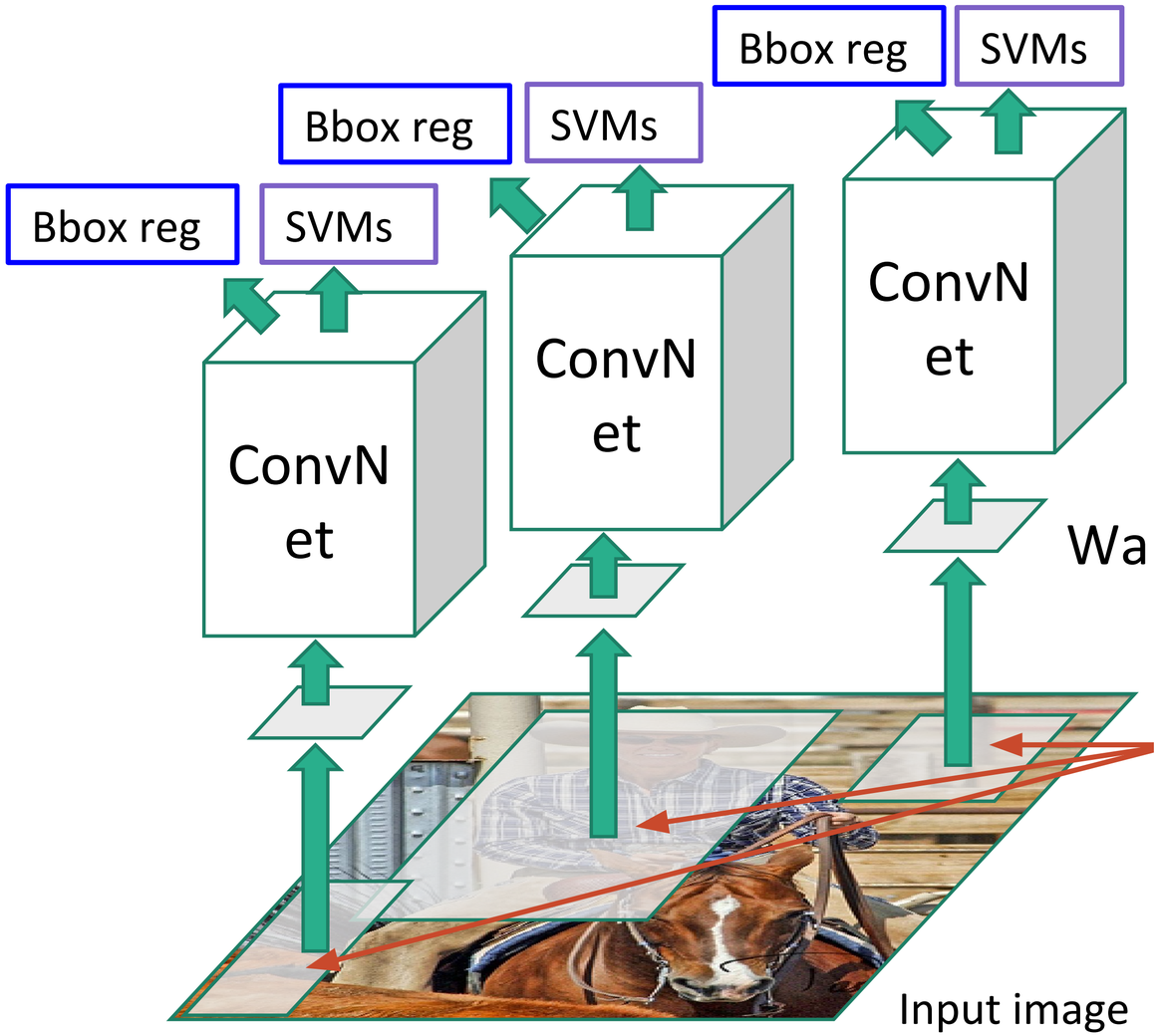


[Ujnings, 2013]

- There is an alternative to looking in every place in an image: look only at regions that seem interesting
- Different methods, most based on some kind of image segmentation

https://ivi.fnwi.uva.nl/isis/publications/bibtexbrowser.php?key=UijlingsIJCV2013

R-CNN [Girschick, 2014]



Predicting both class and bounding box

AlexNet on regions warped to square

Region proposals (slow external method)

- An interim solution, bridging old-school detection methods and convents

http://cs231n.stanford.edu/

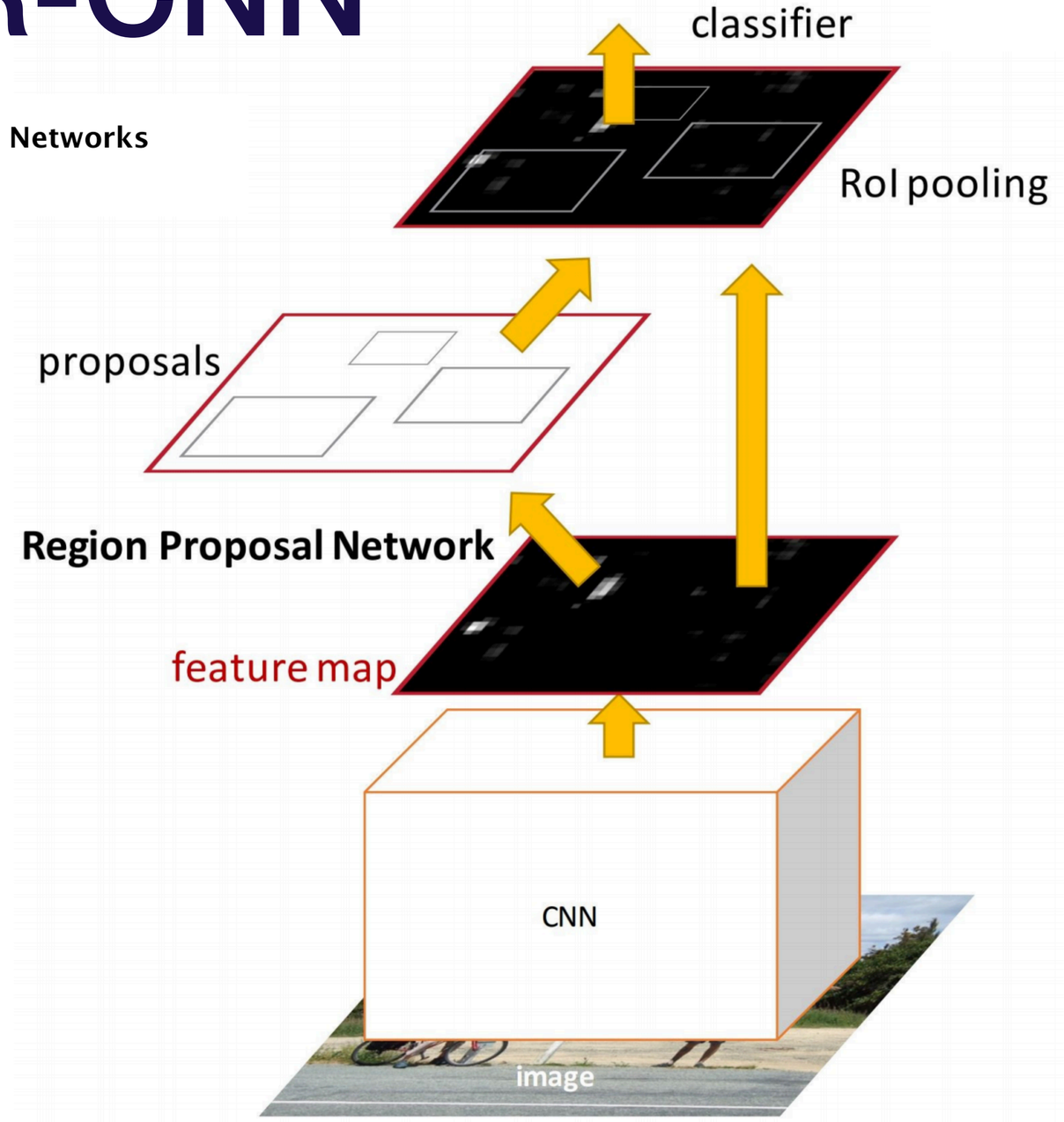
Faster R-CNN

[Submitted on 4 Jun 2015 (v1), last revised 6 Jan 2016 (this version, v3)]

