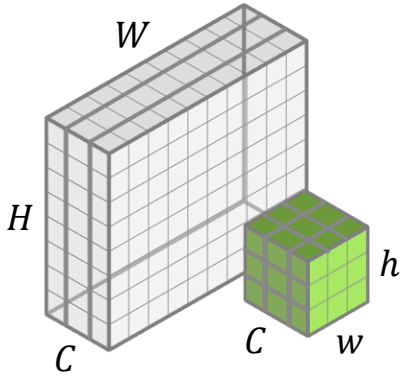
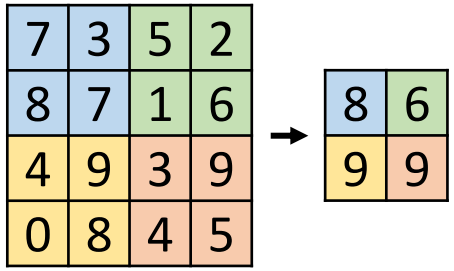
CNN Architectures

November 29th, 2022

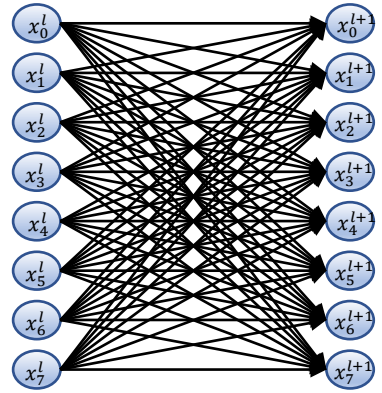
The ConvNet Building Blocks



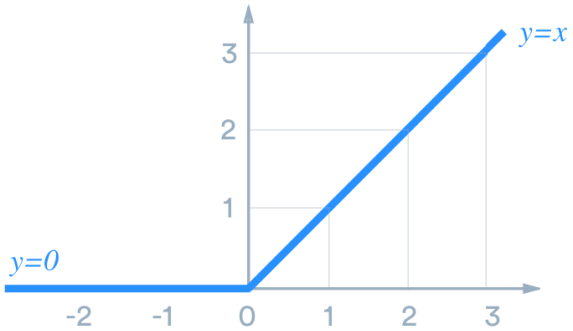
Conv Layer



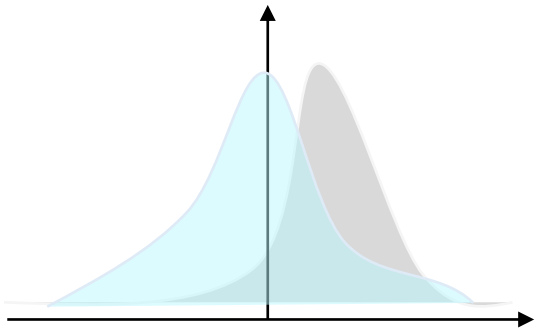
Max Pooling



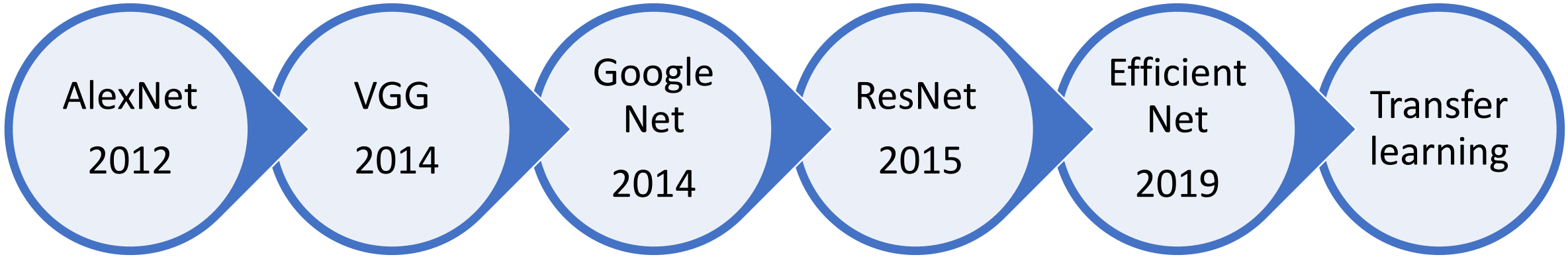
FC Layer



Activation Func.



Batch Norm


AlexNet
2012
VGG
2014
Google
Net
2014
ResNet
2015
Efficient
Net
2019
Transfer
learning

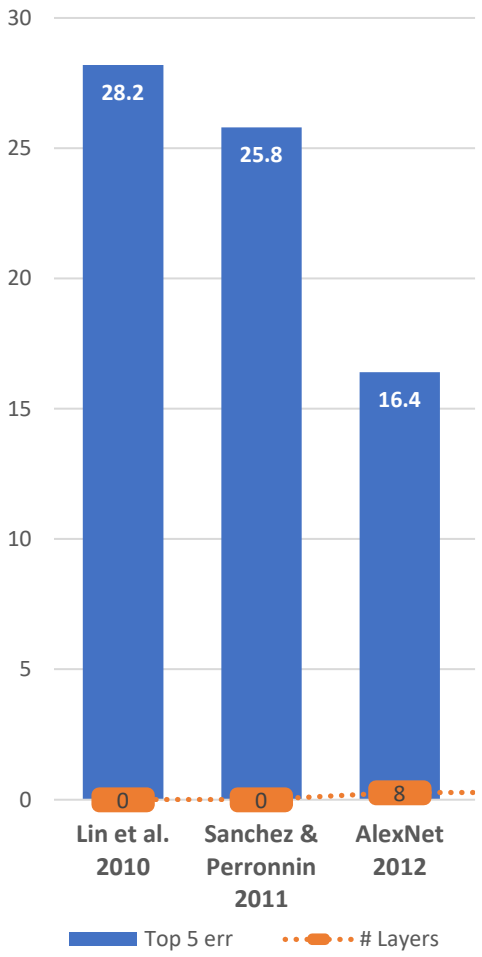
IMAGENET

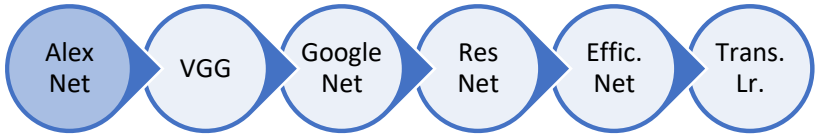
Visual Recognition Challenge (ILSVRC)

- 1.2M training images
- 100K testing images
- 1K categories

Olga Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015.

AlexNet

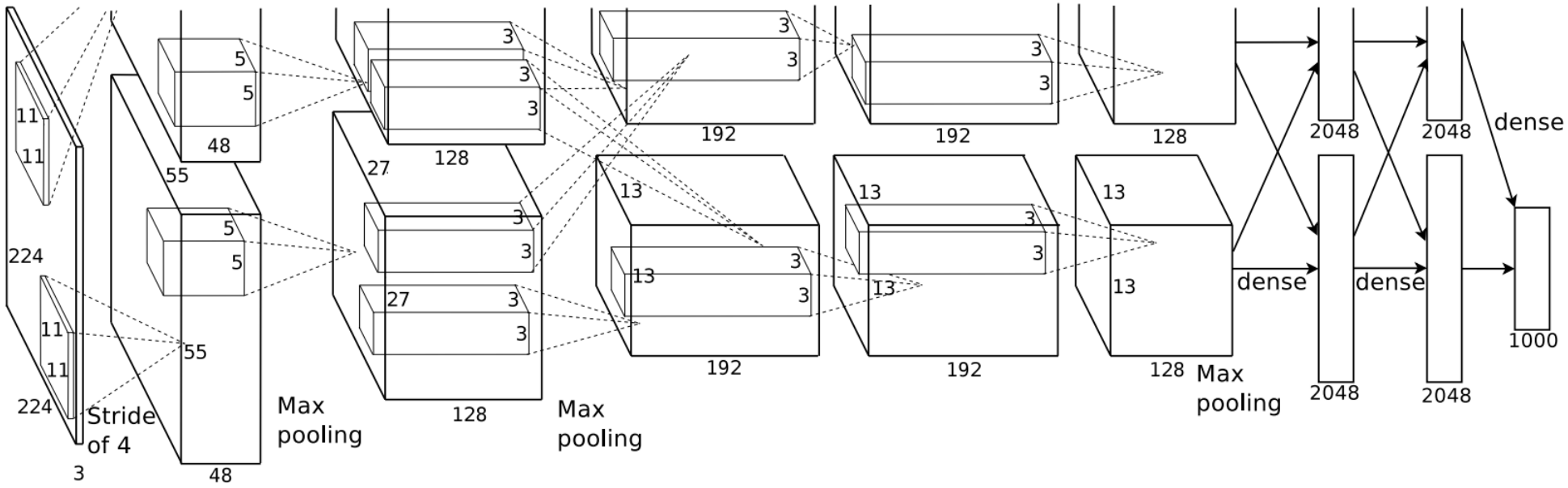




- Fukushima 1980
- LeCunn et al. 1989
- LeCunn et al. 1998

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks.NIPS, 2012

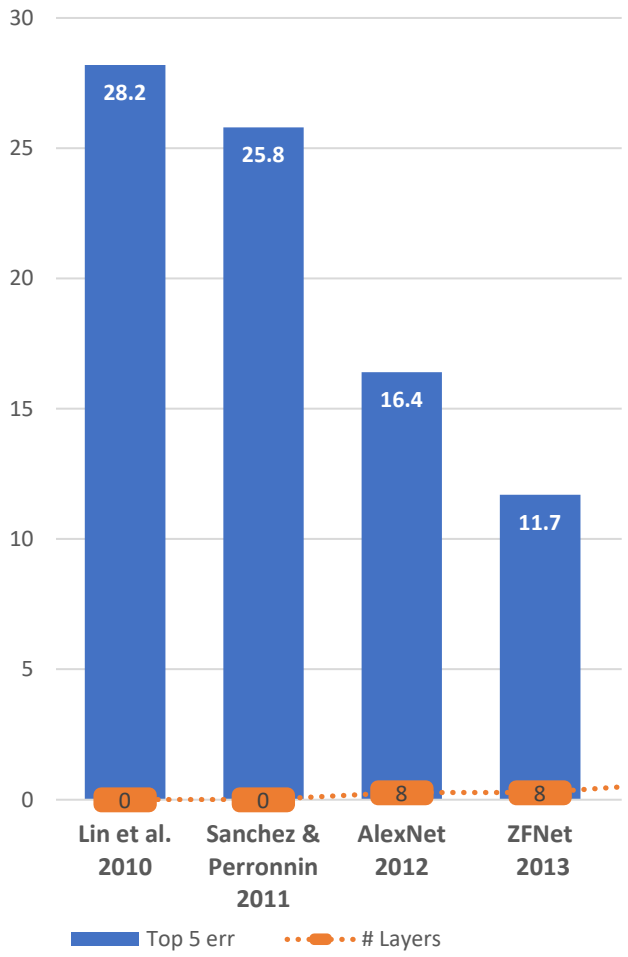
AlexNet

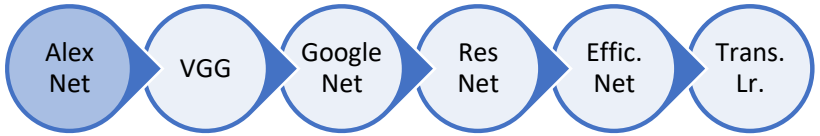


Details:

