Practical Deep learning for Computer Vision

Dr Kien Nguyen
Email: k.nguyenthanh@qut.edu.au
Queensland University of Technology, Australia

Warning

Deep learning is new SEXY (for a reason!)

Week 3: Object Recognition

- Resnet from scratch
- Transfer Learning
- Google Colab

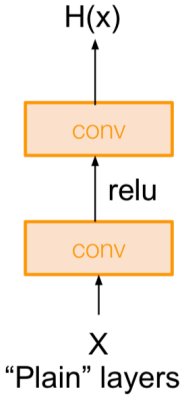
Week 4: Object Detection

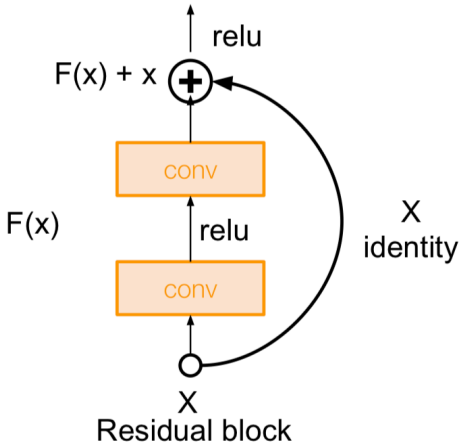
- DarkNet & YOLO
- Transfer Learning
- AWS

Week 5: GAN

- SSD
- GAN

Week 3. Resnet from scratch





IMAGENET

www.image-net.org

22K categories and **14M** images

- Animals
  - Bird
  - Fish
  - Mammal
  - Invertebrate
- Plants
  - Tree
  - Flower
- Food
- Materials
- Structures
- Artifact
  - Tools
  - Appliances
  - Structures
- Person
- Scenes
  - Indoor
  - Geological Formations
- Sport Activities

Deng, Dong, Socher, Li, Li, & Fei-Fei, 2009

IMAGENET Large Scale Visual Recognition Challenge

Steel drum

The Image Classification Challenge:

1,000 object classes

1,431,167 images

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

✔

Output:
Scale
T-shirt
Giant panda
Drumstick
Mud turtle

✗

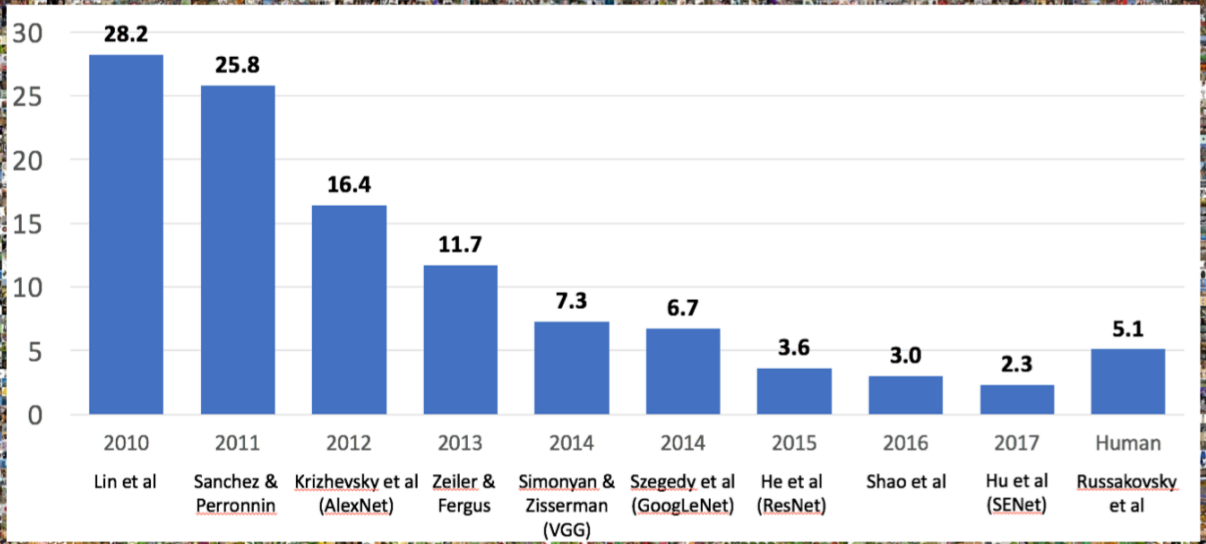
Russakovsky et al. arXiv, 2014

IMAGENET Large Scale Visual Recognition Challenge

The Image Classification Challenge:

1,000 object classes

1,431,167 images



Russakovsky et al. IJCV 2015

IMAGENET Large Scale Visual Recognition Challenge

Year 2010

NEC-UIUC

Dense descriptor grid: HOG, LBP

Coding: local coordinate, super-vector

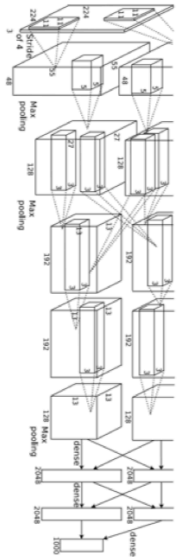
Pooling, SPM

Linear SVM

[Lin CVPR 2011]

Year 2012

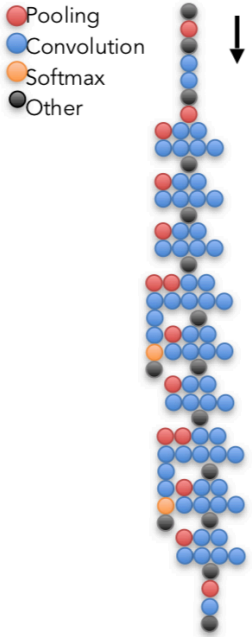
SuperVision



[Krizhevsky NIPS 2012]

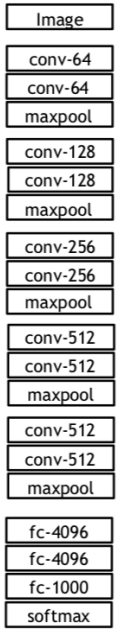
Year 2014

GoogLeNet



[Szegedy arxiv 2014]

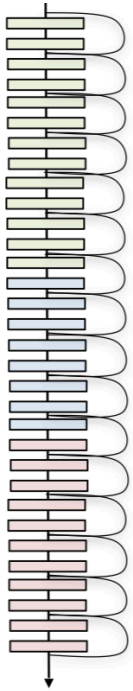
VGG



[Simonyan arxiv 2014]

Year 2015

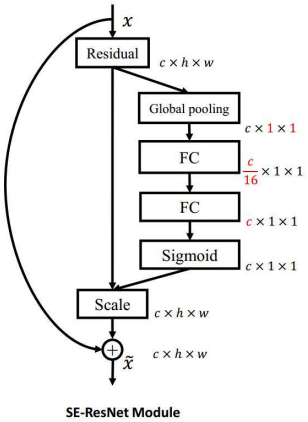
ResNet



[He ICCV 2015]