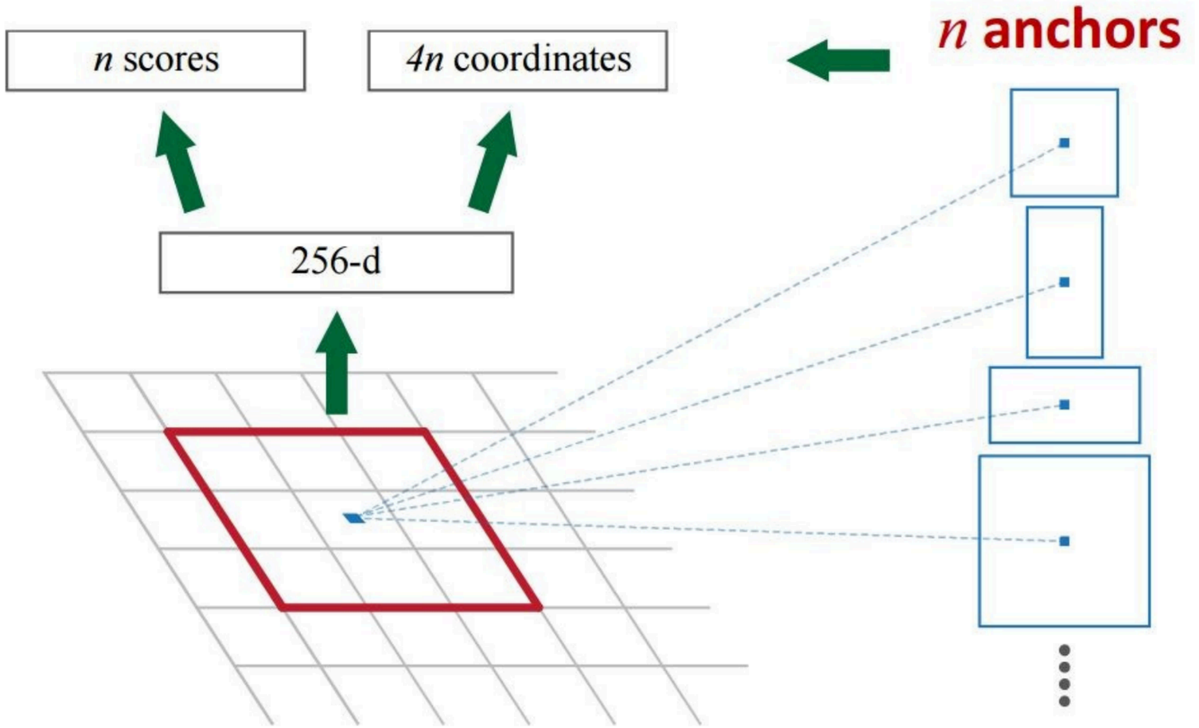
Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun

- Insert Region Proposal Network after the last convnet layer
- The proposals are then ROI-pooled, and classified
- Fast, because everything is in-network!