- ReLU activation
- Max Pooling
- Dropout
- Batch
- SGD Momentum

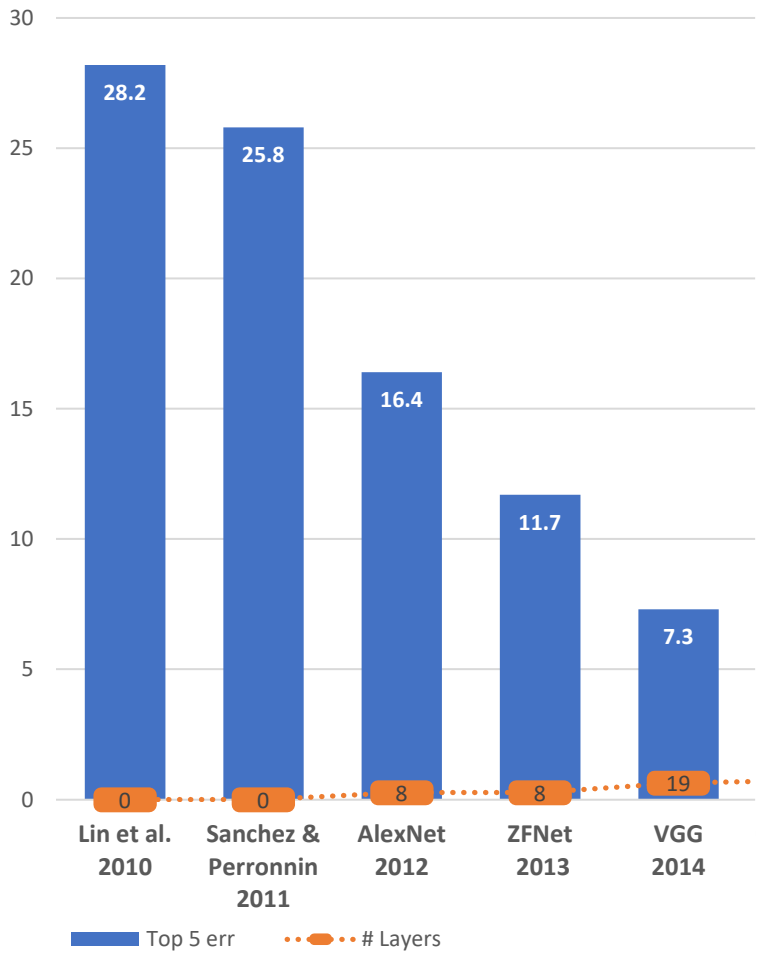
ZFNet

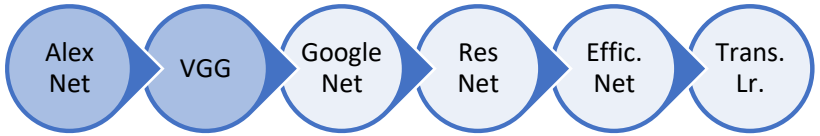


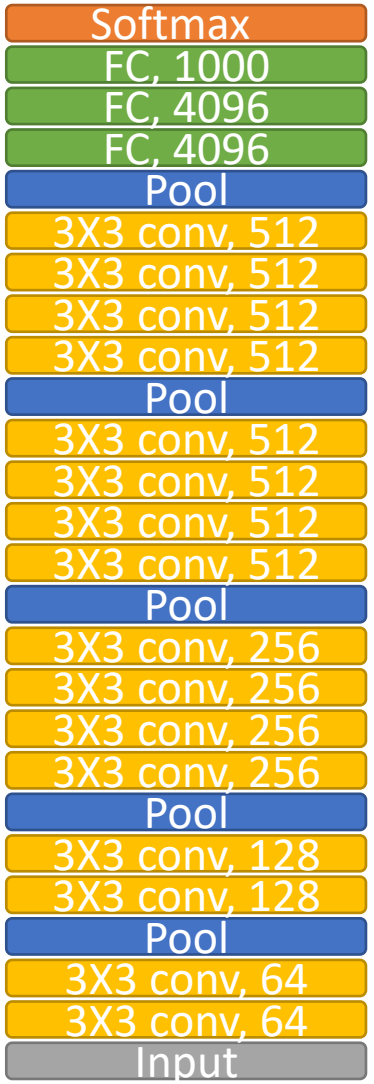


Matthew D Zeiler, Rob Fergus. Visualizing and Understanding Convolutional Networks. ECCV, 2014.

VGG

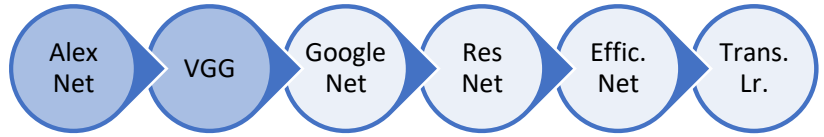




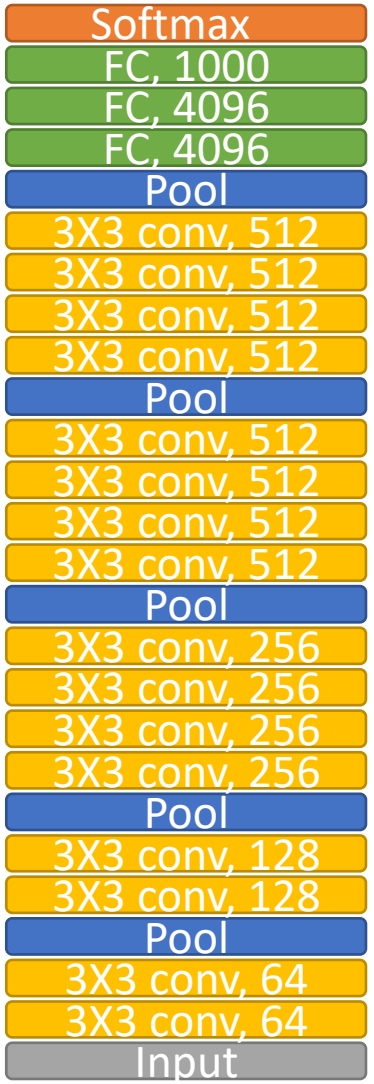


Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

VGG

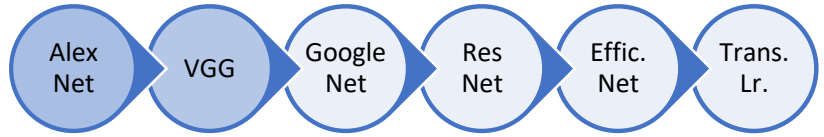


- All conv layers are 3x3, stride 1, pad 1
- All pooling layers are 2x2, stride 2
- After pooling, double the number of channels



Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

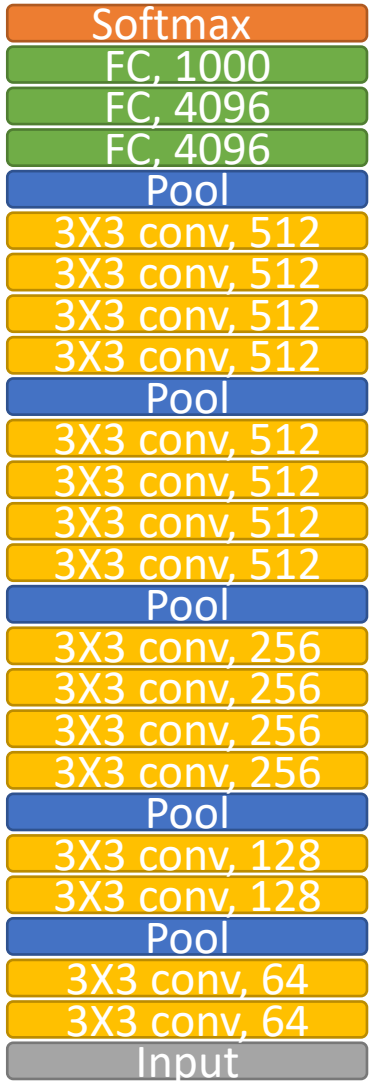
Why using 3x3 conv layers?



5X5 conv, C

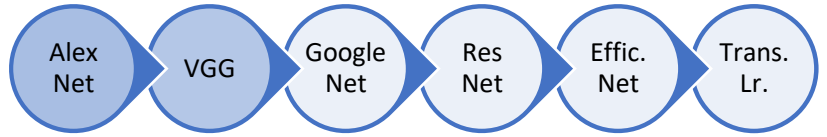
3X3 conv, C
3X3 conv, C

Receptive Field

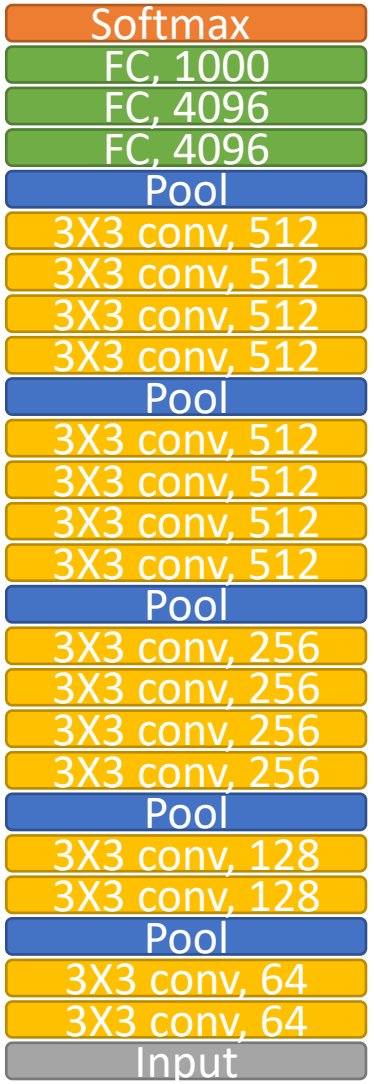


Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

Why using 3x3 conv layers?

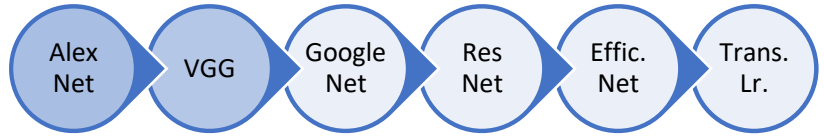


	5X5 conv, C	3X3 conv, C 3X3 conv, C
Receptive Field	5	5
# Parameters		

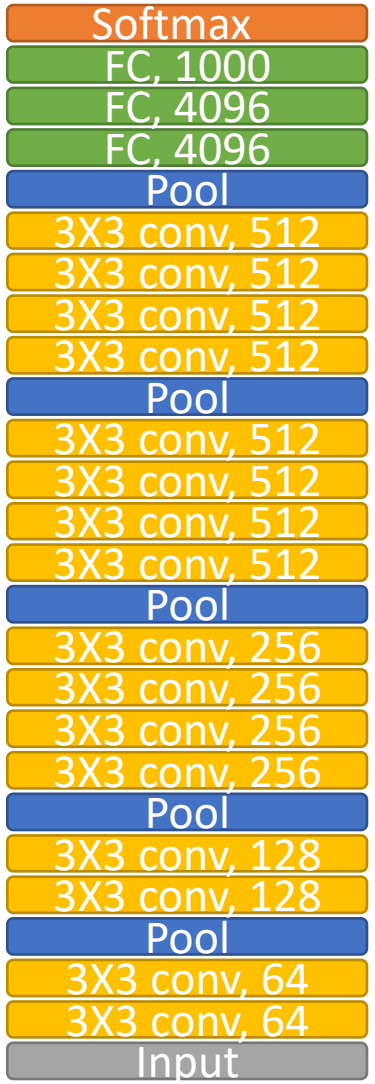


Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

Why using 3x3 conv layers?

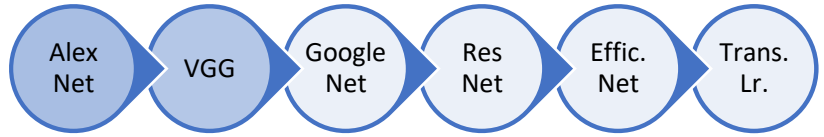


	5X5 conv, C	3X3 conv, C 3X3 conv, C
Receptive Field	5	5
# Parameters	$25C^2$	$9C^2+9C^2 = 18C^2$
# FLOPs		

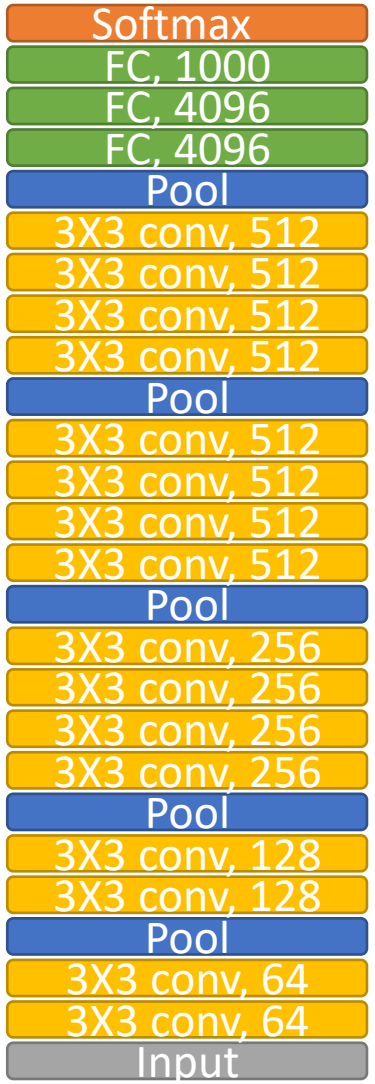


Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

Why using 3x3 conv layers?

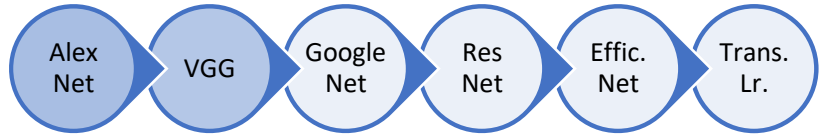


	5X5 conv, C	3X3 conv, C 3X3 conv, C
Receptive Field	5	5
# Parameters	$25C^2$	$9C^2+9C^2 = 18C^2$
# FLOPs	$25C^2HW$	$18C^2HW$

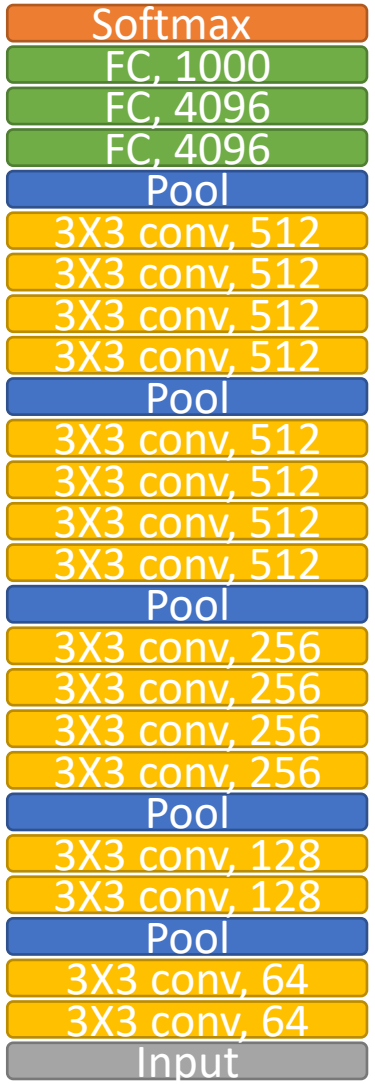


Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

Why using 3x3 conv layers?