Year 2017

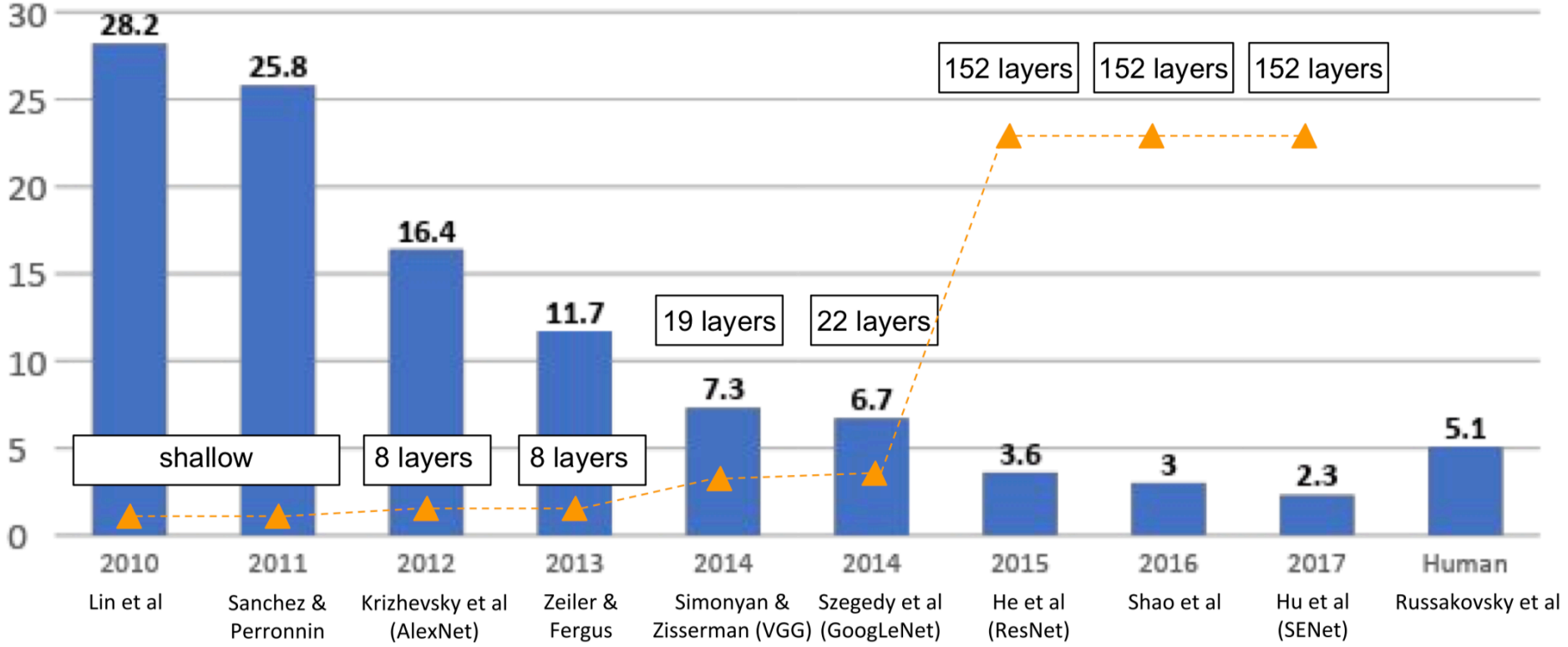
SENet

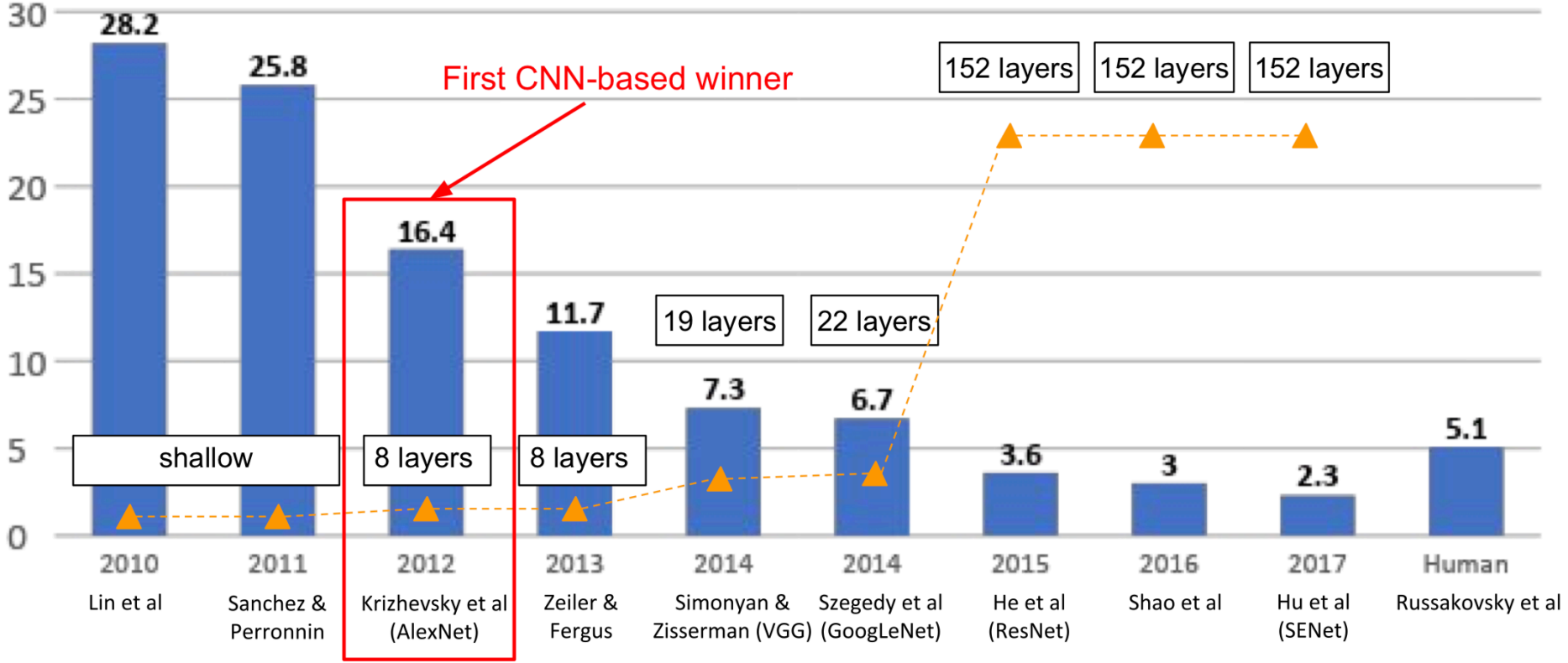


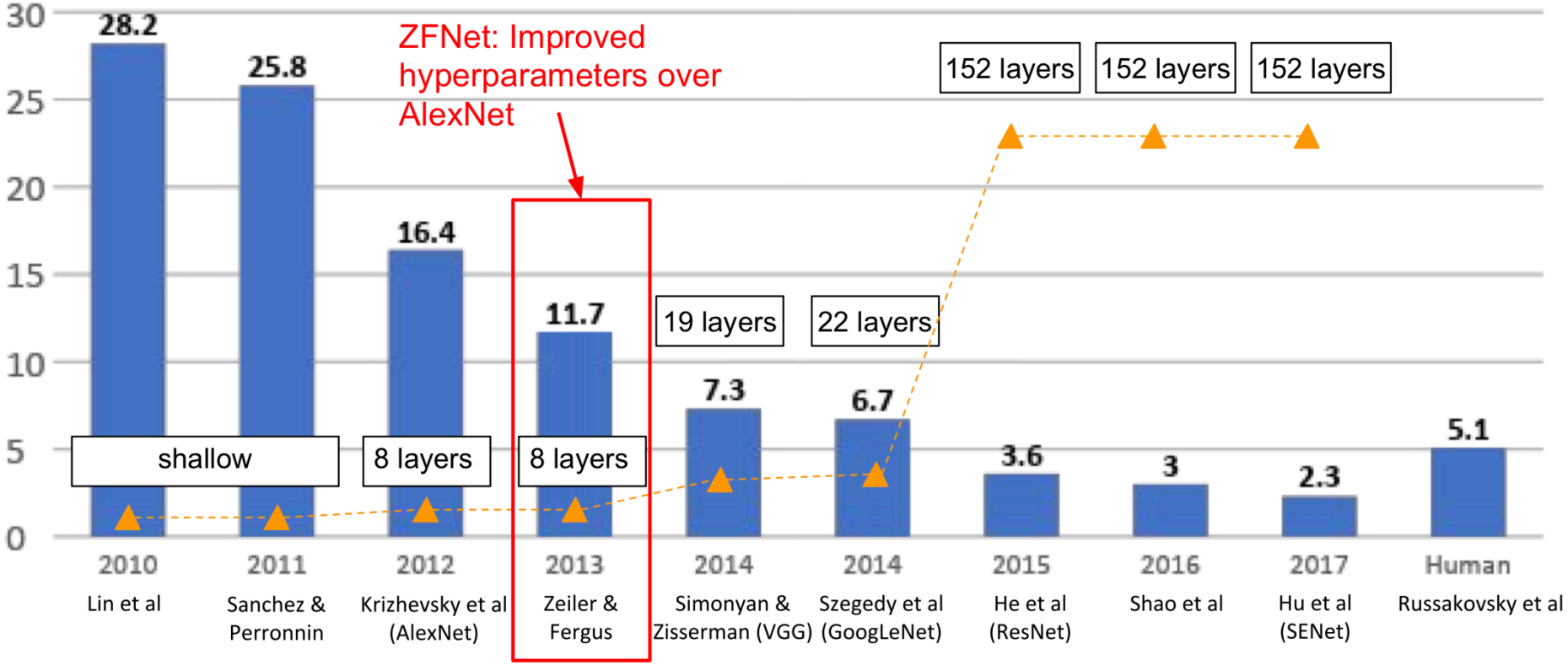
SE-ResNet Module

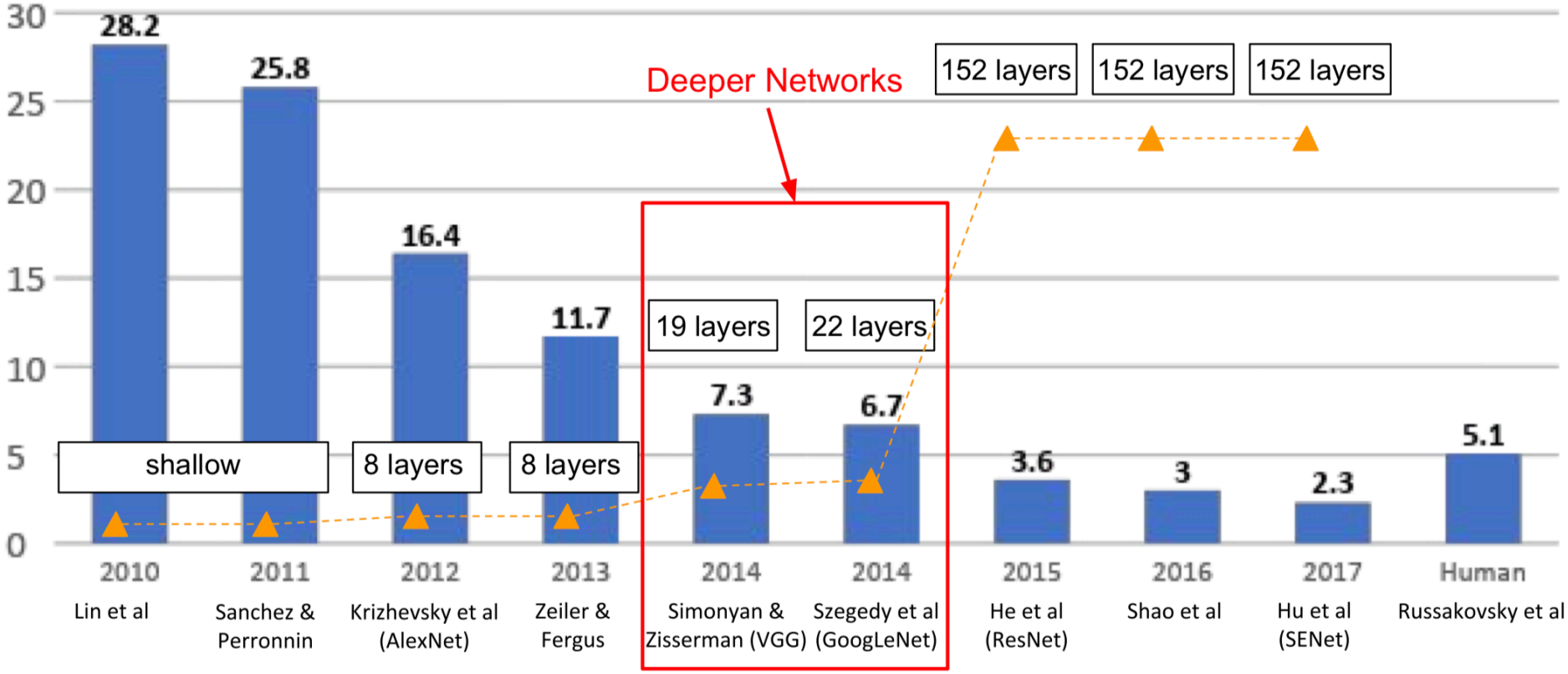
Hu CVPR 2018

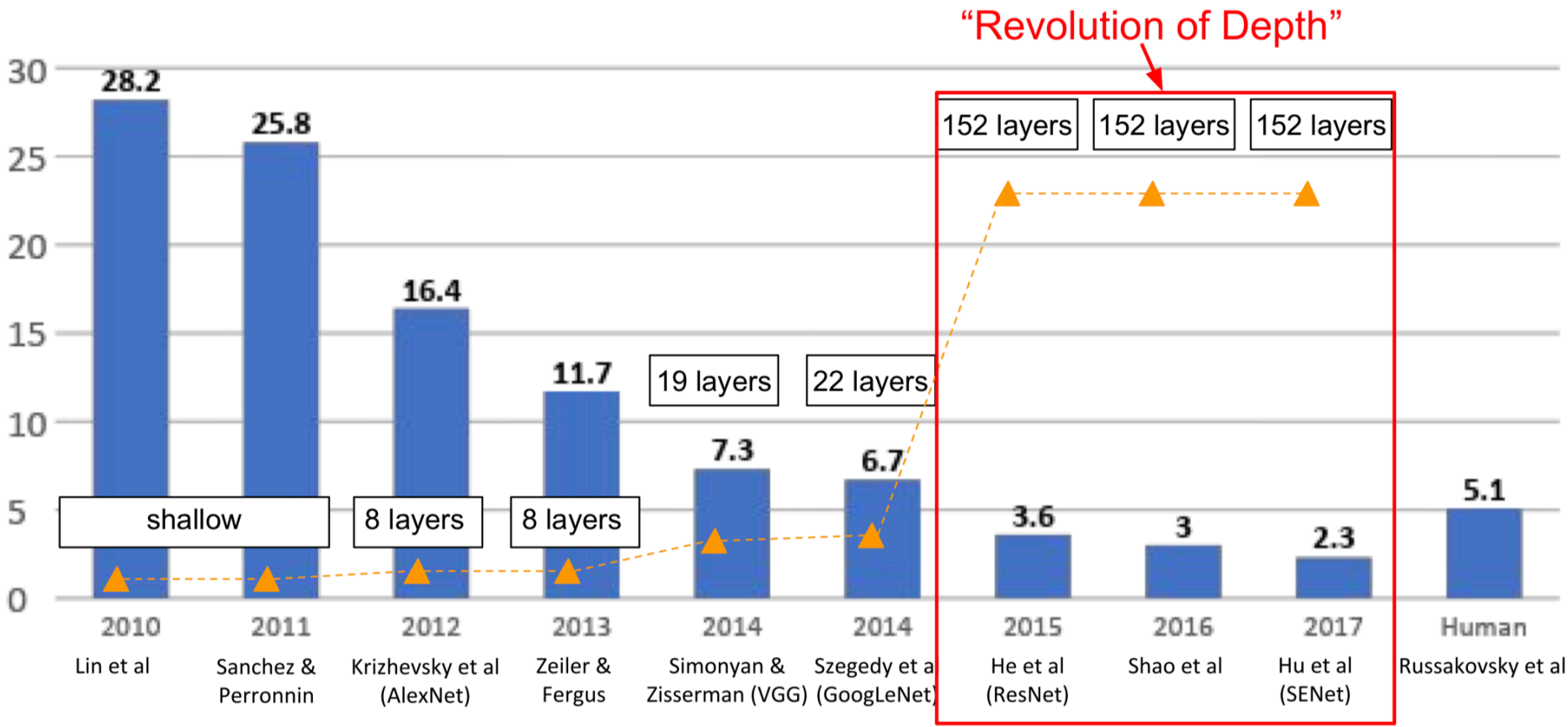
Revolution of Depth



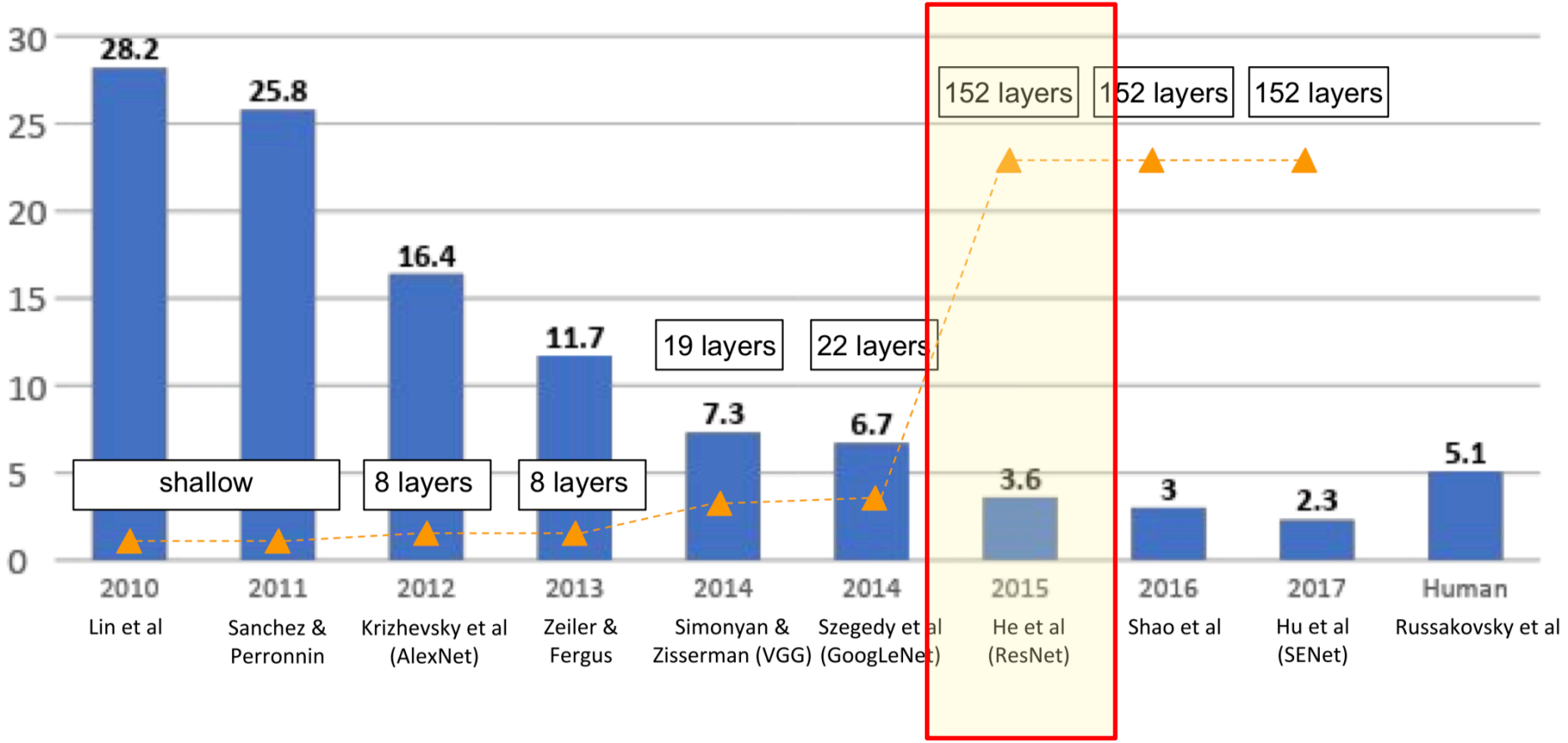

30
25
20
15
10
5
0
28.2
25.8
16.4
11.7
7.3
6.7
3.6
3
2.3
5.1
First CNN-based winner
shallow
8 layers
8 layers
19 layers
22 layers
152 layers
152 layers
152 layers
2010
Lin et al
2011
Sanchez & Perronnin
2012
Krizhevsky et al (AlexNet)
2013
Zeiler & Fergus
2014
Simonyan & Zisserman (VGG)
2014
Szegedy et al (GoogLeNet)
2015
He et al (ResNet)
2016
Shao et al
2017
Hu et al (SENet)
Human