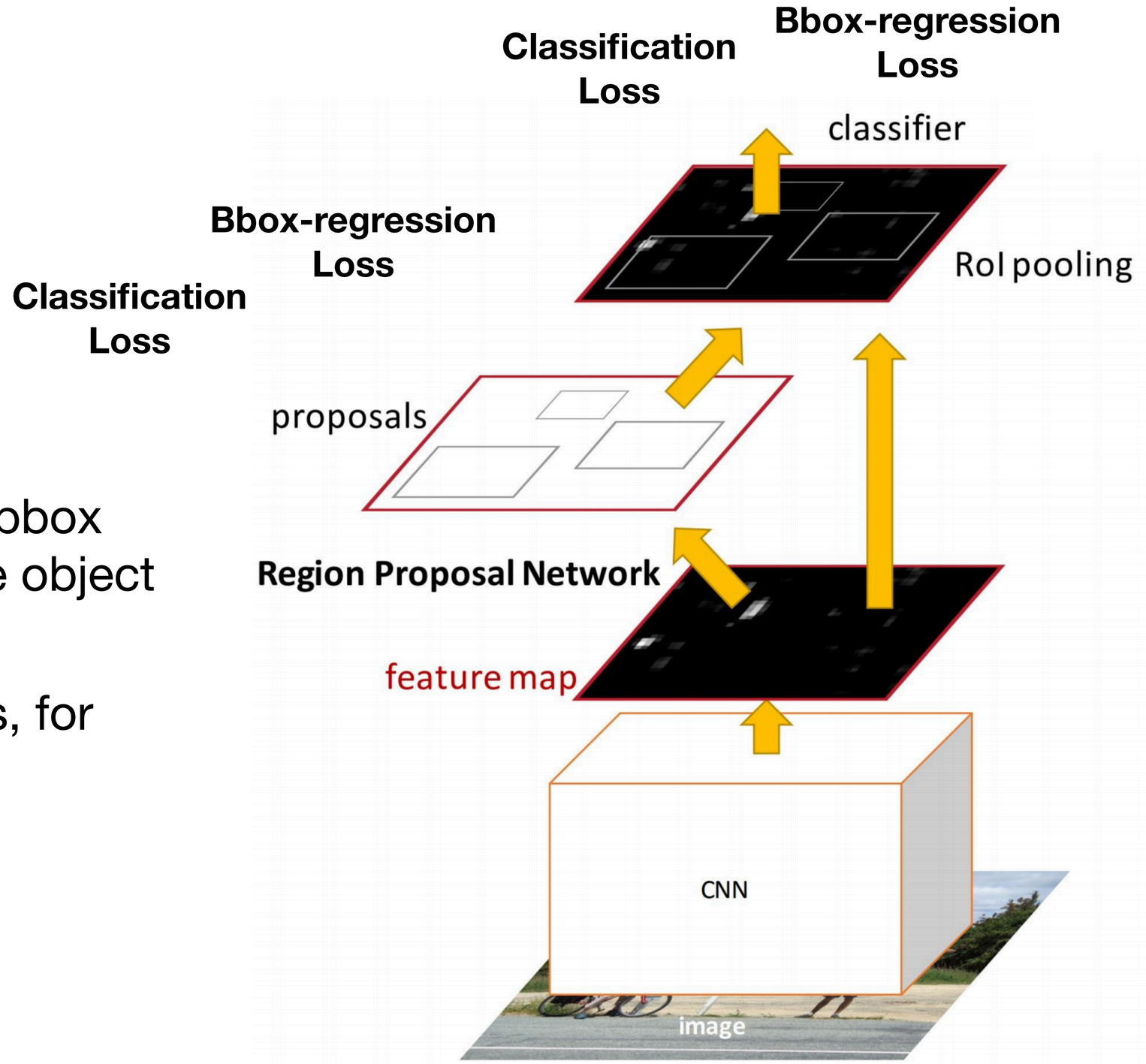


RPN



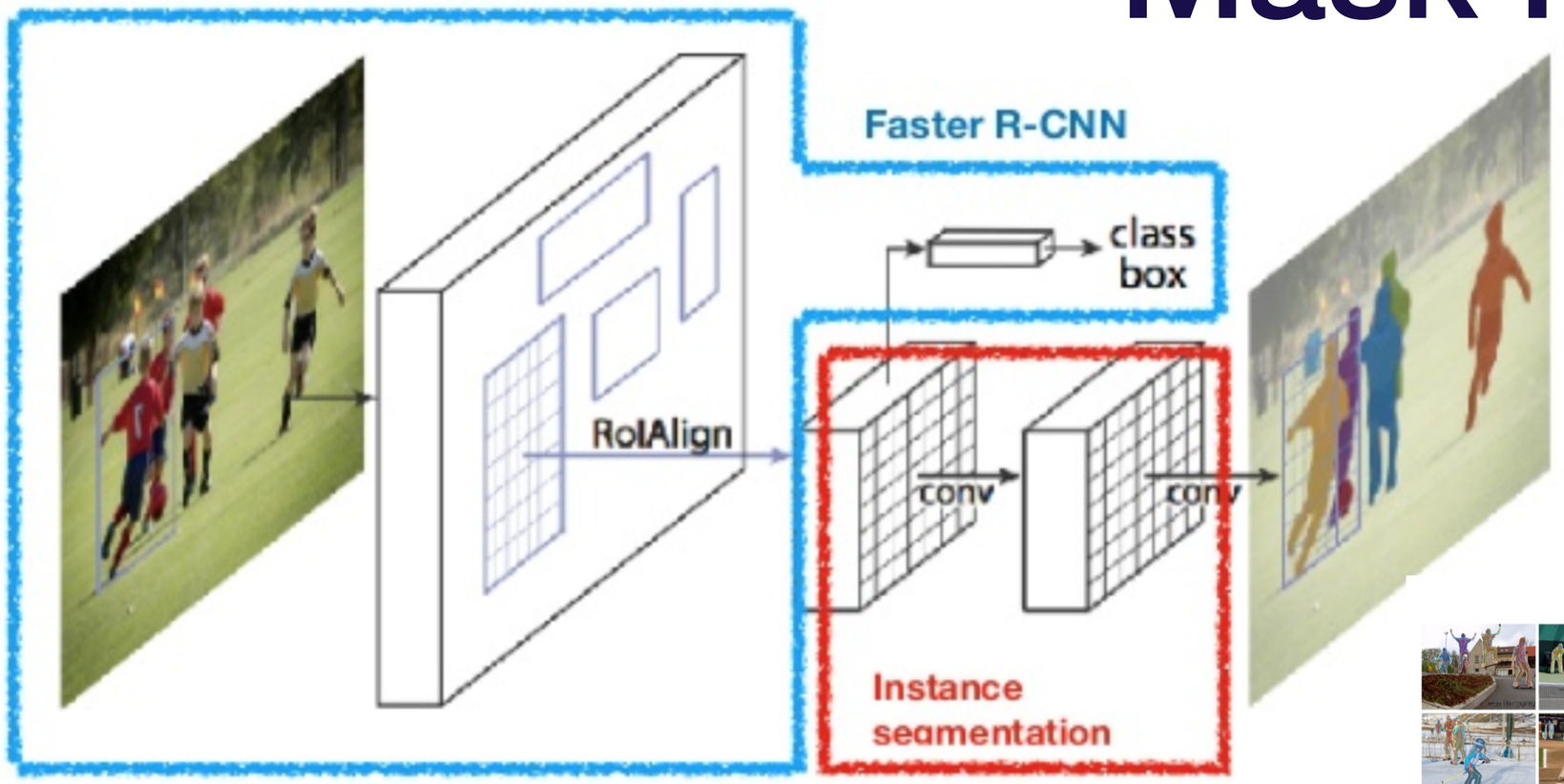
- Region Proposal Network is a fully convolutional method for scoring a bunch of candidate windows for “objectness”
- At each location, it considers multiple canonical anchor boxes, and also predicts bounding box offset from each anchor (like YOLO)

Faster R-CNN Training



- Four losses total: classifier and bbox regression for both RPN and the object classifier
- …what if we added another loss, for segmentation?

Mask R-CNN



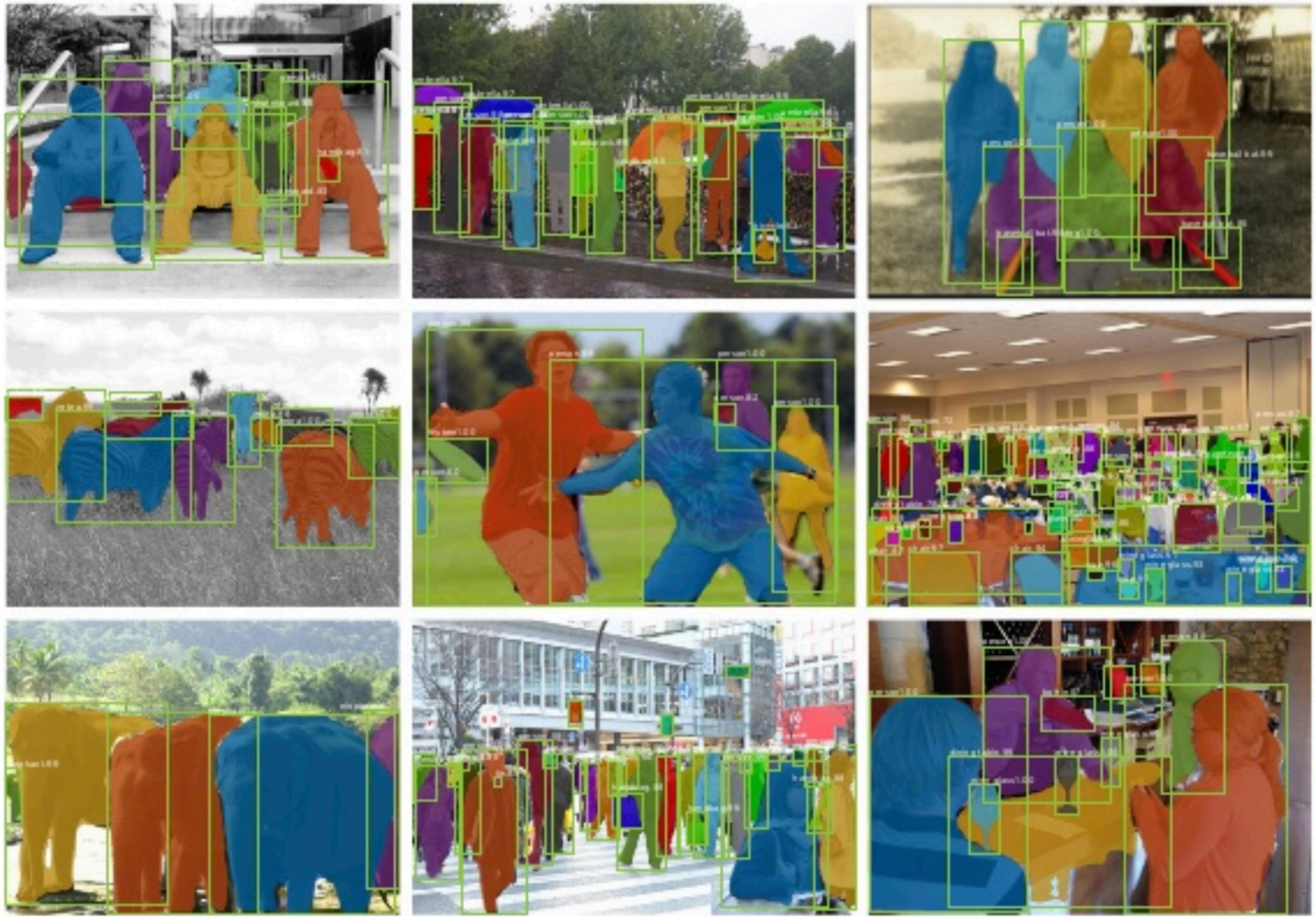
Mask R-CNN
Kaiming He, **Georgia Gkioxari**, Piotr Dollàr and Ross Girshick
International Conference of Computer Vision (ICCV), 2017 **(oral)**
Best Paper Award (Marr Prize)