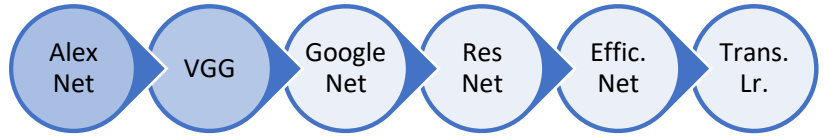


	5X5 conv, C	3X3 conv, C 3X3 conv, C
Receptive Field	5	5
# Parameters	$25C^2$	$9C^2+9C^2 = 18C^2$
# FLOPs	$25C^2HW$	$18C^2HW$
Memory Size (Output)		

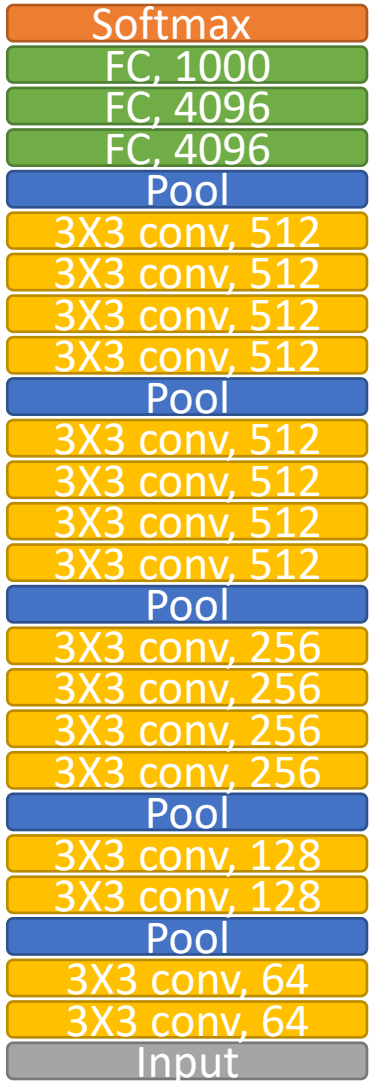


Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

Why using 3x3 conv layers?

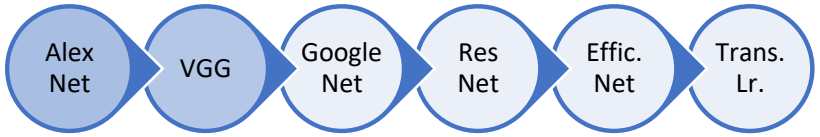


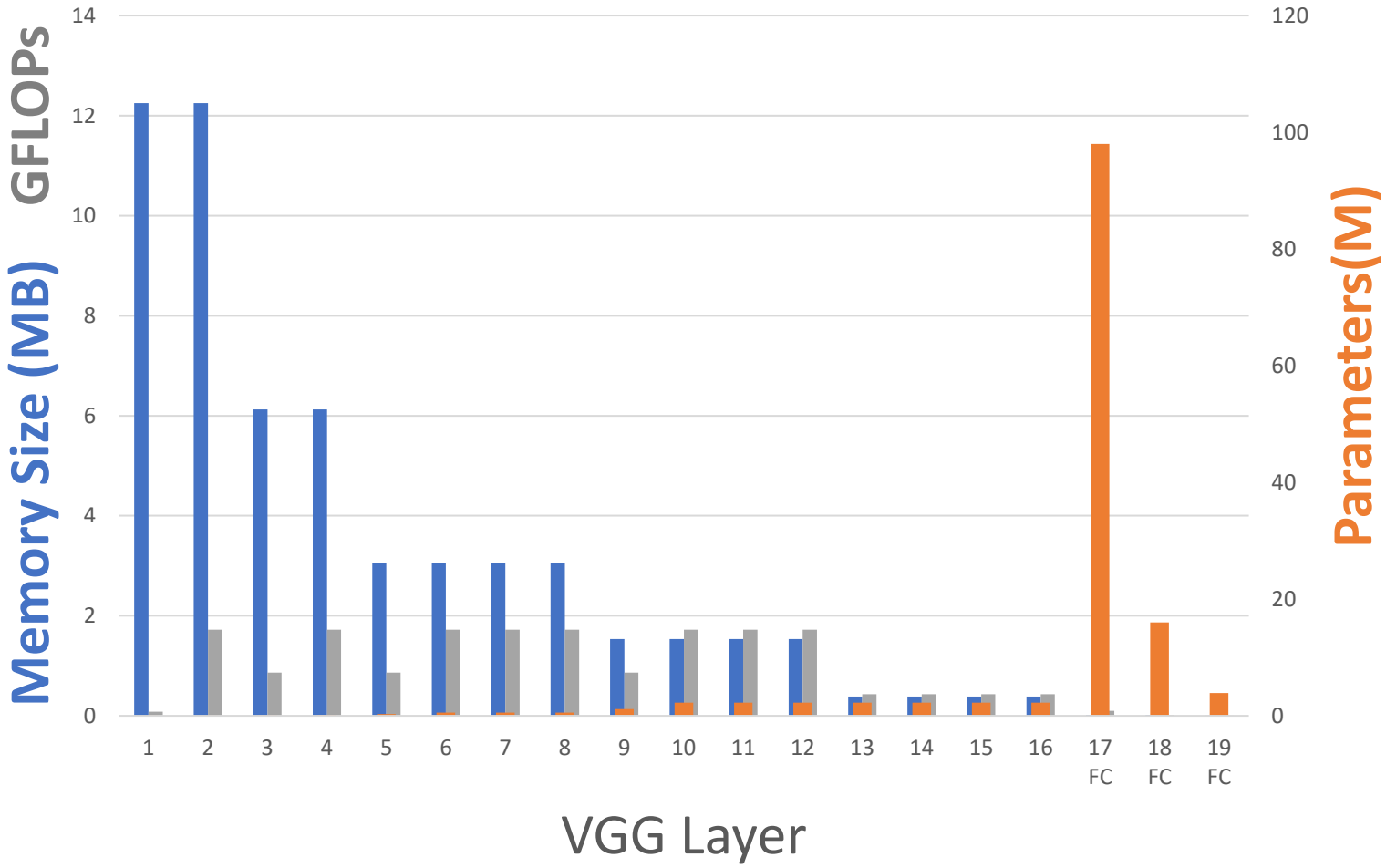
	5X5 conv, C	3X3 conv, C 3X3 conv, C
Receptive Field	5	5
# Parameters	$25C^2$	$9C^2+9C^2 = 18C^2$
# FLOPs	$25C^2HW$	$18C^2HW$
Memory Size (Output)	HWC	2HWC

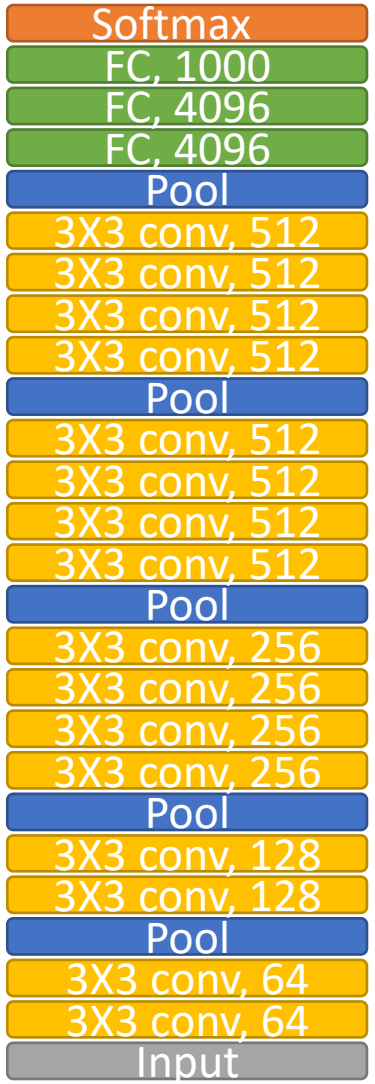


Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

VGG resources

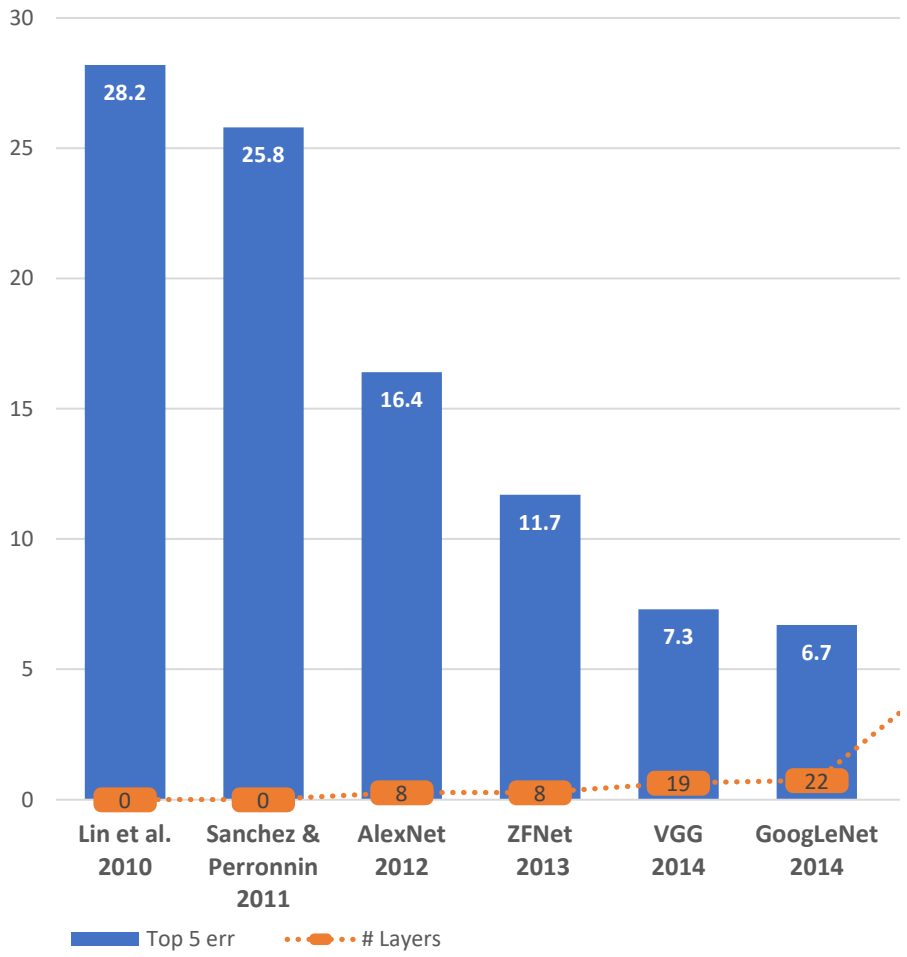


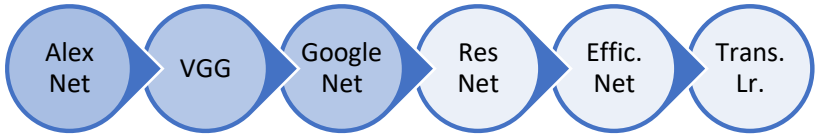




Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

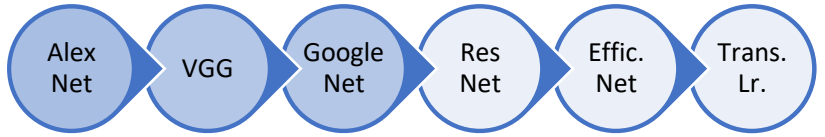
GoogLeNet

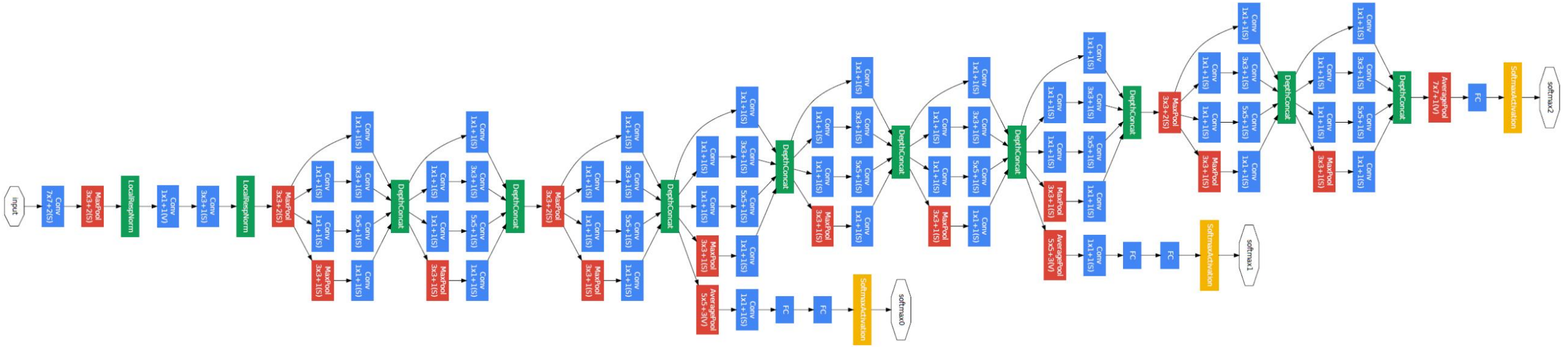




Christian Szegedy et al. Going deeper with convolutions. CVPR, 2015.