Russakovsky et al


30
25
20
15
10
5
0
28.2
25.8
16.4
11.7
7.3
6.7
3.6
3
2.3
5.1
ZFNet: Improved hyperparameters over AlexNet
shallow
8 layers
8 layers
19 layers
22 layers
152 layers
152 layers
152 layers
2010
Lin et al
2011
Sanchez & Perronnin
2012
Krizhevsky et al (AlexNet)
2013
Zeiler & Fergus
2014
Simonyan & Zisserman (VGG)
2014
Szegedy et al (GoogLeNet)
2015
He et al (ResNet)
2016
Shao et al
2017
Hu et al (SENet)
Human
Russakovsky et al


30
25
20
15
10
5
0
28.2
25.8
16.4
11.7
7.3
6.7
3.6
3
2.3
5.1
shallow
8 layers
8 layers
19 layers
22 layers
Deeper Networks
152 layers
152 layers
152 layers
2010
2011
2012
2013
2014
2014
2015
2016
2017
Human
Lin et al
Sanchez & Perronnin
Krizhevsky et al (AlexNet)
Zeiler & Fergus
Simonyan & Zisserman (VGG)
Szegedy et al (GoogLeNet)
He et al (ResNet)
Shao et al
Hu et al (SENet)
Russakovsky et al


"Revolution of Depth"
30
25
20
15
10
5
0
28.2
25.8
16.4
11.7
7.3
6.7
3.6
3
2.3
5.1
shallow
8 layers
8 layers
19 layers
22 layers
152 layers
152 layers
152 layers
2010
2011
2012
2013
2014
2014
2015
2016
2017
Human
Lin et al
Sanchez & Perronnin
Krizhevsky et al (AlexNet)
Zeiler & Fergus
Simonyan & Zisserman (VGG)
Szegedy et a (GoogLeNet)
He et al (ResNet)
Shao et al
Hu et al (SENet)
Russakovsky et al

Revolution of Depth

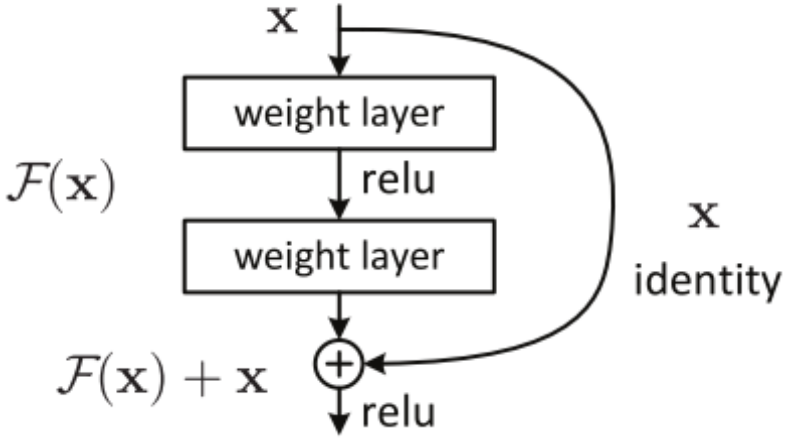


Microsoft's ResNet

[He et al., 2015]

- WHAT?
  - Very deep networks using residual connections

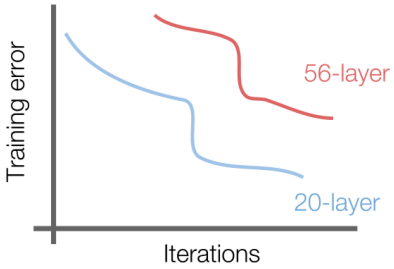