arxiv / code / bibtex

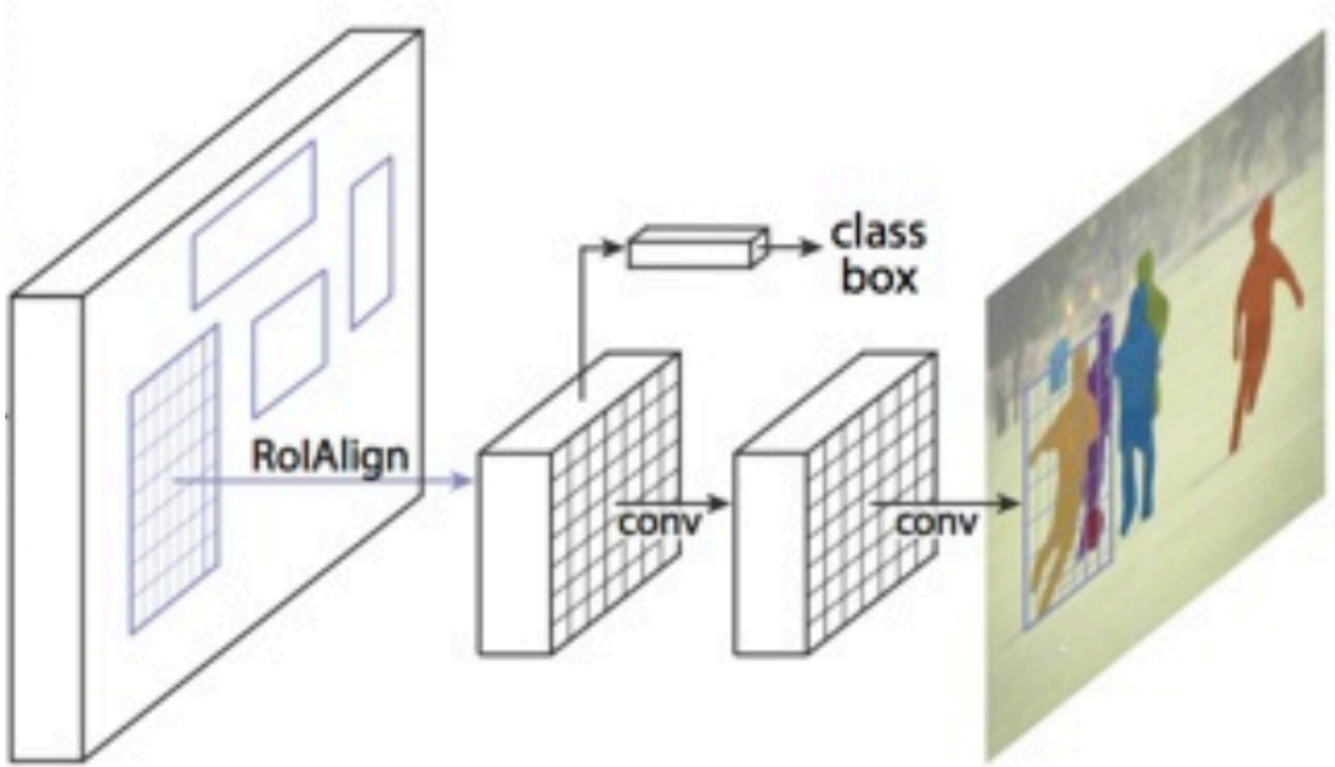
- ROIs are not only classified and bbox-regressed, but also go through an instance segmentation module
- RoiAlign is needed to preserve exact locations (10-50% improvement)

https://github.com/wannabeOG/Mask-RCNN

Mask R-CNN

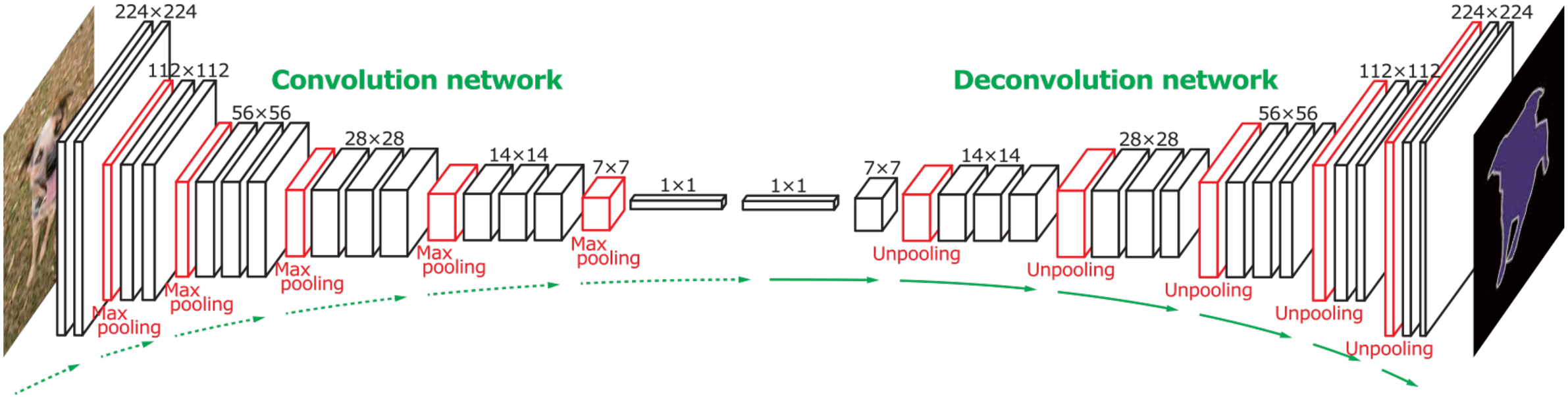


Mask R-CNN



- Each region is masked with just a couple of non-downsampling convolutions
- What if we wanted to apply that to the whole image, not just RoI's?

Fully Convolutional Nets

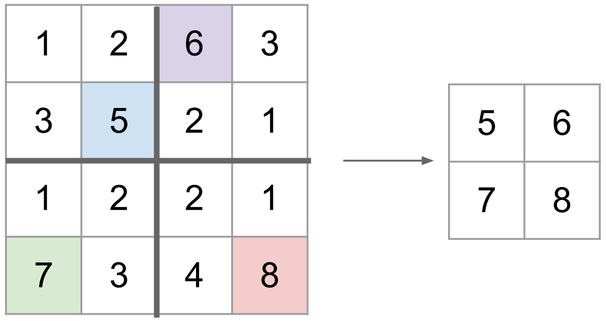


- Convolutions at the original scale get expensive
- Encode/Decode: shrink down, then upsample
- But how to upsample?

Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015

Upsampling



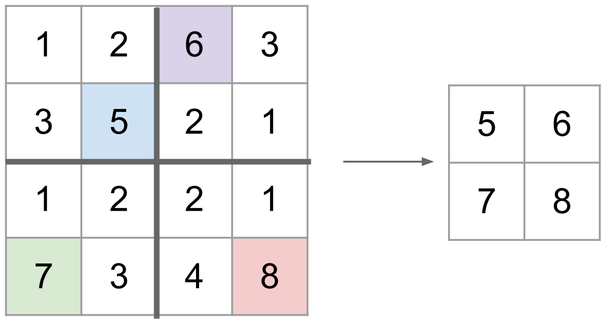
Unpooling

- **Unpooling**: Remember which pixels max-pooling selected, and use the same when upsampling

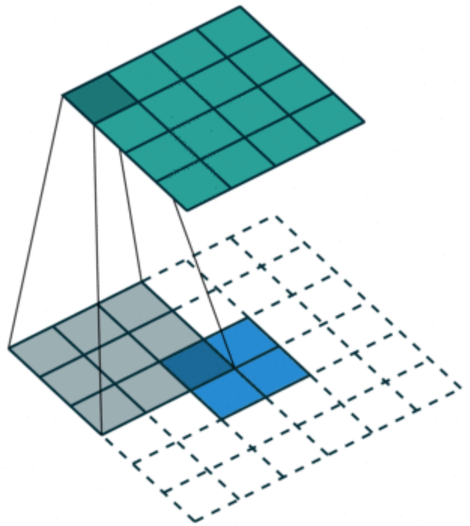
Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015

Upsampling



Unpooling



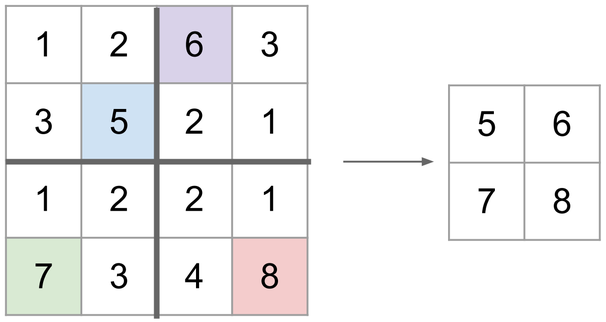
Transpose Convolutions

- **Unpooling**: Remember which pixels max-pooling selected, and use the same when upsampling
- **Transpose convolutions**: learnable upsampling

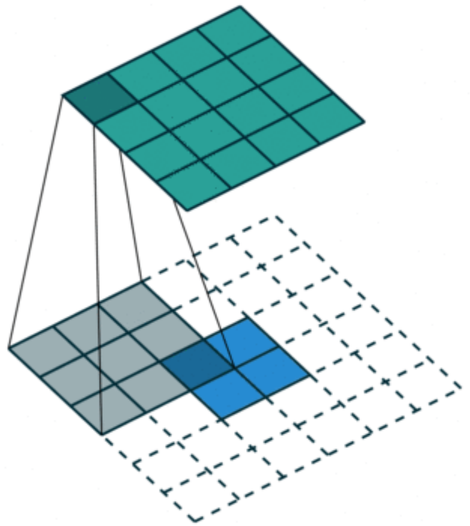
Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015

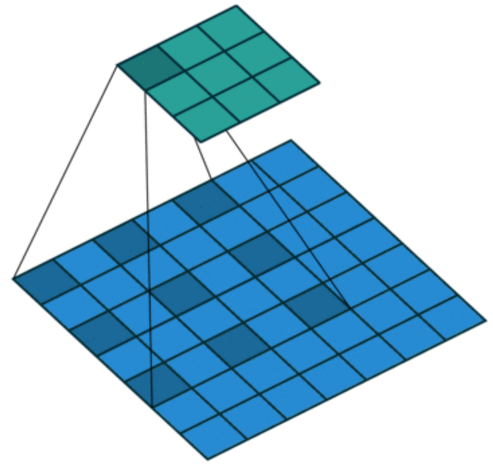
Upsampling



Unpooling



Transpose Convolutions



Dilated Convolutions

- **Unpooling**: Remember which pixels max-pooling selected, and use the same when upsampling
- **Transpose convolutions**: learnable upsampling
- + **Dilated convolutions**: lose less information as you downsample

Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015

Agenda

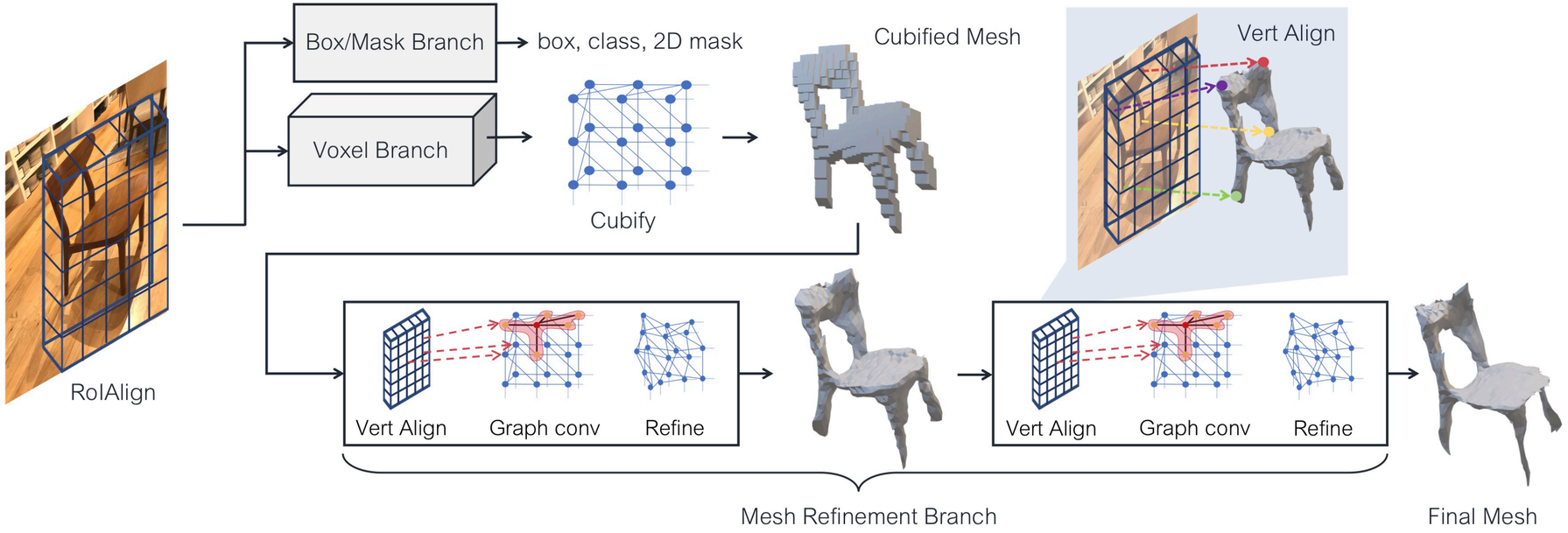
1. Tour of Convnet Architectures
2. Localization, Detection, Segmentation
3. **Misc**

3D Shape Inference

Mesh R-CNN

Georgia Gkioxari Jitendra Malik Justin Johnson

Facebook AI Research



- Given a 2D image, detect objects and predict their 3D meshes.
- Need labeled data!

ShapeNet is an ongoing effort to establish a richly-annotated, large-scale dataset of 3D shapes. We provide researchers around the world with this data to enable research in computer graphics, computer vision, robotics, and other related disciplines. ShapeNet is a collaborative effort between researchers at Princeton, Stanford and TTIC.

Browse Taxonomy | Search Models

Overview

ShapeNet consists of several subsets:

ShapeNetCore

ShapeNetCore is a subset of the full ShapeNet dataset with single clean 3D models and manually verified category and alignment annotations. It covers 55 common object categories with about 51,300 unique 3D models. The 12 object categories of PASCAL 3D+, a popular computer vision 3D benchmark dataset, are all covered by ShapeNetCore.

ShapeNetSem

ShapeNetSem is a smaller, more densely annotated subset consisting of 12,000 models spread over a broader set of 270 categories. In addition to manually verified category labels and consistent alignments, these models are annotated with real-world dimensions, estimates of their material composition at the category level, and estimates of their total volume and weight.