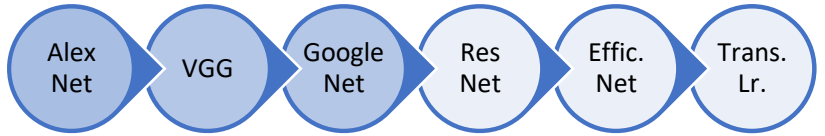
GoogLeNet

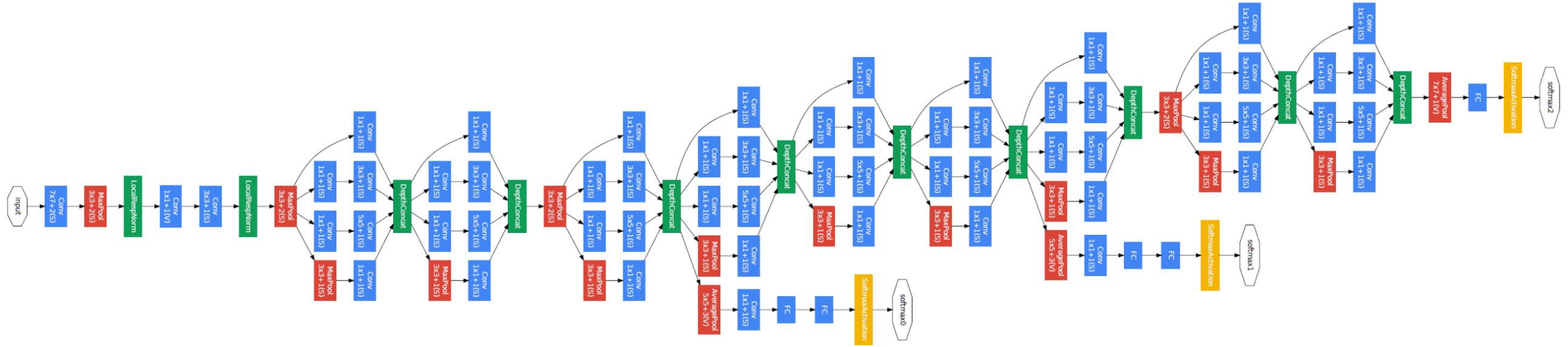


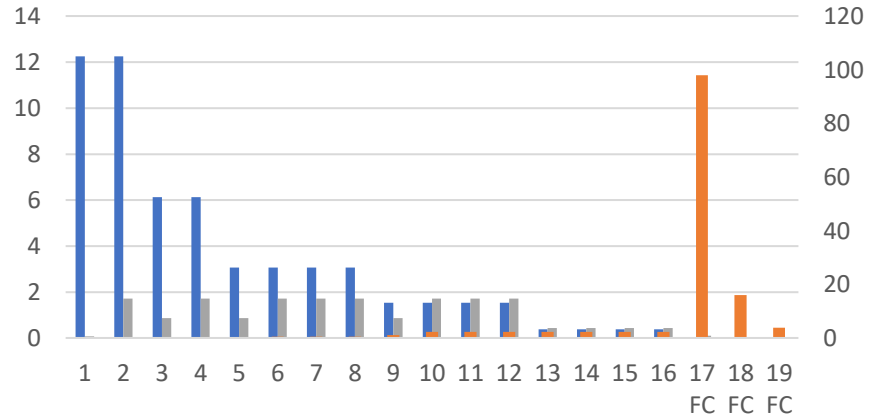


Christian Szegedy et al. Going deeper with convolutions. CVPR, 2015.

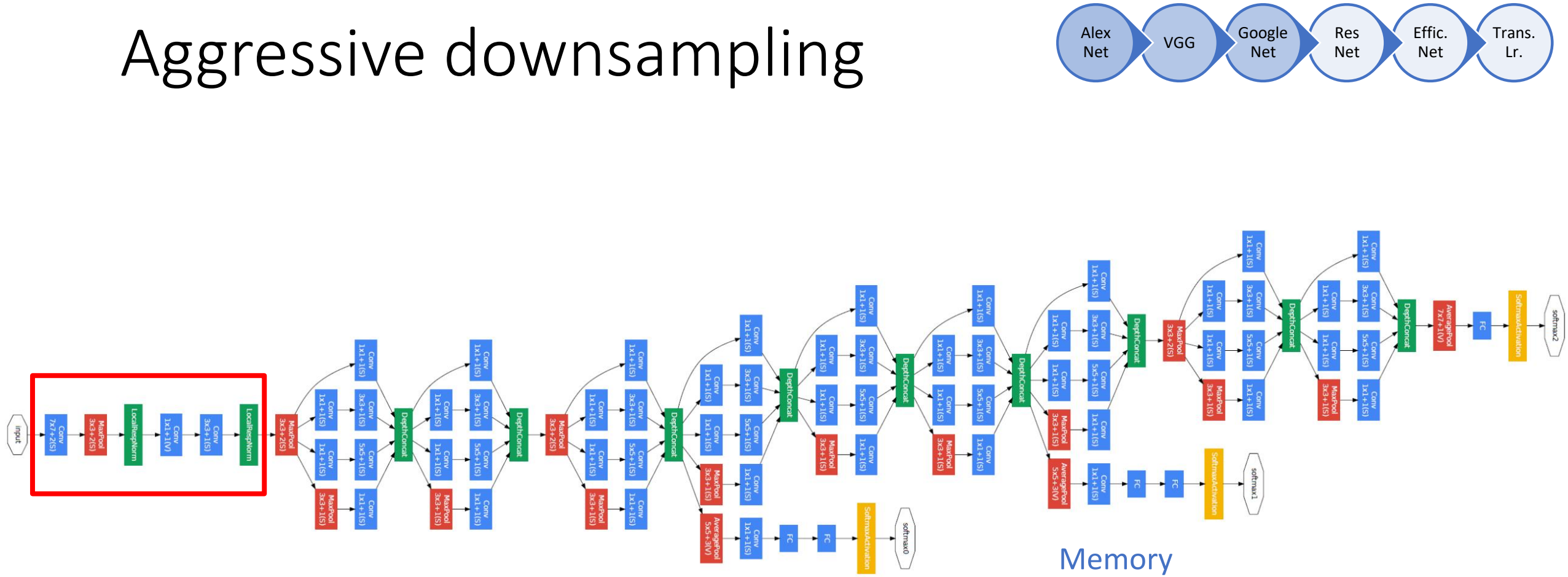
GoogLeNet


Alex Net
VGG
Google Net
Res Net
Effic. Net
Trans. Lr.


input
softmax0
softmax1
softmax2

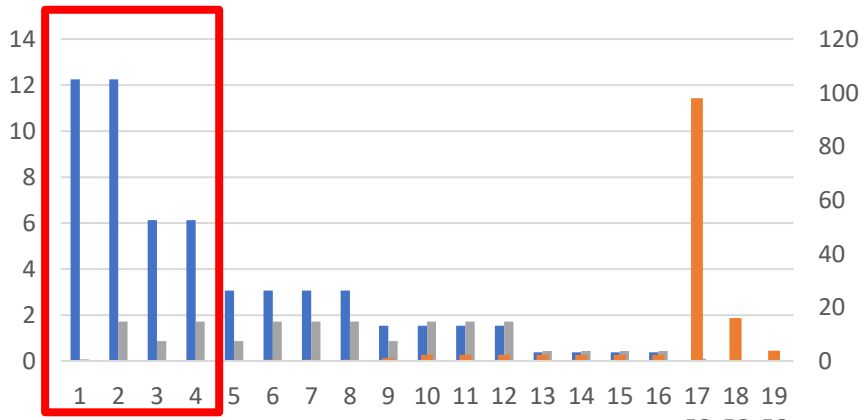

14
12
10
8
6
4
2
0
120
100
80
60
40
20
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
FC FC FC

Aggressive downsampling

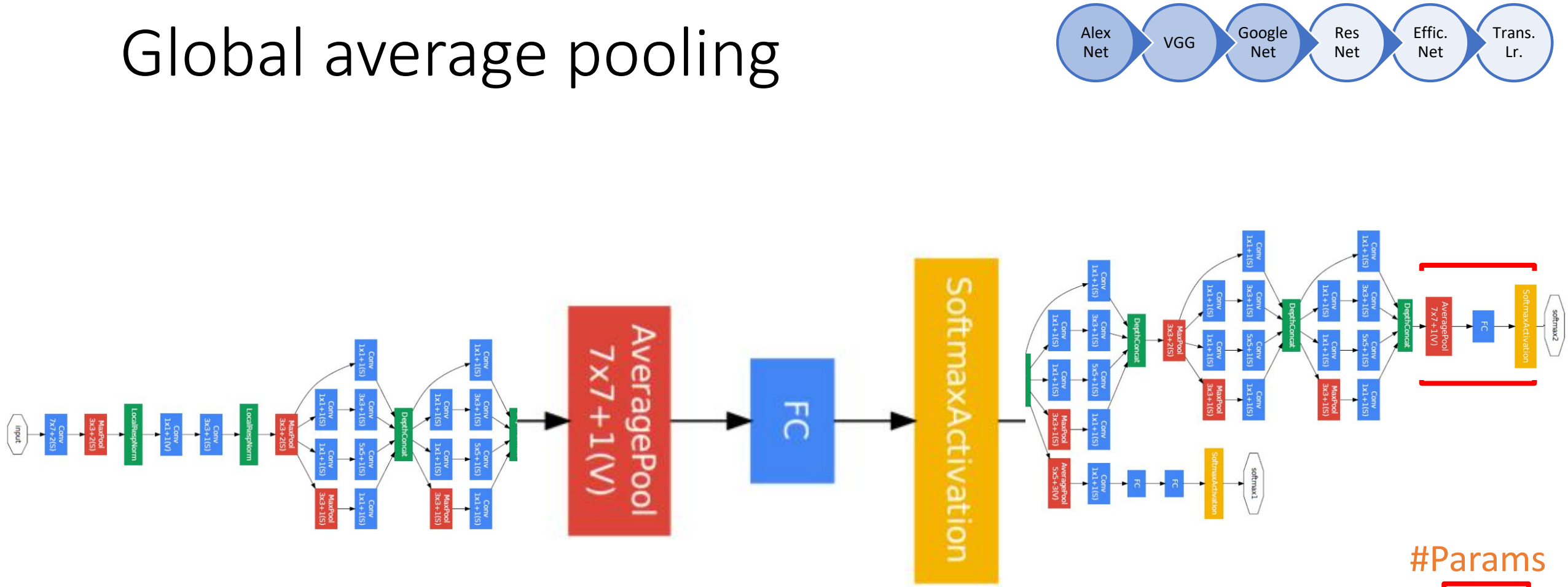


Alex Net → VGG → Google Net → Res Net → Effic. Net → Trans. Lr.