- 152-layer model for ImageNet
- ILSVRC'15 classification winner (3.57% top 5 error)
- Swept all classification and detection competitions in ILSVRC'15 and COCO'15!

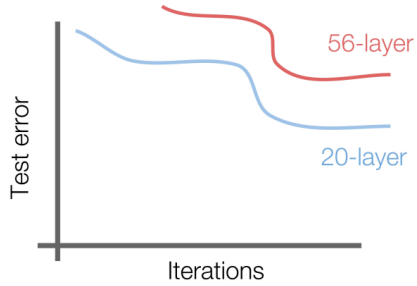


Microsoft's ResNet

[He et al., 2015]

- WHY?
  - What happens when we continue stacking deeper layers on a "plain" CNN?





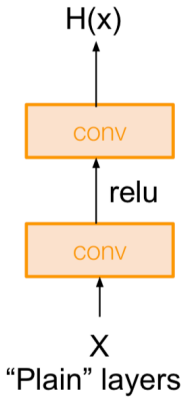
Hypothesis:
- the problem is an *optimization* problem, deeper models are harder to optimize
- Harder because the gradient vanishing and exploding effect

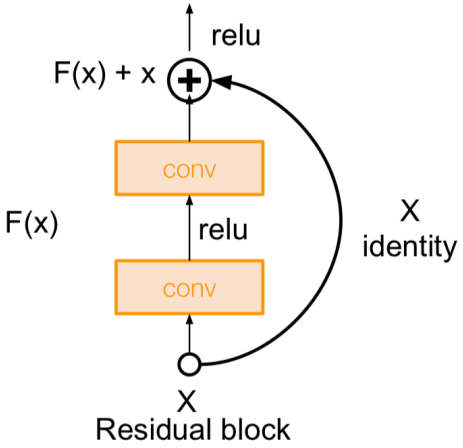
56-layer model performs worse on both training and test error
-> The deeper model performs worse, but it's not caused by overfitting!