News

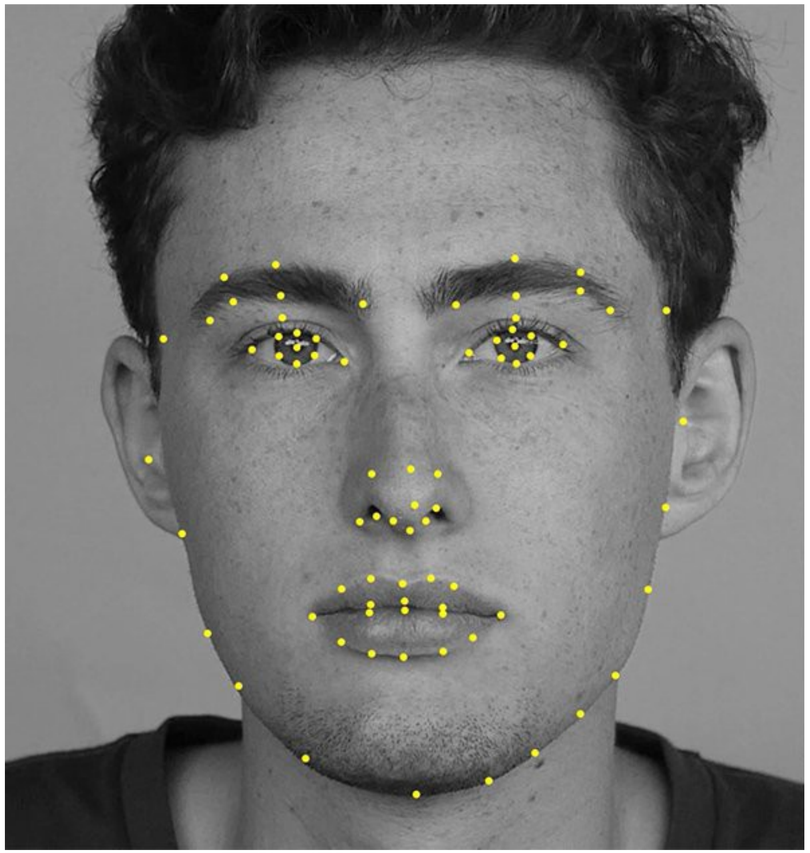
January, 2020 We are happy to announce the release of ShapeGlot. ShapeGlot provides contrastive referential language for shapes of ShapeNet.

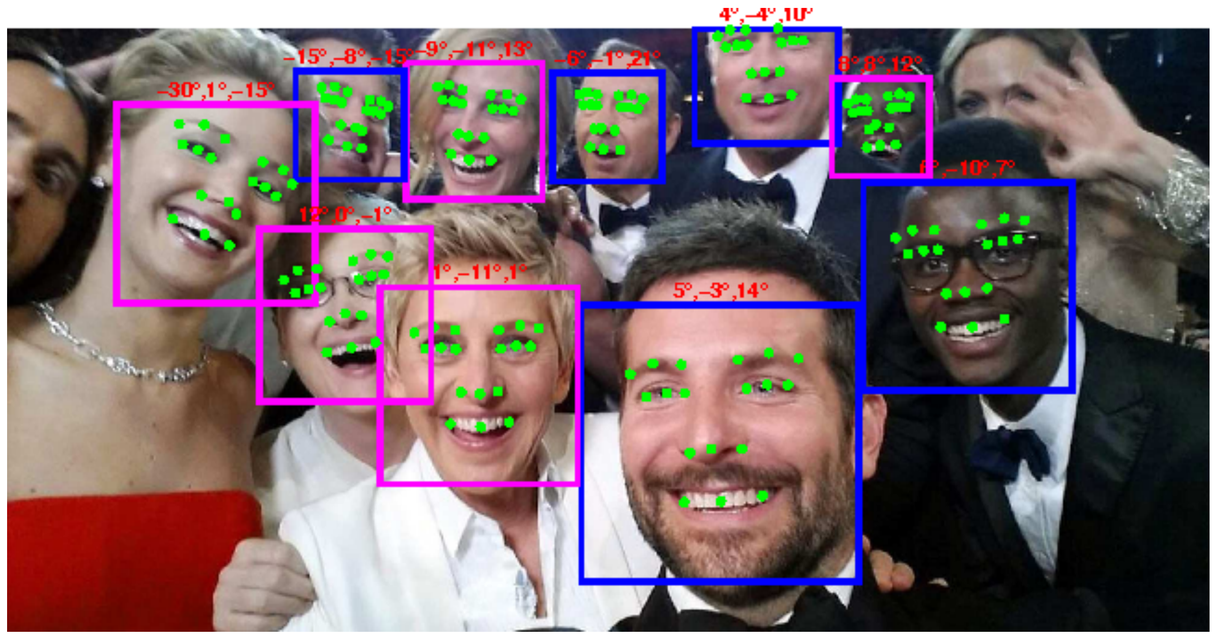
March, 2019 We are happy to announce the prelease of PartNet v0. PartNet provides fine-grained, hierarchical part annotations from ShapeNet..

Aug, 2017 We are organizing a ShapeNet challenge at ICCV 2017. More information available at https://shapenet.cs.stanford.edu/iccv17/.

https://www.shapenet.org

Face Landmark Recognition



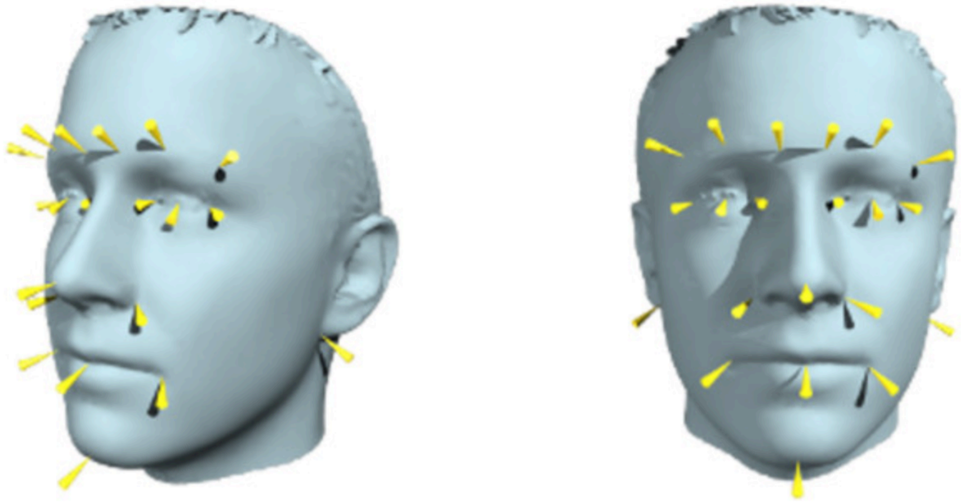


- Given an image, predict landmarks, such as face keypoints.
- Need labeled data!

Annotated Faces in the Wild

Annotated Facial Landmarks in the Wild (AFLW)

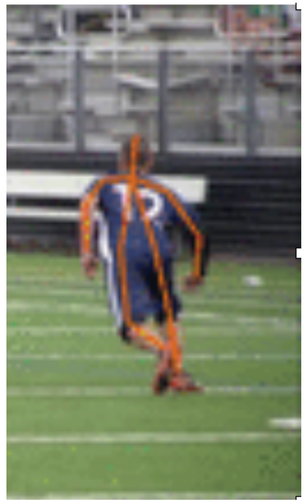
Annotated Facial Landmarks in the Wild (AFLW) provides a large-scale collection of annotated face images gathered from the web, exhibiting a large variety in appearance (e.g., pose, expression, ethnicity, age, gender) as well as general imaging and environmental conditions. In total about **25k faces** are annotated with up to **21 landmarks** per image.



To Top

https://www.tugraz.at/institute/icg/research/team-bischof/lrs/downloads/aflw/

Pose Estimation



- Given an image, predict landmarks, such as joint locations.
- Need labeled data!

MS COCO Dataset

COCO Common Objects in Context

info@cocodataset.org

Home People Dataset Tasks Evaluate

News

- We are pleased to announce the COCO 2020 **Detection**, **Keypoint**, **Panoptic**, and **DensePose** Challenges.
- **The new rules and awards for this year challenges encourage innovative methods.**
- Results to be announced at the **Joint COCO and LVIS Recognition** ECCV workshop.