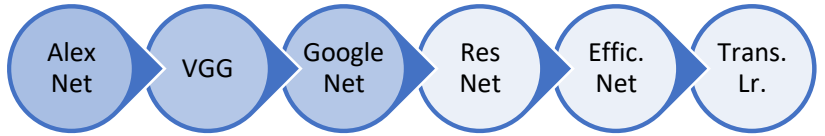
Memory

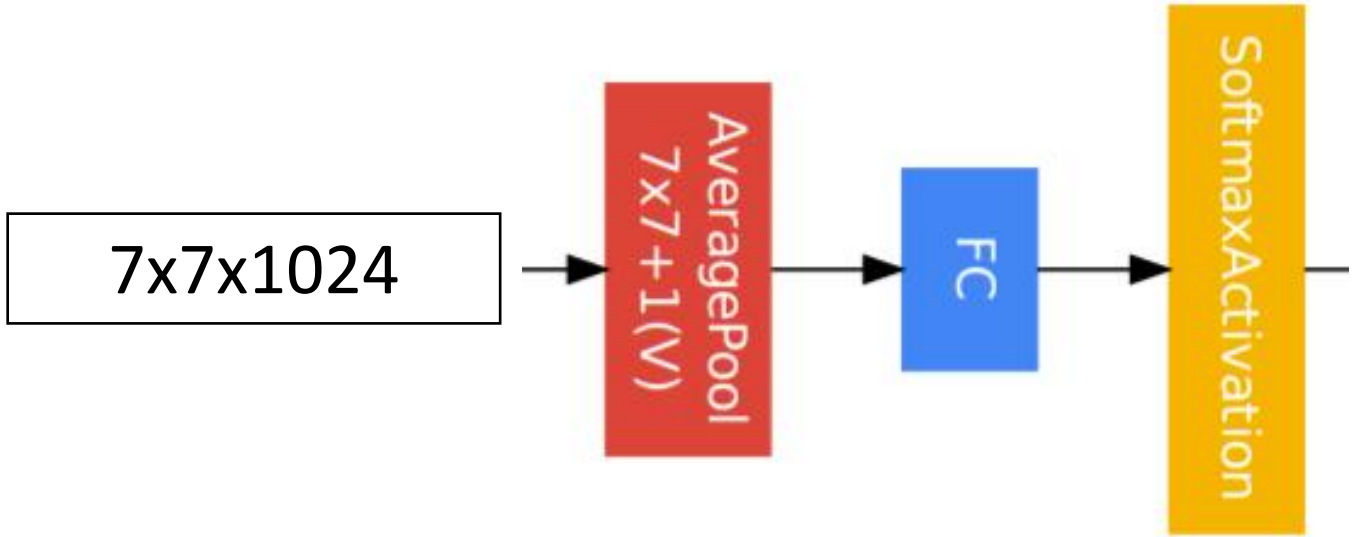


Global average pooling

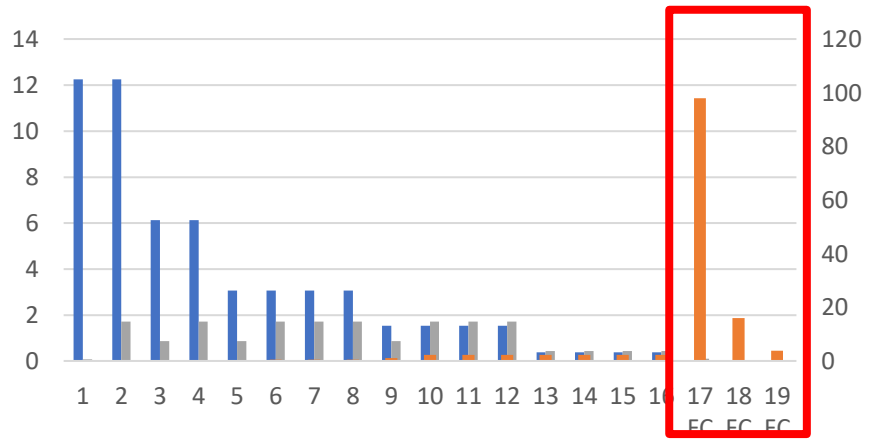


Global average pooling

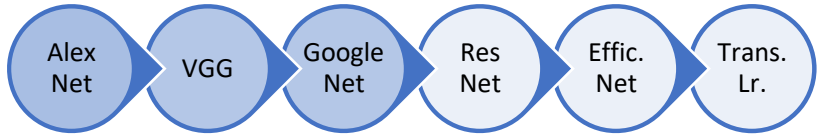


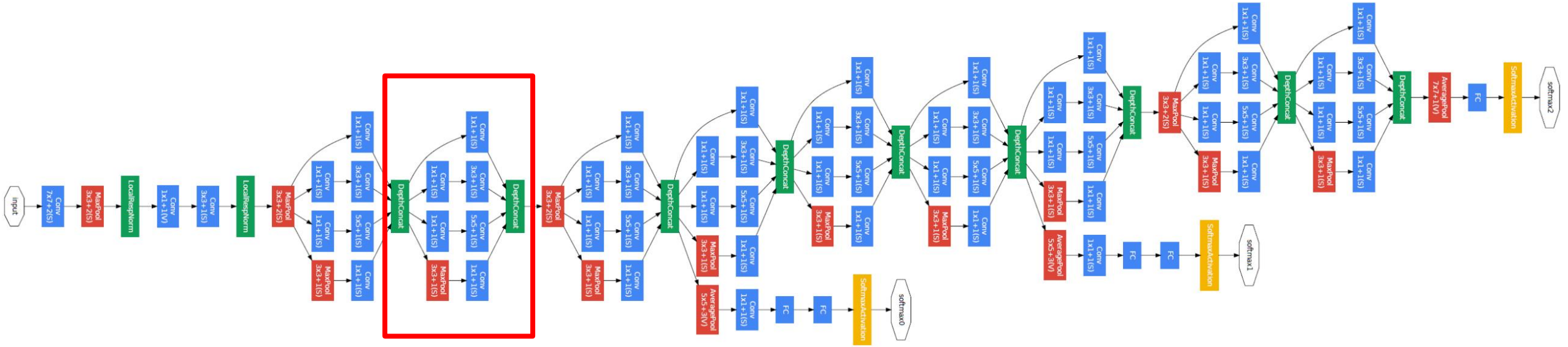


#Params



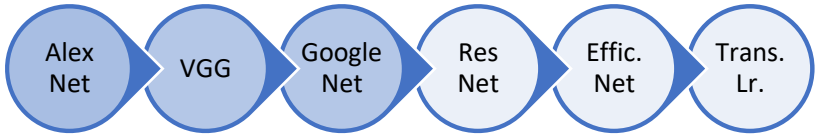
GoogLeNet

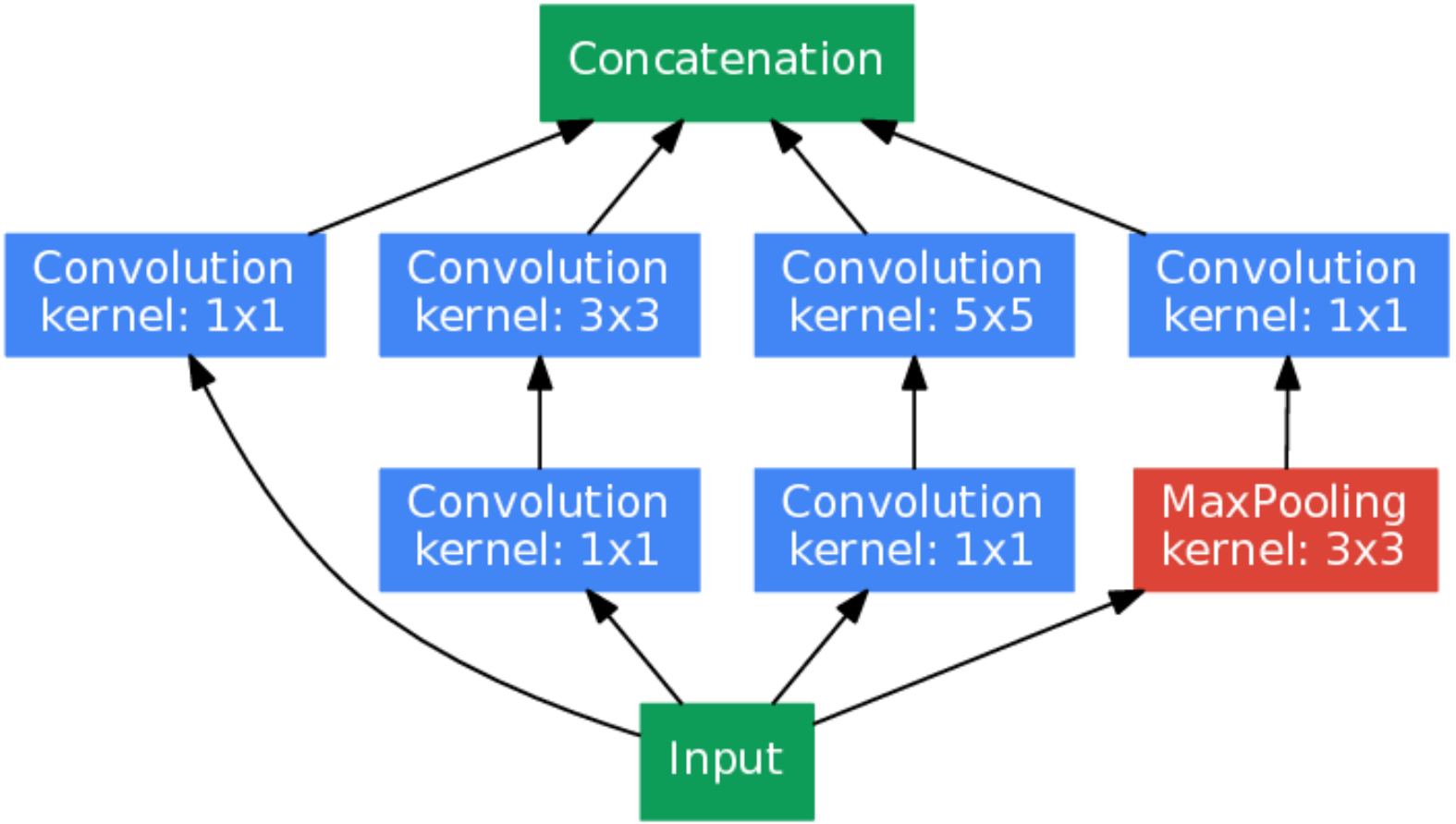




Christian Szegedy et al. Going deeper with convolutions. CVPR, 2015.

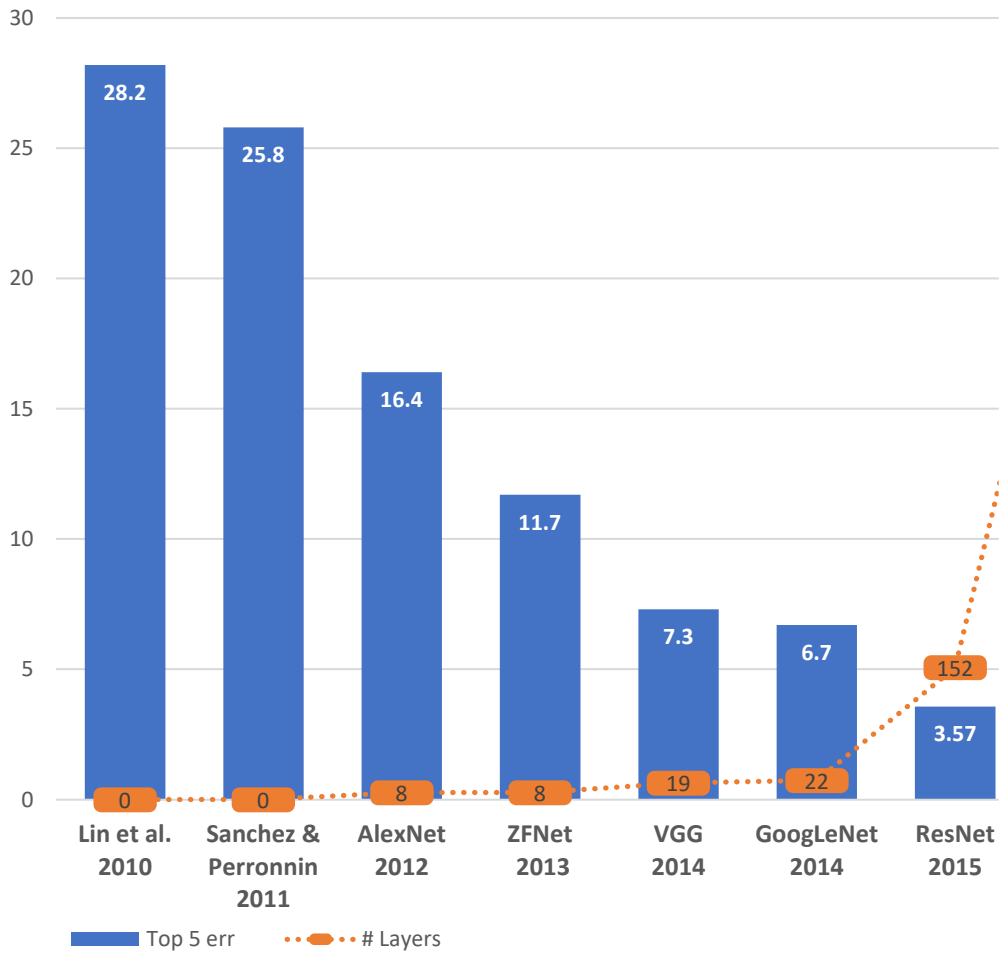
GoogLeNet: Inception blocks

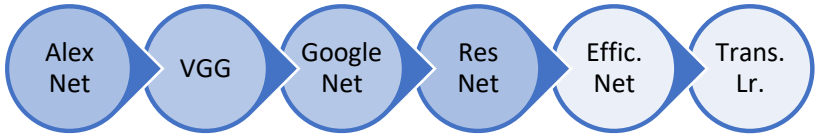




Christian Szegedy et al. Going deeper with convolutions. CVPR, 2015.

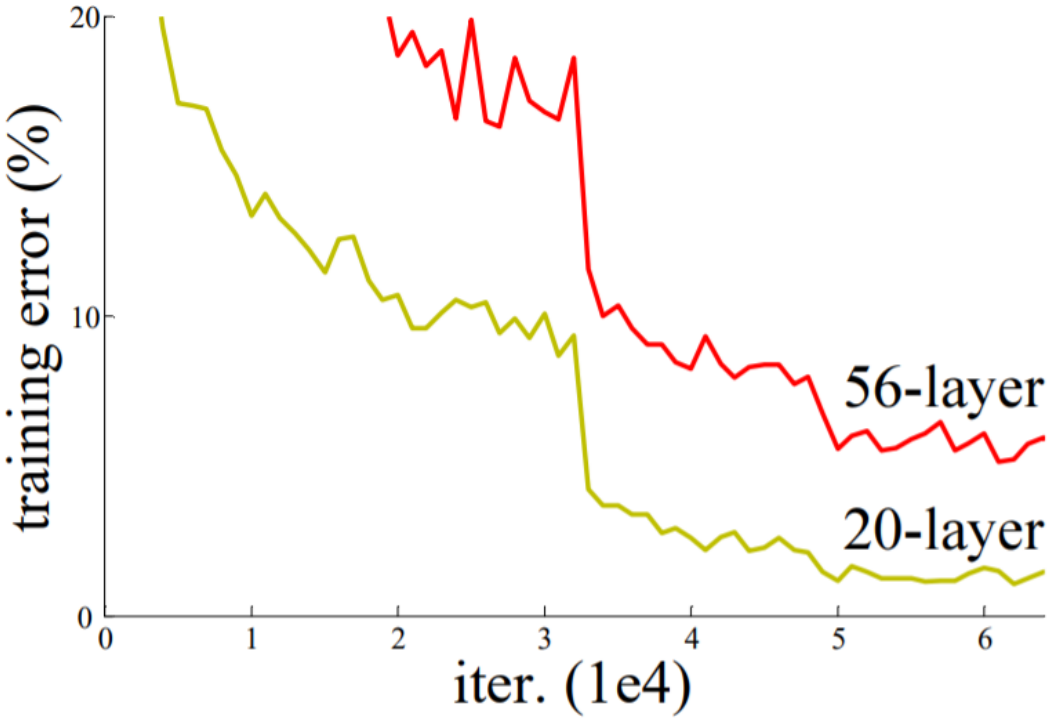
ResNet



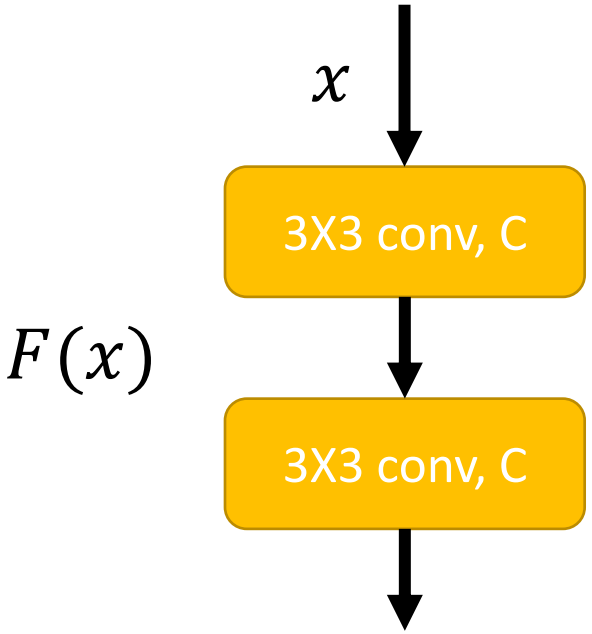


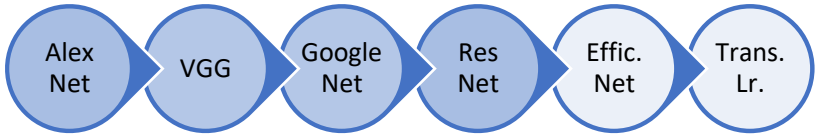
Kaiming He et al. Deep Residual Learning for Image Recognition. CVPR, 2016.