Microsoft's ResNet

[He et al., 2015]

- HOW?
  - Use network layers to fit a residual mapping instead of directly trying to fit a desired underlying mapping



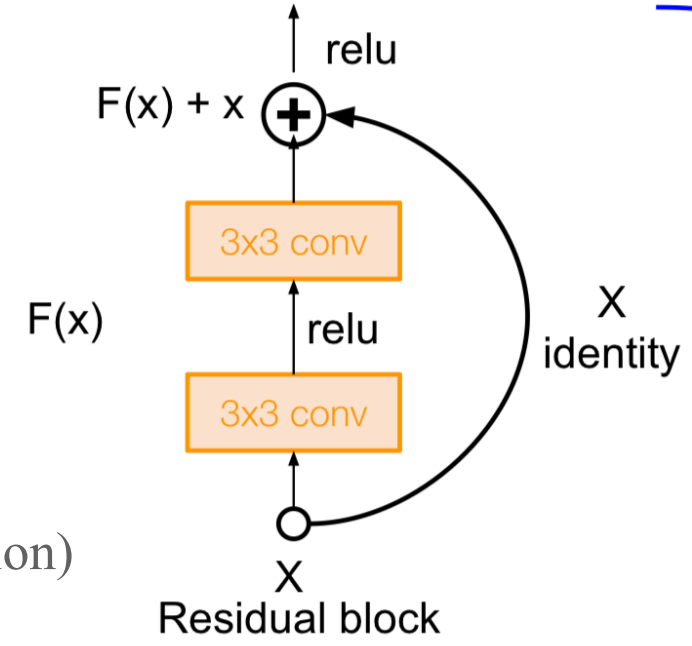


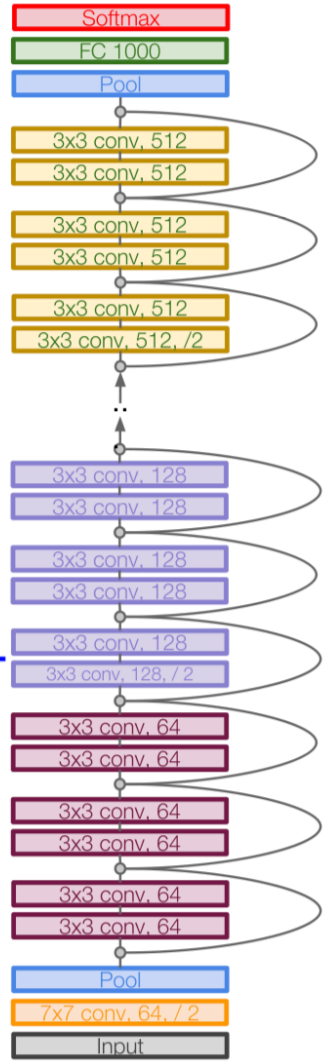
Microsoft's ResNet

[He et al., 2015]

Full ResNet architecture:

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2 (/2 in each dimension)

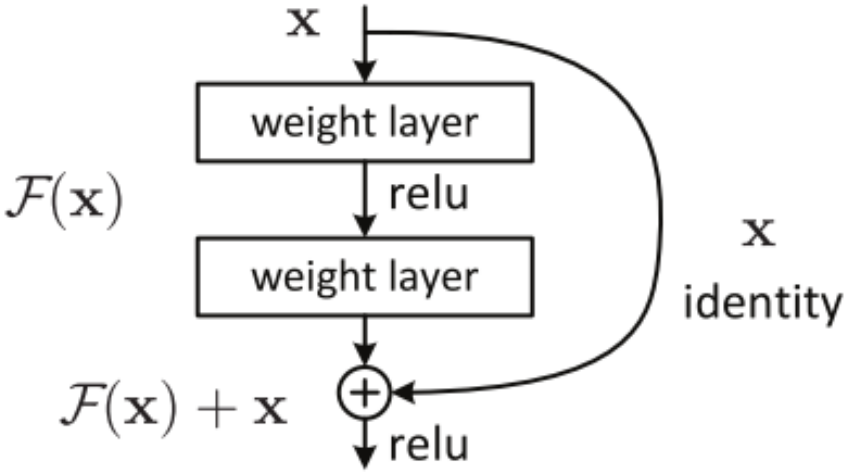


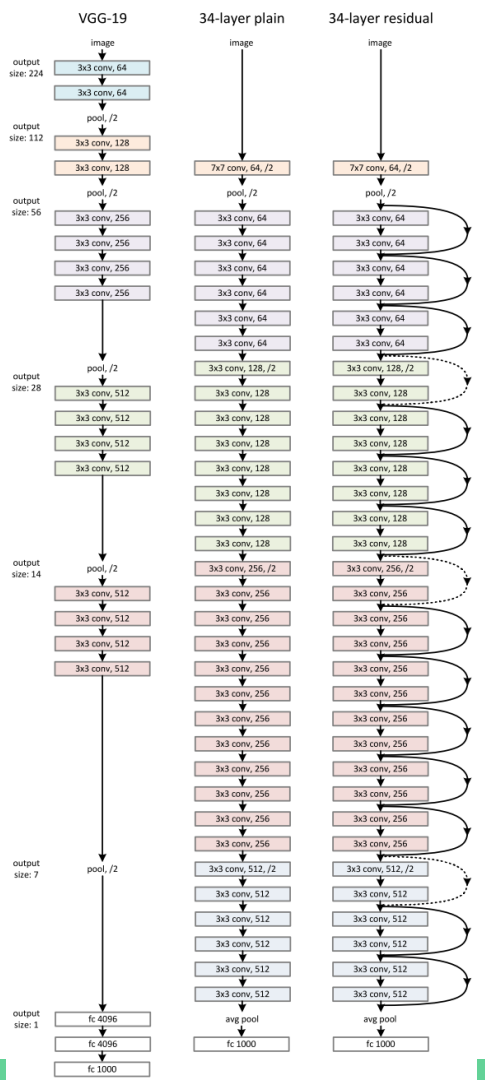


Microsoft's ResNet

[He et al., 2015]

- Very deep networks using residual connections





ResNet variants

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Training ResNet in practice

- Batch Normalization after every CONV layer
- Xavier 2/ initialization from He et al.
- SGD + Momentum (0.9)
- Learning rate: 0.1, divided by 10 when validation error plateaus
- Mini-batch size 256
- Weight decay of 1e-5
- No dropout used

Implementations on popular platforms

- Tensorflow implementation
  - https://github.com/tensorflow/tensorflow/tree/361a82d73a50a800510674b3aaa20e4845e56434/tensorflow/contrib/slim/python/slim/nets