What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- **Object segmentation**
- **Recognition in context**
- **Superpixel stuff segmentation**
- **330K images (>200K labeled)**
- **1.5 million object instances**
- **80 object categories**
- **91 stuff categories**
- **5 captions per image**
- **250,000 people with keypoints**

Collaborators

Tsung-Yi Lin Google Brain

Genevieve Patterson MSR, Trash TV

Matteo R. Ronchi Caltech

Yin Cui Google

Michael Maire TTI-Chicago

Serge Belongie Cornell Tech

Lubomir Bourdev WaveOne, Inc.

Ross Girshick FAIR

James Hays Georgia Tech

Pietro Perona Caltech

Deva Ramanan CMU

Larry Zitnick FAIR

Piotr Dollár FAIR

Sponsors

CVDF

Microsoft

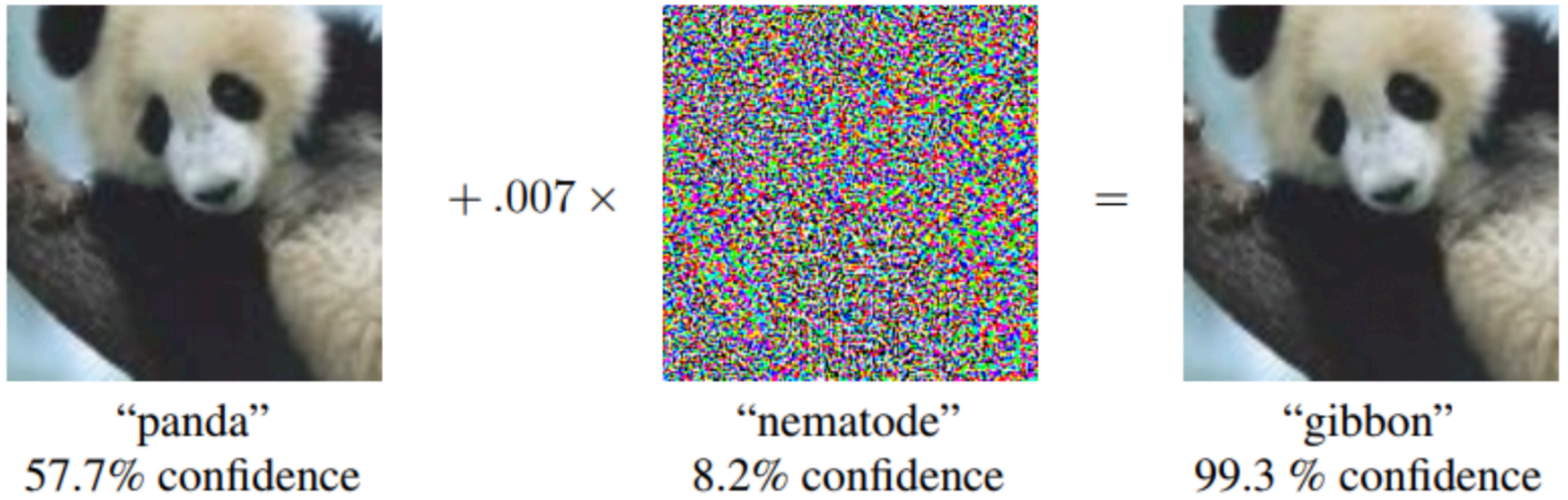
facebook

Mighty AI

https://cocodataset.org/#home

Basically, any type of vision task where there is a lot of labeled training data is a good match.

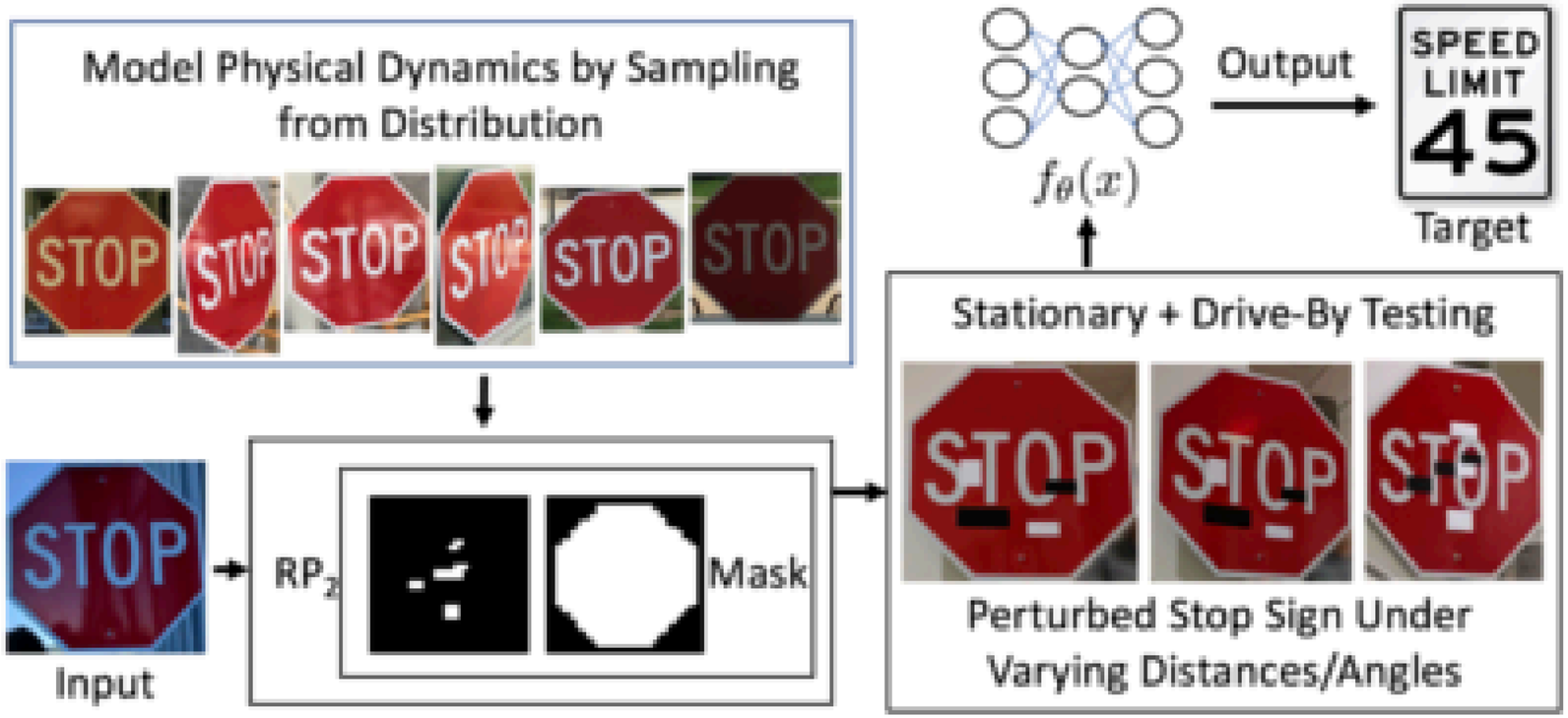
Adversarial Attacks



Example attack from Explaining and Harnessing Adversarial Examples.

- Convnets are powerful, but brittle in surprising ways.
- *White-box* attacks have access to model parameters; *black-box* don’t.
- Intuition is that neural nets learned to predict the training data manifold, and are not effective when pushed off of it.