ResNet

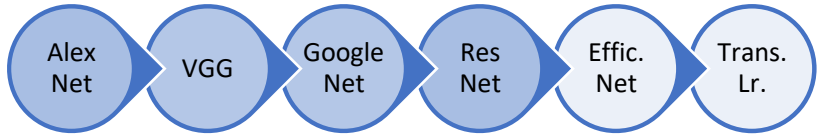


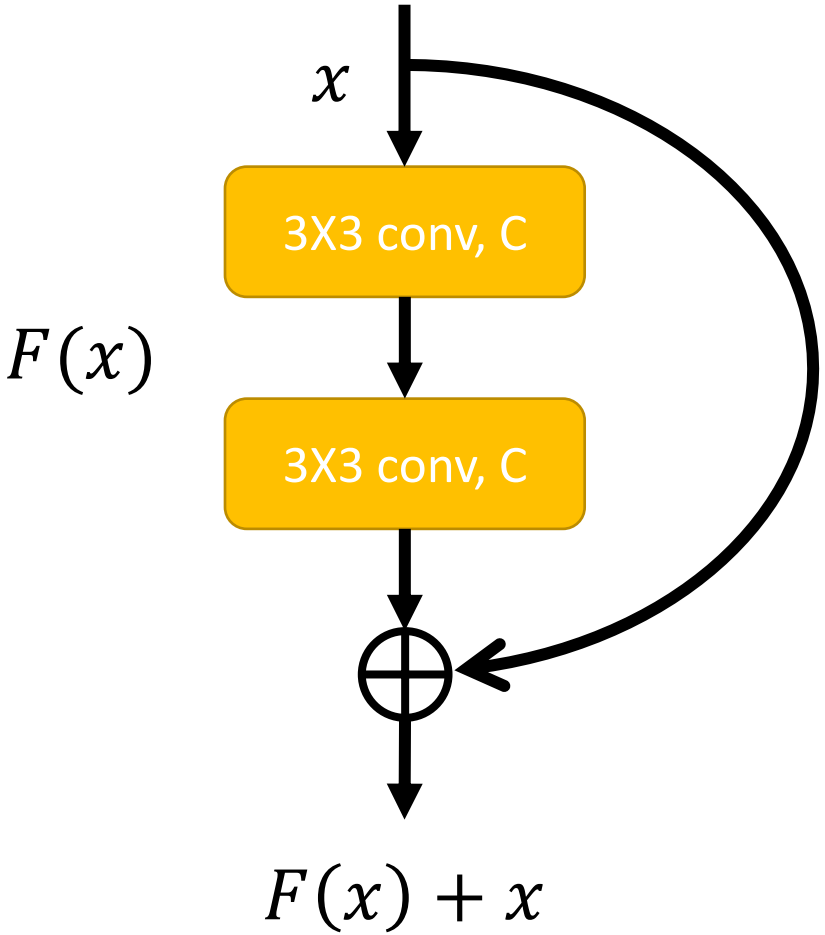
Residual Building Block



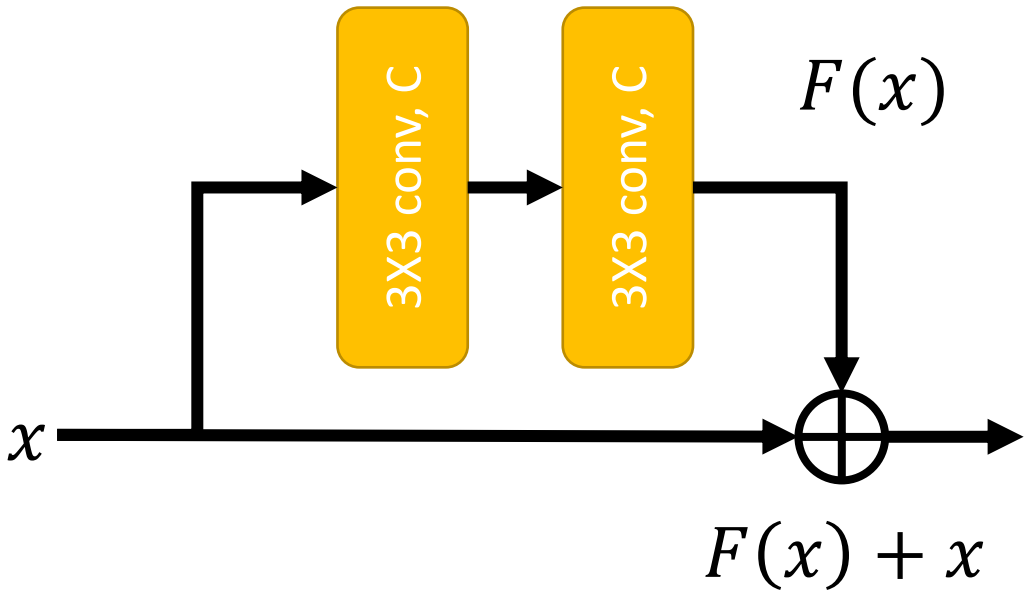


Residual Building Block



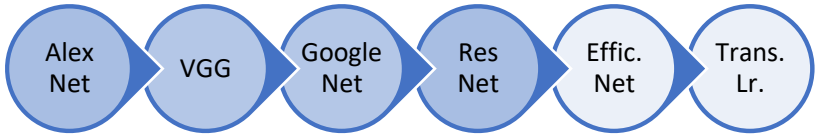


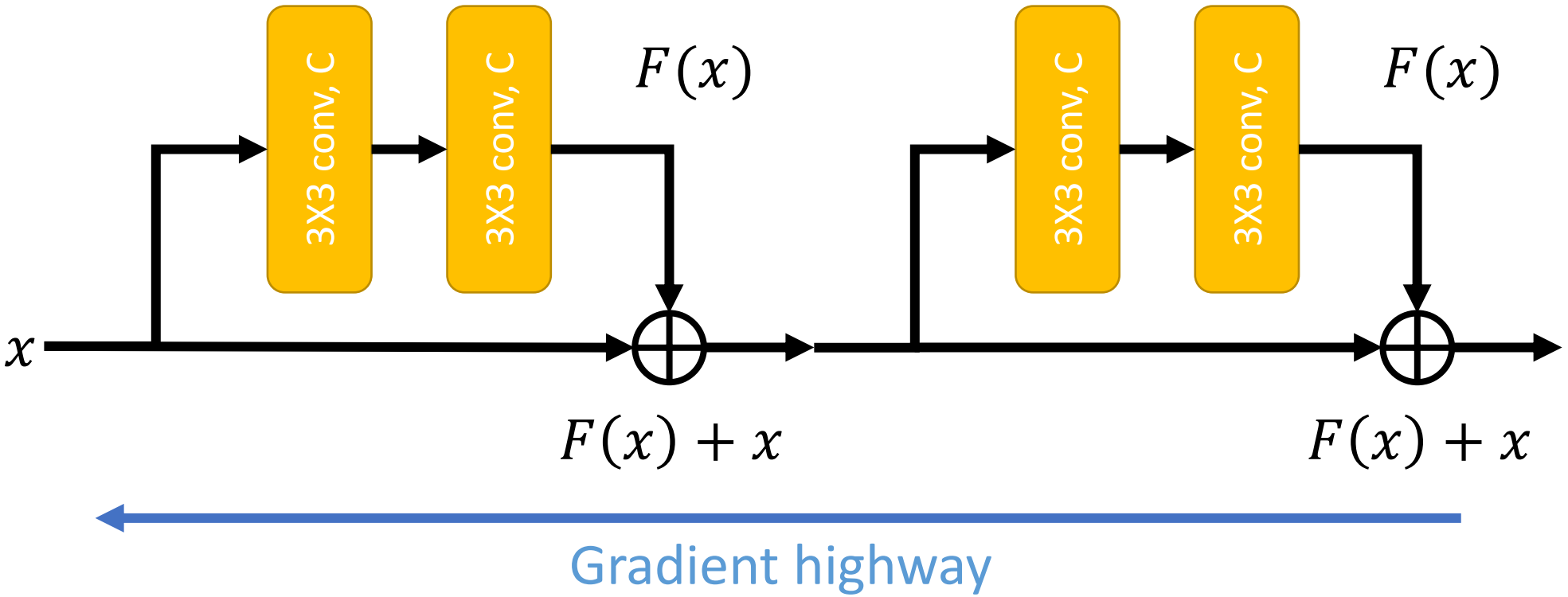
Same blocks



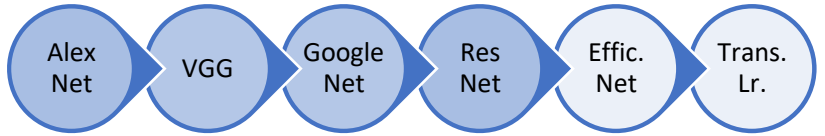
Encourage identity mapping

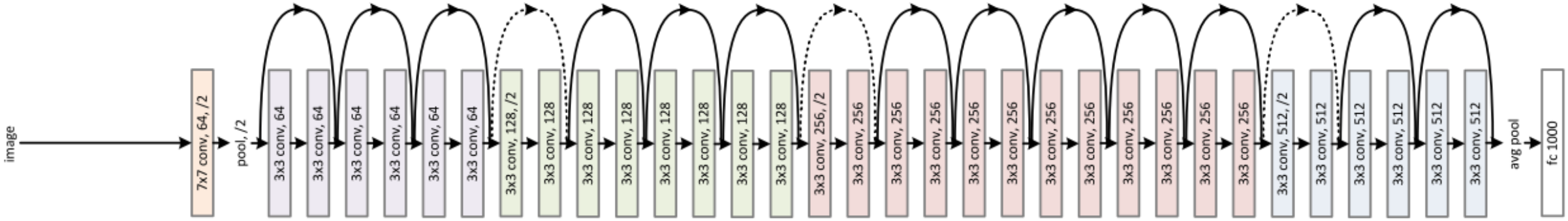
ResNet Gradient propagation



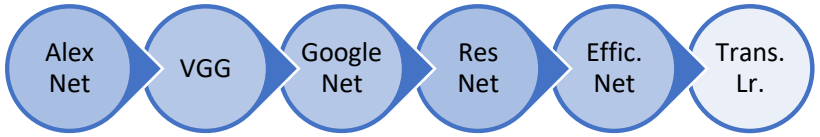


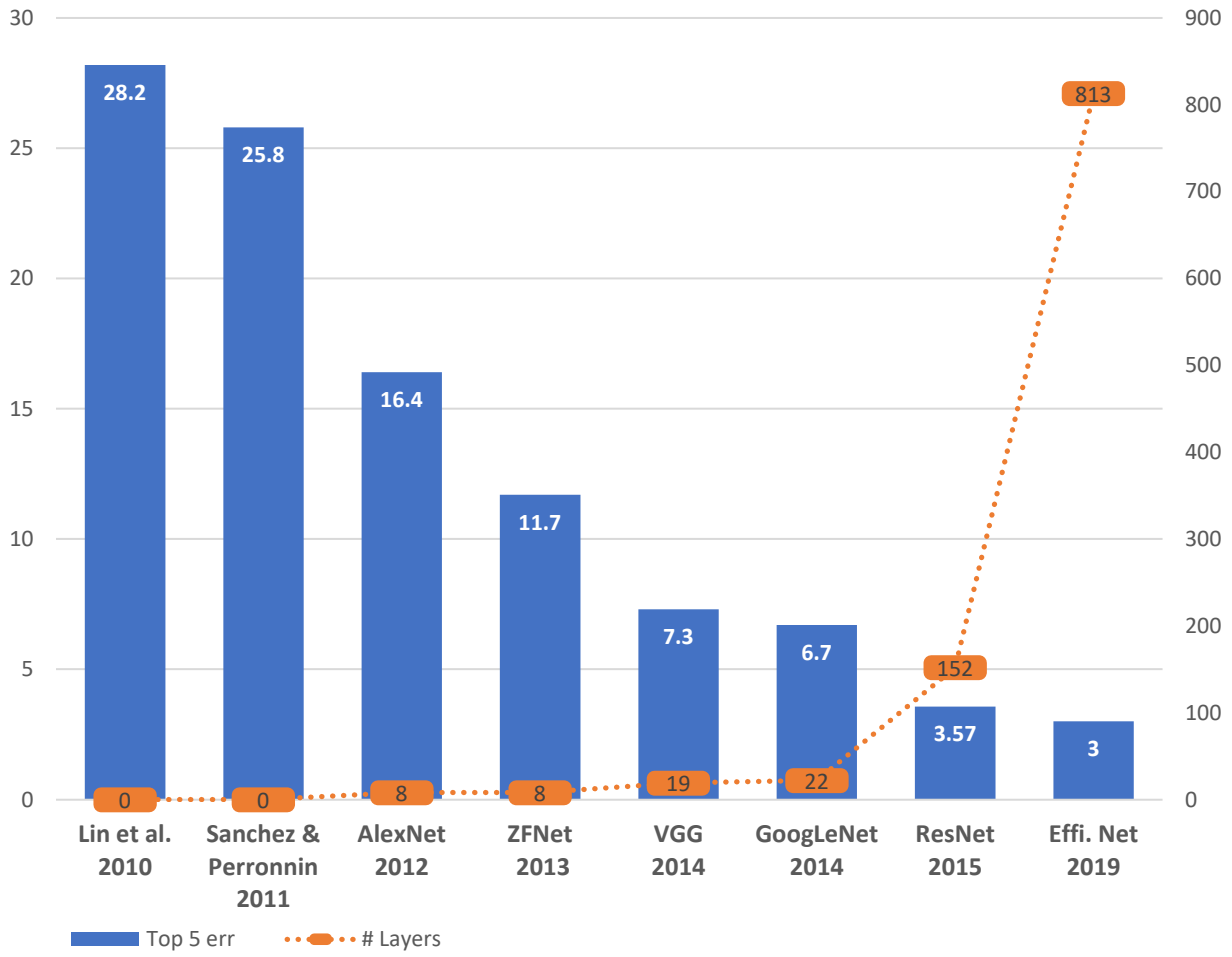
ResNet Architecture





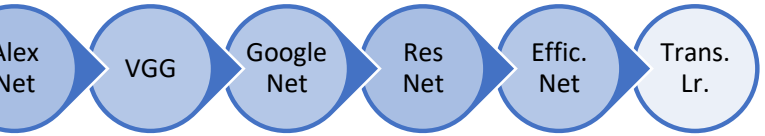
EfficientNet

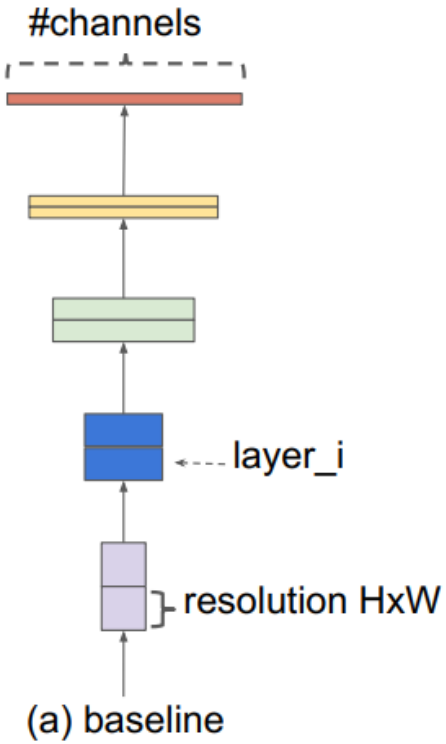




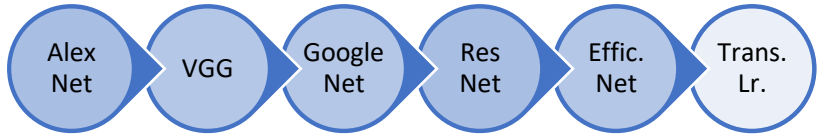
Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. ICML, 2019.

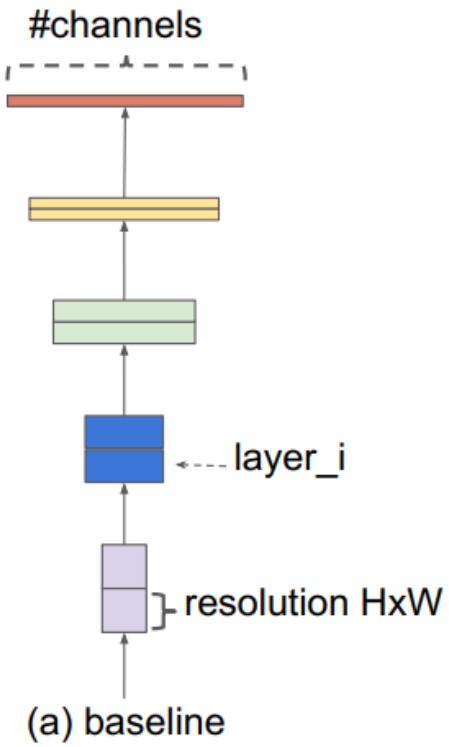
EfficientNet

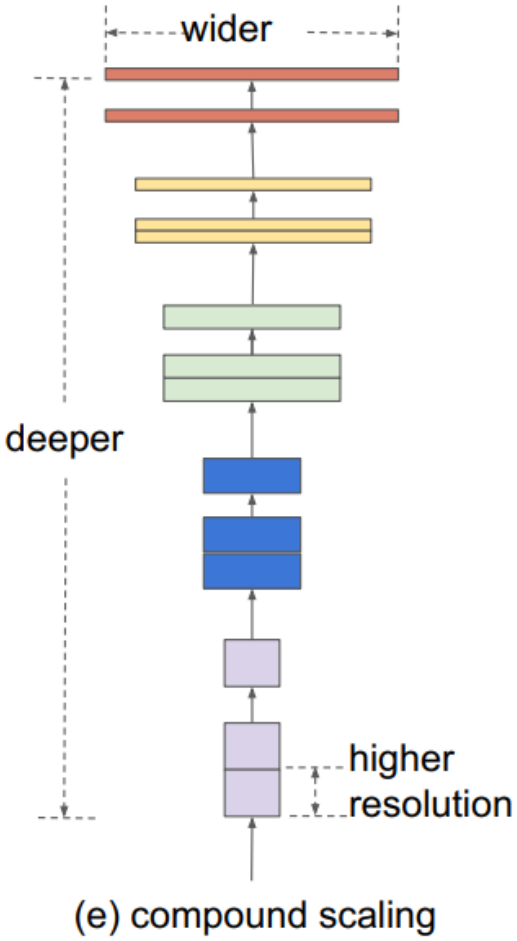




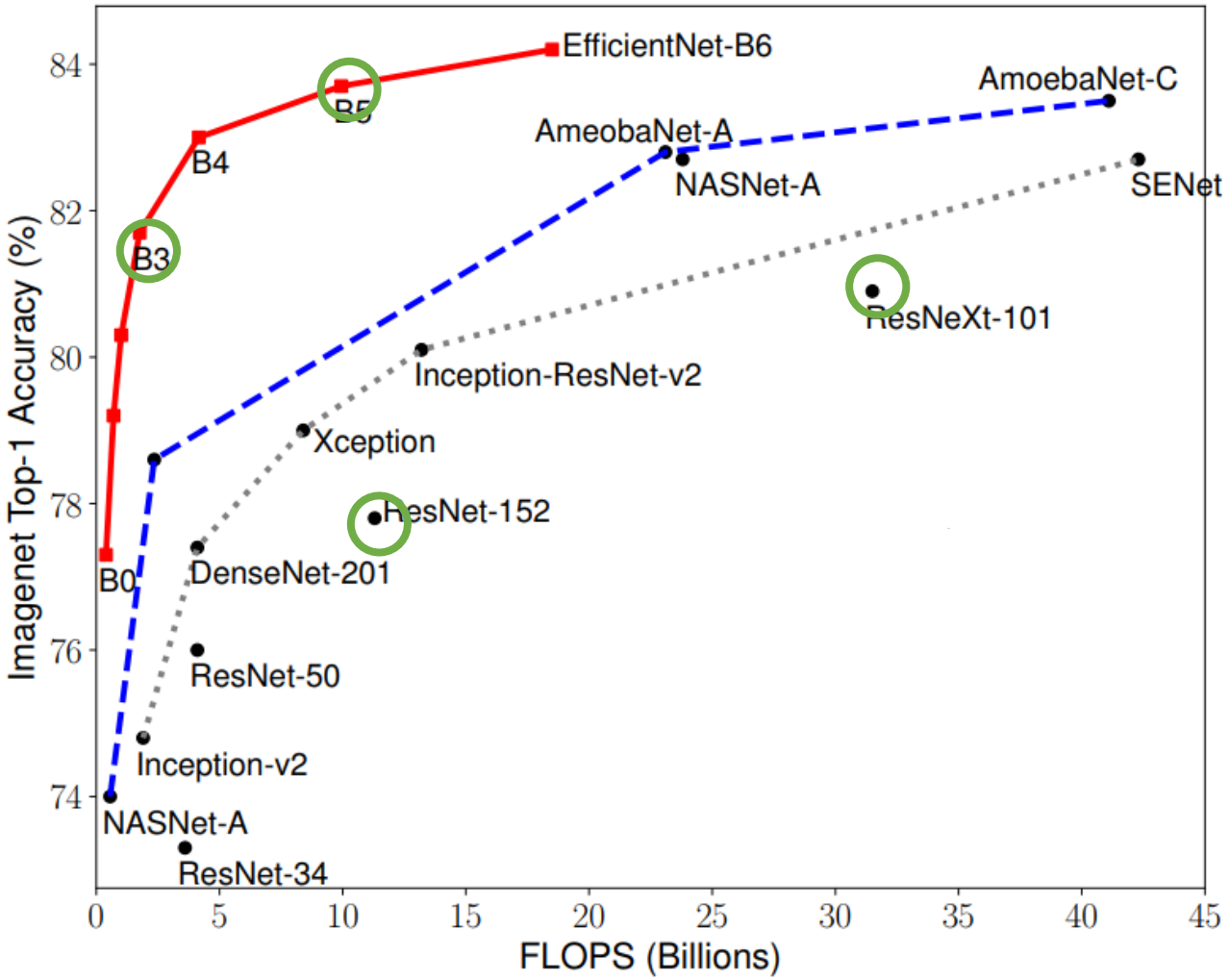
EfficientNet



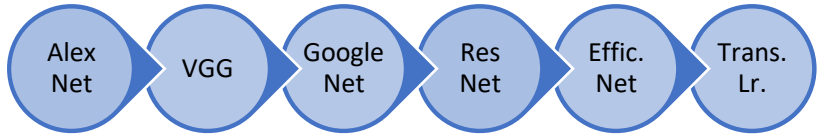


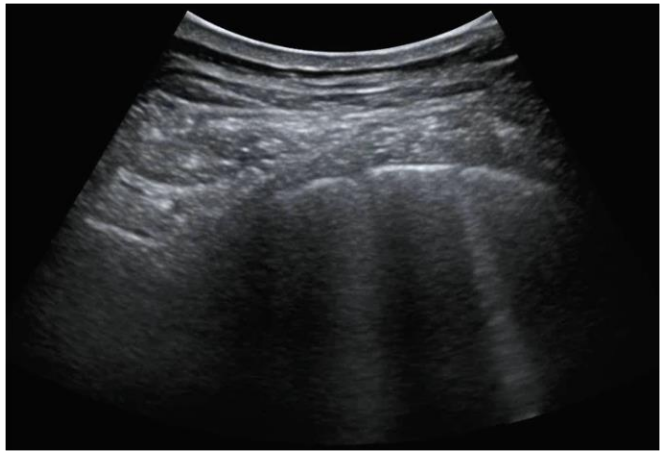


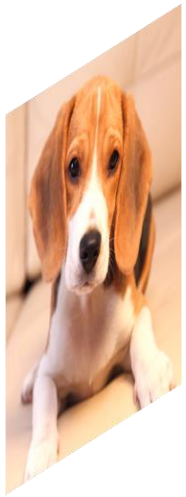
Results

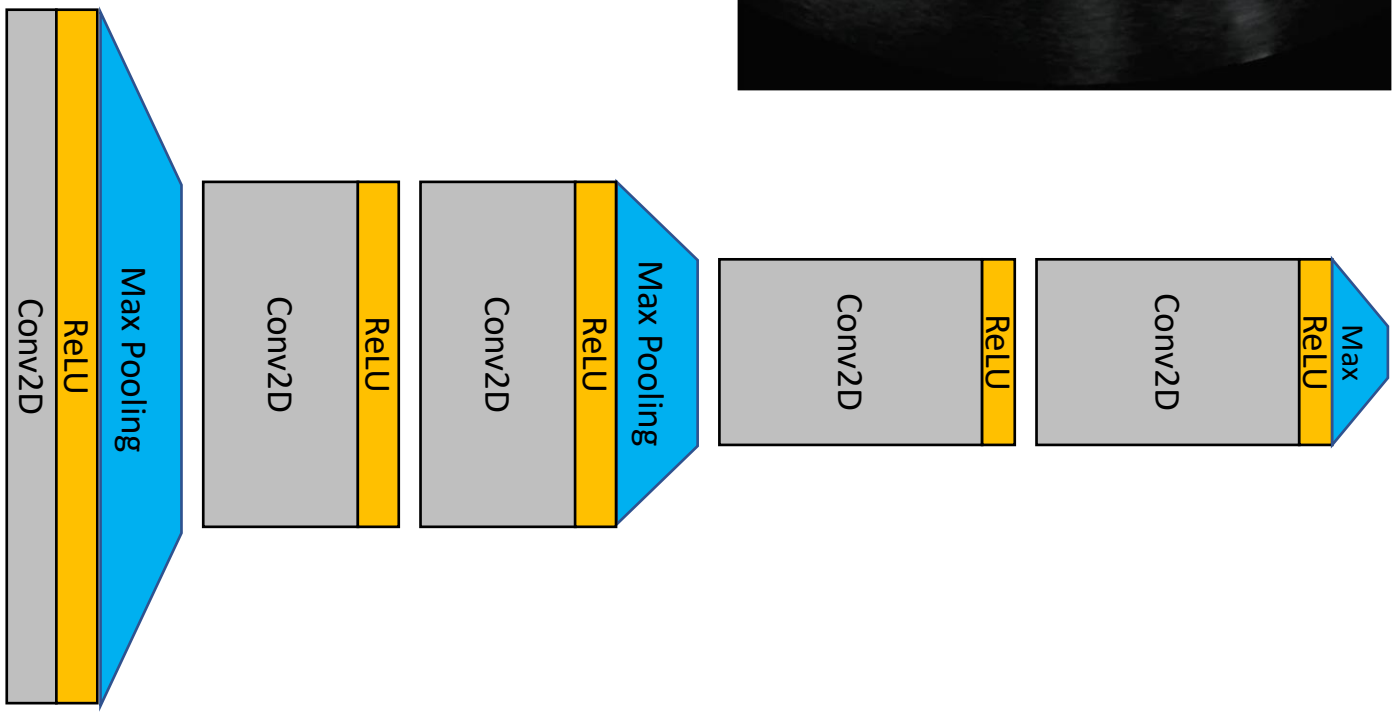

EfficientNet-B6
AmoebaNet-C
AmeobaNet-A
B4
NASNet-A
SENet
B3
ResNeXt-101
Inception-ResNet-v2
Xception
ResNet-152
B0
DenseNet-201
ResNet-50
Inception-v2
NASNet-A
ResNet-34
Imagenet Top-1 Accuracy (%)
FLOPS (Billions)
84
82
80
78
76
74
0
5
10
15
20
25
30
35
40
45