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

Experimental Results

- Able to train very deep

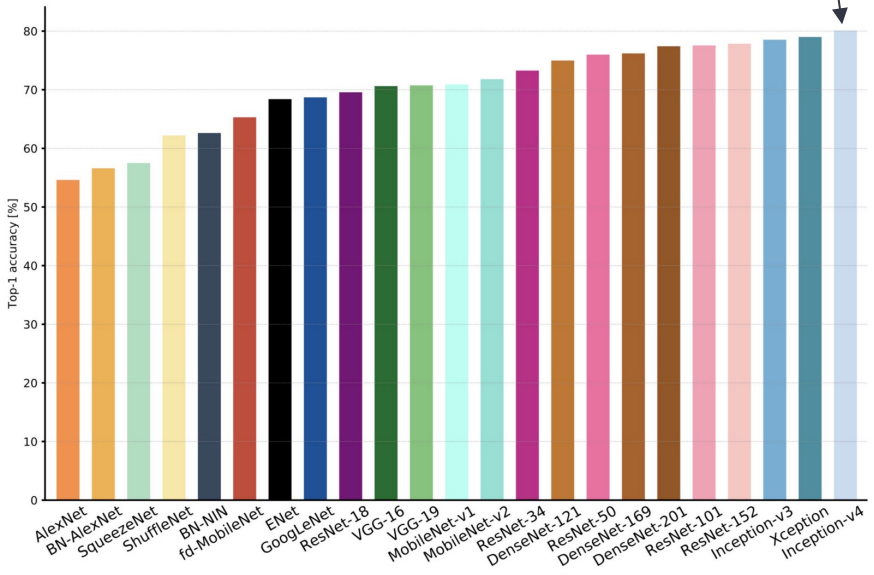
networks without degrading (152 layers on ImageNet, 1202 on Cifar)

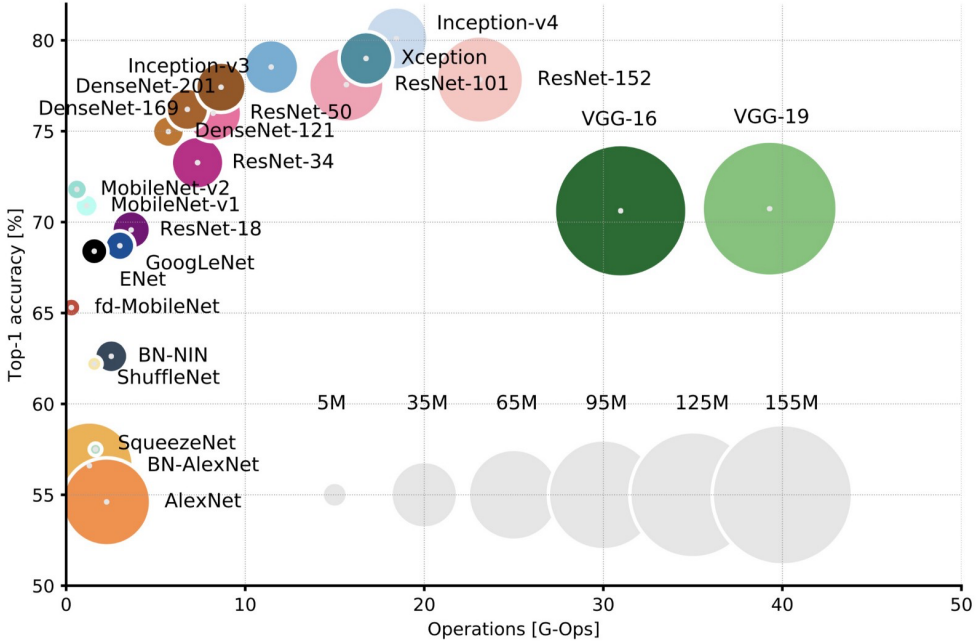
- Deeper networks now achieve lowing training error as expected
- Swept 1st place in all ILSVRC and COCO 2015 competitions

ILSVRC 2015 classification winner (3.6% top 5 error) -- better than "human performance"! (Russakovsky 2014)

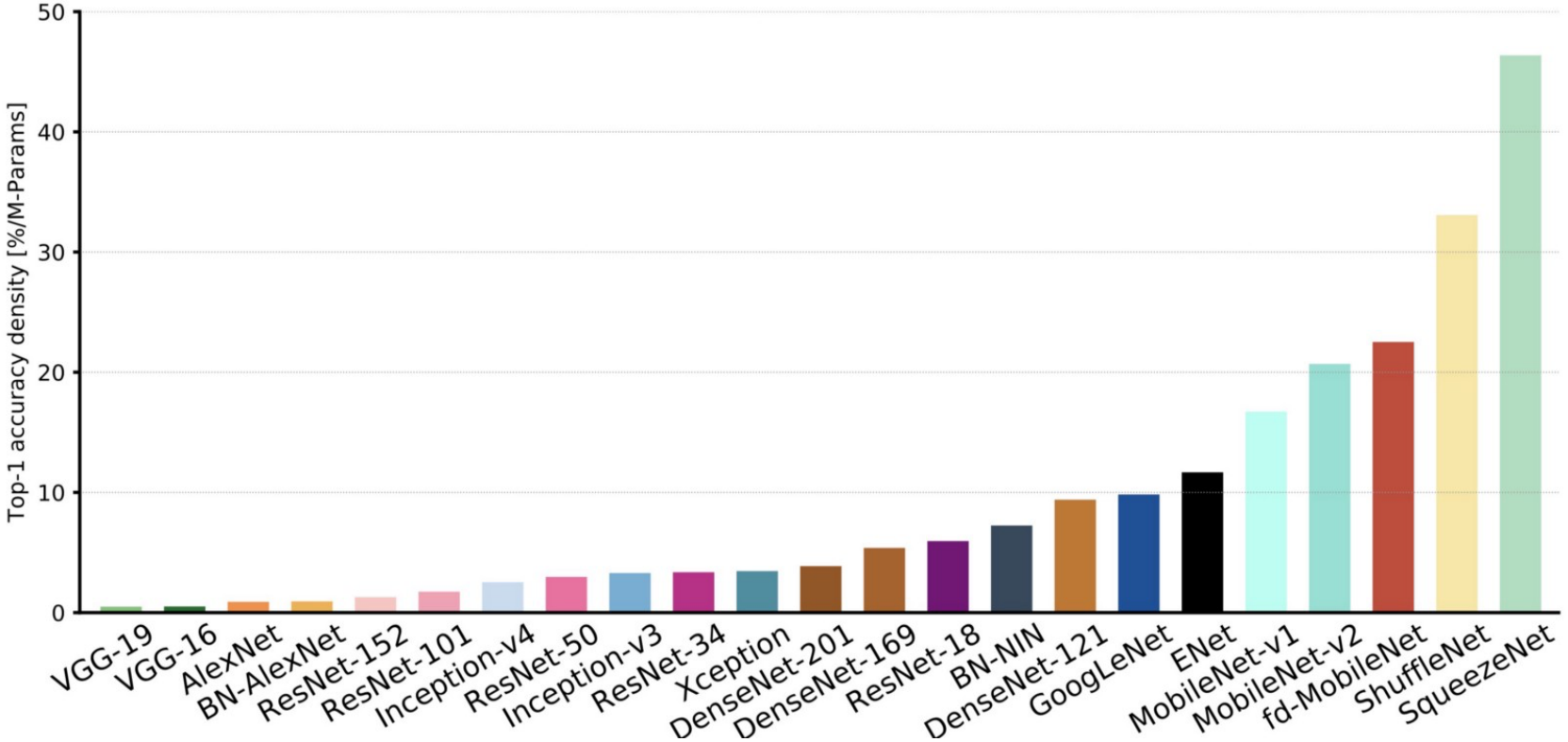
Top1 vs. operations vs. parameters

Inception-v4: Resnet + Inception!