Adversarial Attacks

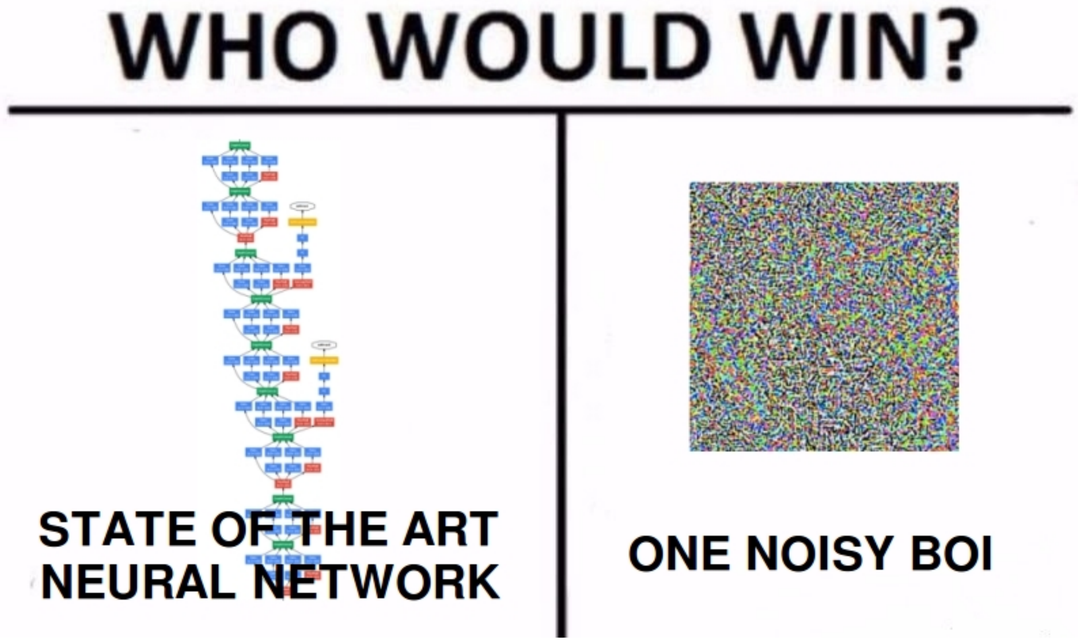


- Can even work printed out, under a variety of viewing angles

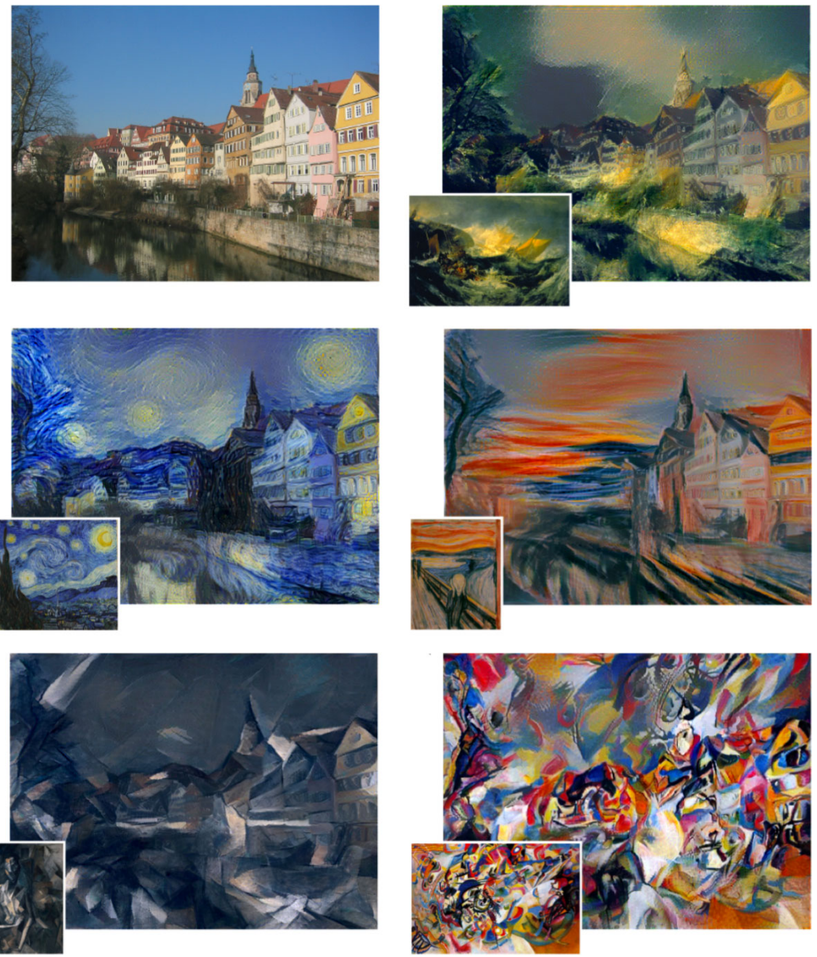
Robust Physical-World Attacks on Deep Learning Visual Classification Kevin Eykholt[*1], Ivan Evtimov[*2], Earlence Fernandes[2], Bo Li[3], Amir Rahmati[4], Chaowei Xiao[1], Atul Prakash[1], Tadayoshi Kohno[2], and Dawn Song[3] CVPR 2018

Adversarial Attacks

- Attacking
  - Some form of modifying the input in the direction of the loss gradient
- Defending
  - Train with adversarial examples in the set (doesn’t work well)
  - Smooth the class decision boundaries
    - “Defensive distillation”



Style Transfer



Style target

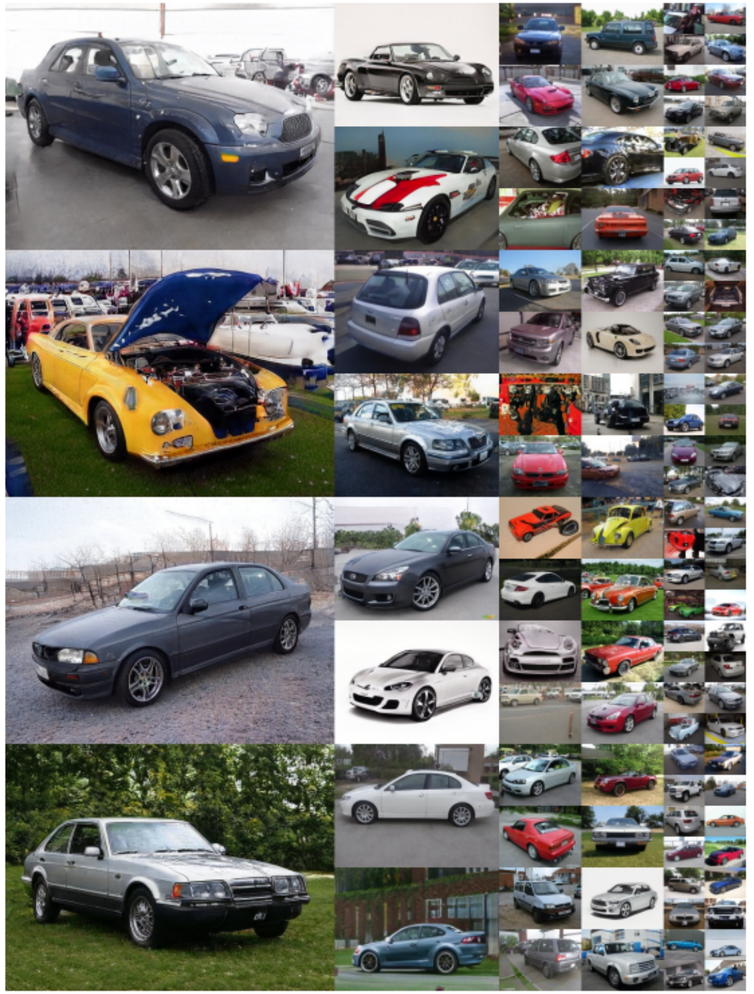
A Neural Algorithm of Artistic Style by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge.

@eliotandres

https://ndres.me/post/machine-learning-with-gifs-style-transfer/

GANs

- Will be covered in Research Directions lecture!



Learn More

- https://www.paperswithcode.com/area/computer-vision