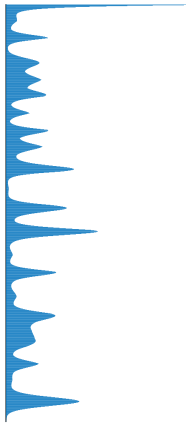
Transfer Learning





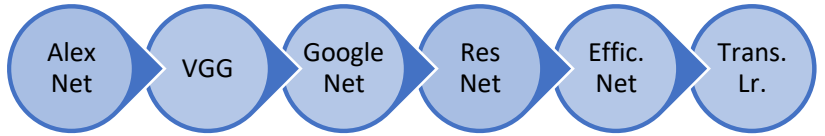


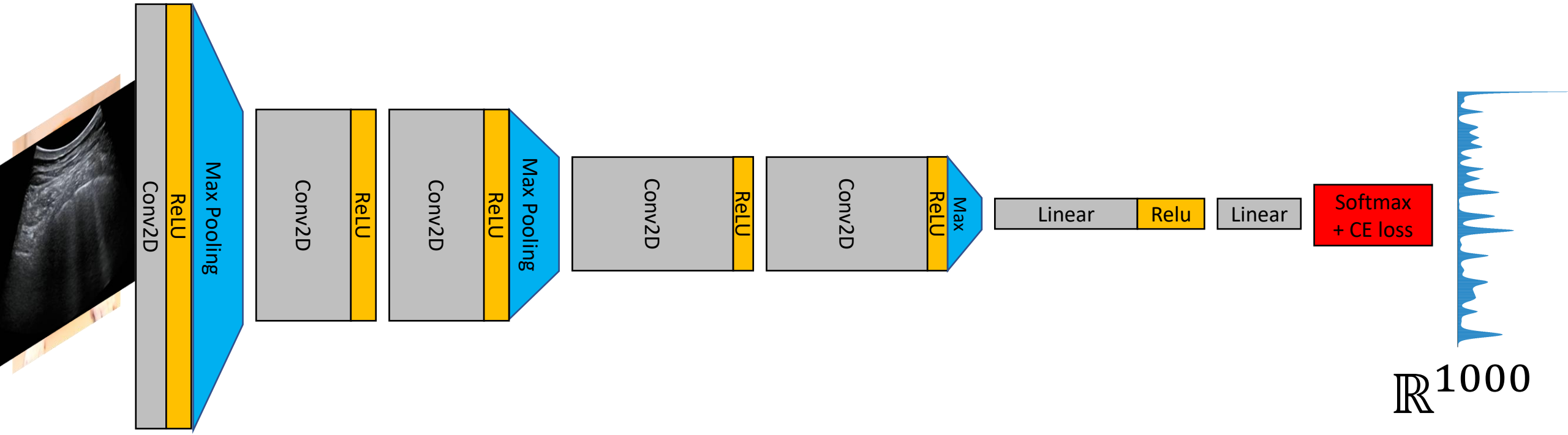




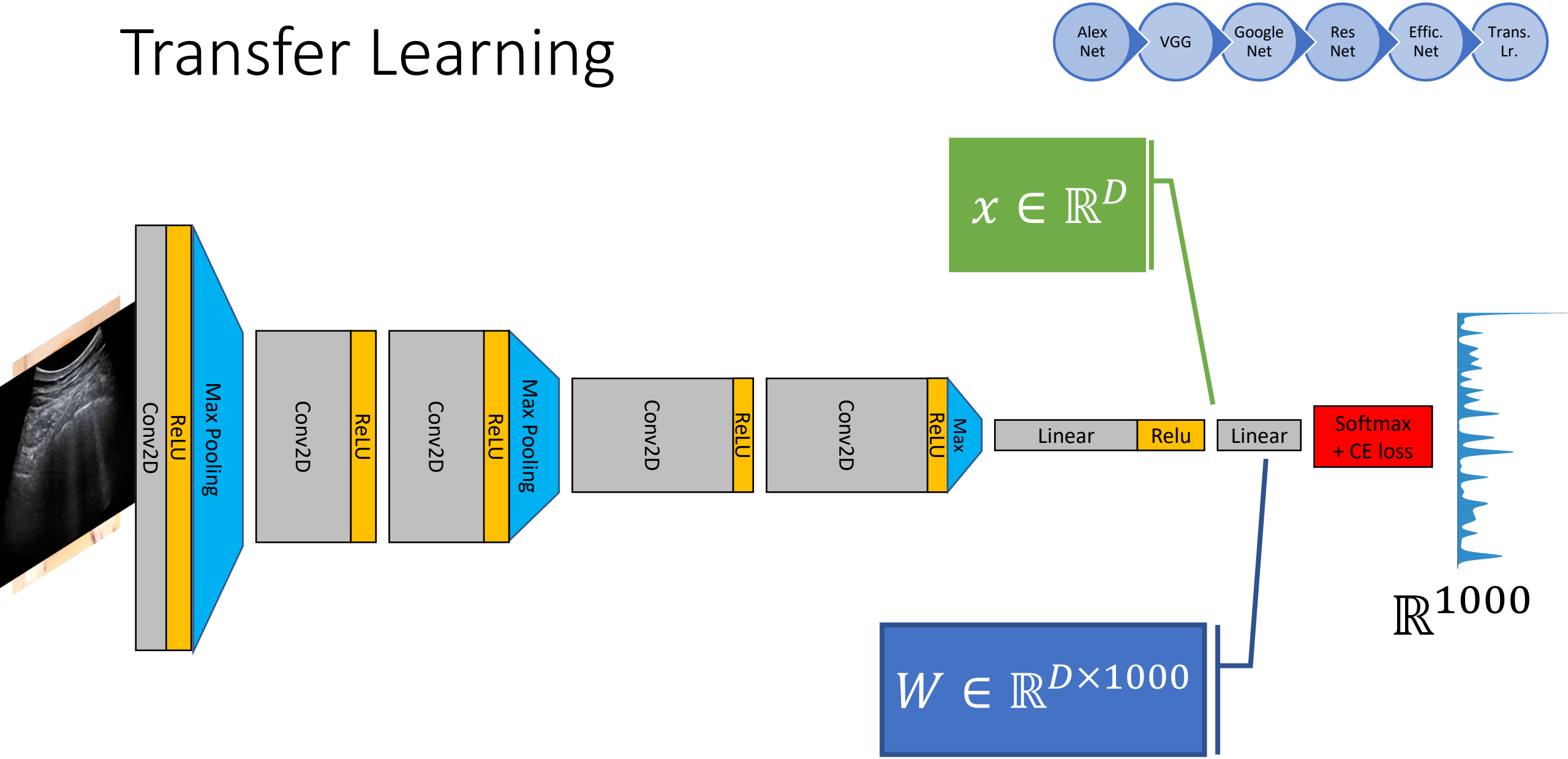
$\mathbb{R}^{1000}$

Transfer Learning

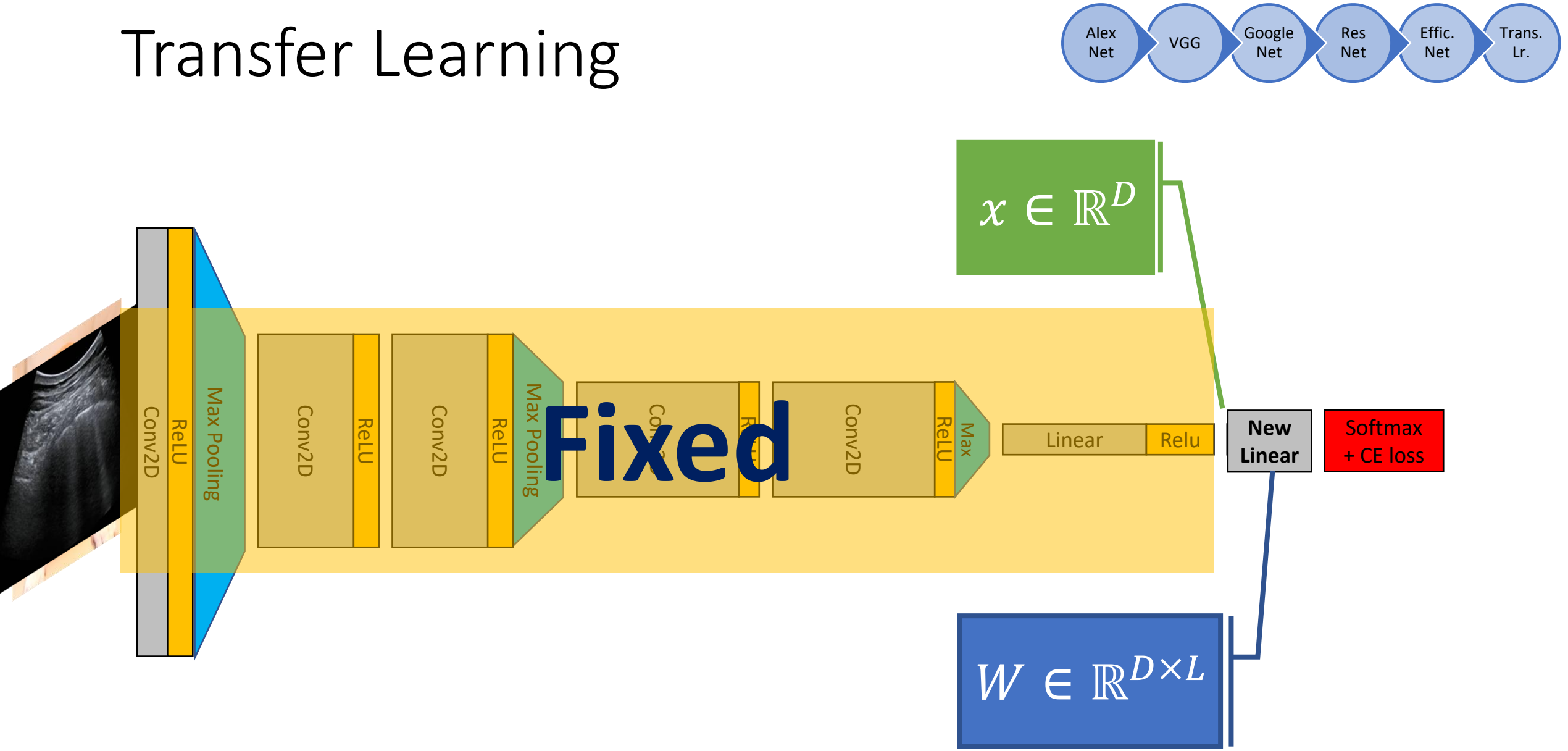




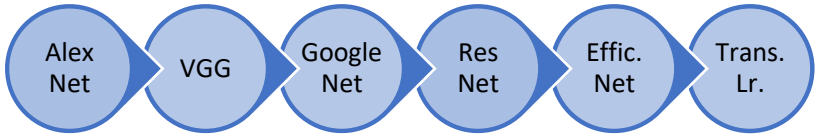
Transfer Learning



Transfer Learning



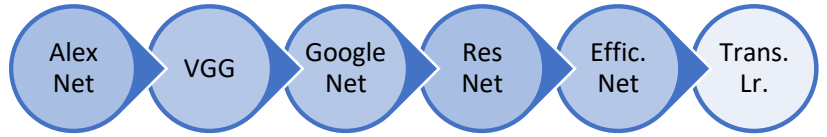
Transfer Learning



- In practice:

```python
from torchvision.prototype.models import resnet50, ResNet50_Weights

# Legacy weights with accuracy 76.130%
model = resnet50(weights=ResNet50_Weights.IMAGENET1K_V1)

# New weights with accuracy 80.858%
model = resnet50(weights=ResNet50_Weights.IMAGENET1K_V2)

# Best available weights (currently alias for IMAGENET1K_V2)
model = resnet50(weights=ResNet50_Weights.DEFAULT)

# No weights - random initialization
model = resnet50(weights=None)
```

Architecture summary



- Design your network according to your task and resources
- Take into consideration
  - # Parameters
  - # FLOPs
  - Memory Size
- Use identity mapping (Skip connections)
- Use **existing architectures** when possible
- Use **pre-trained models** when possible

Questions?