Accuracy per parameter



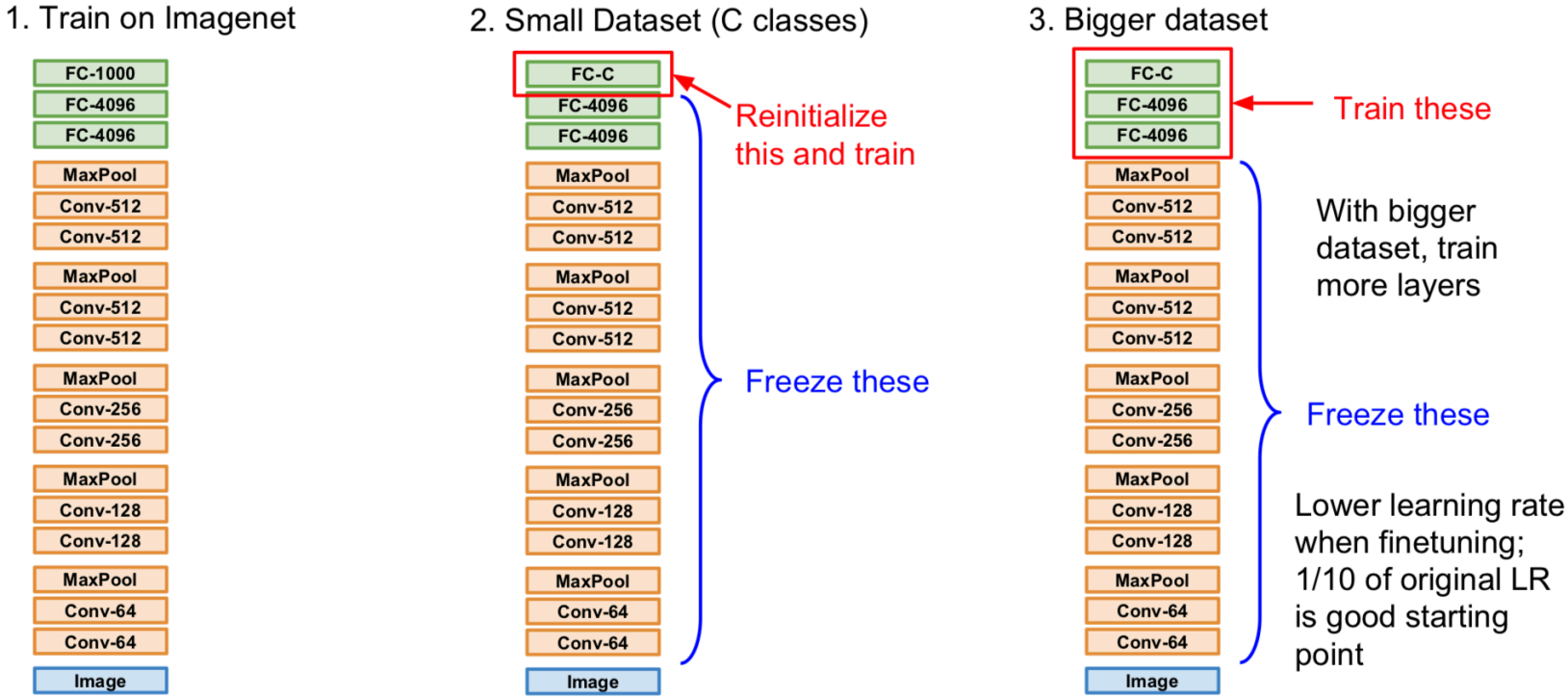
Transfer Learning

- **WHAT?**
  - Instead, it is common to pretrain a ConvNet on a very large dataset (e.g. ImageNet, which contains 1.2 million images with 1000 categories), then apply Transfer Learning to adapt the pretrained model to a new task
- **WHY?**
  - Computation: modern ConvNets take 2-3 weeks to train across multiple GPUs
  - Data: on ImageNet (>1M images)
  - => Rarely you have these luxury resources available

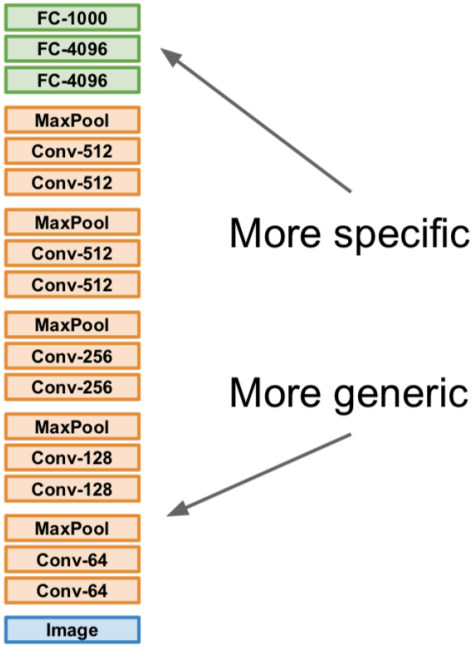
Transfer Learning

- **HOW?**
  - ConvNet as fixed feature extractor
  - Fine-tuning the ConvNet

Transfer Learning



Transfer Learning



	very similar dataset	**very different dataset**
very little data	Use Linear Classifier on top layer	You're in trouble… Try linear classifier from different stages
quite a lot of data	Finetune a few layers	Finetune a larger number of layers

Google Colab

- Google Colab is a *free cloud* service and now it supports free *GPU*!

- You can;
  - improve your **Python** programming language coding skills.
  - develop deep learning applications using popular libraries such as **Keras**, **TensorFlow**, **PyTorch, OpenCV**

Google Colab

- Tutorial
  - https://medium.com/deep-learning-turkey/google-colab-free-gpu-tutorial-e113627b9f5d

  - **Creating Folder on Google Drive**
  - **Creating New Colab Notebook**
  - **Setting Python version and Free CPU/GPU**
  - **Running Basic Python Codes with Google Colab**
  - Accessing data from Google Drive

Resnet and Transfer learning tutorial codes on Colab

- Design your own simple CNN
- Use pretrained Resnets for classification
- Visualize Resnet networks
- Design your own ResNet from scratch
- Transfer learning with Resnet

- All my tutorial codes are here:
  - Keras:
    - https://colab.research.google.com/drive/1iVChsQa7dAJych6gciq21QZB5tEVlzIw
    - https://colab.research.google.com/drive/159QWa2LUIf8cQQ2ksnKVjBYnEb6wNKb3
  - Pytorch: https://colab.research.google.com/drive/1MoE5CvDz5pOOoJkmWRstqqb1JzD46Lmt