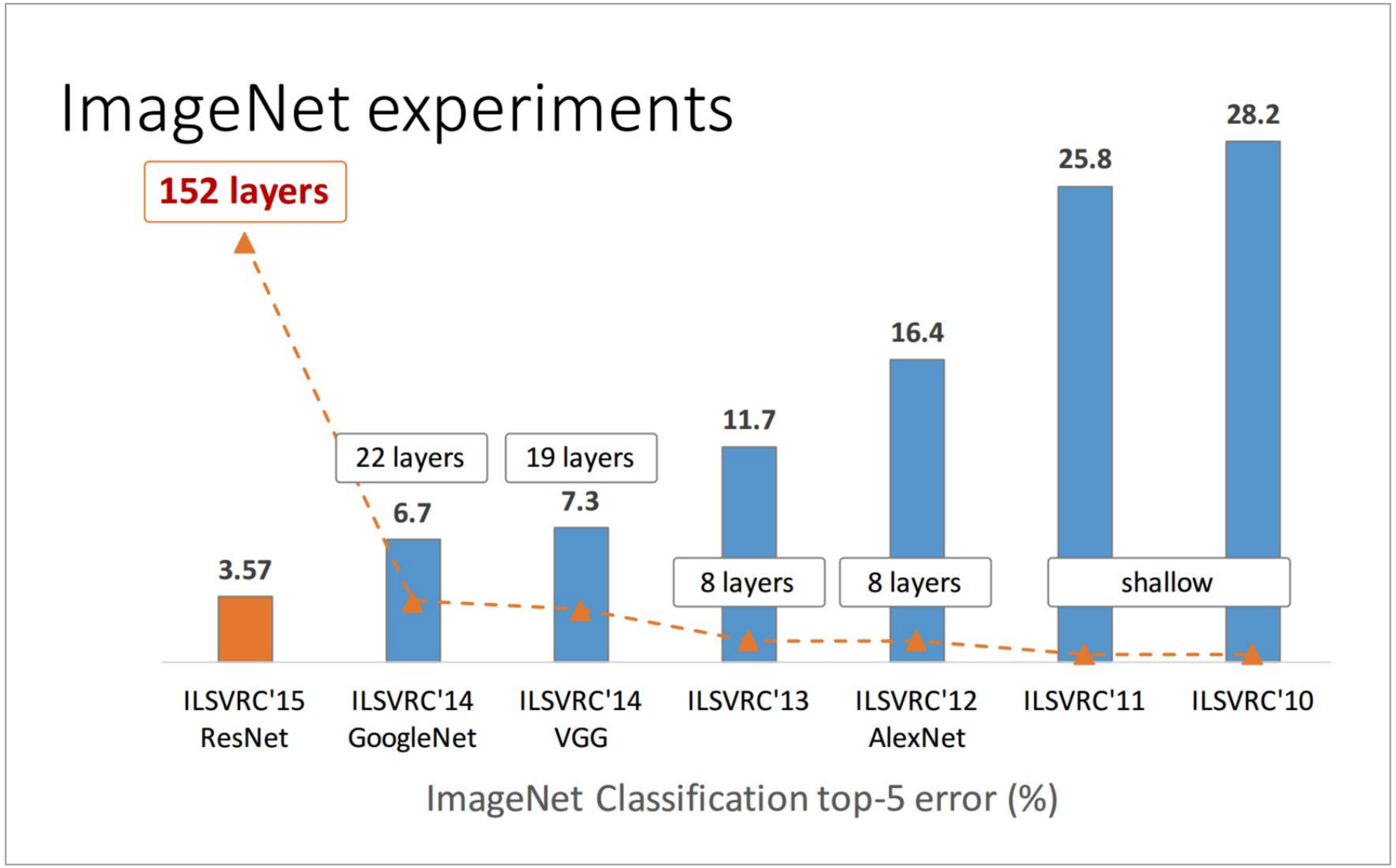
# Modern CNNs

Industrial AI Lab.

Prof. Seungchul Lee

# ImageNet

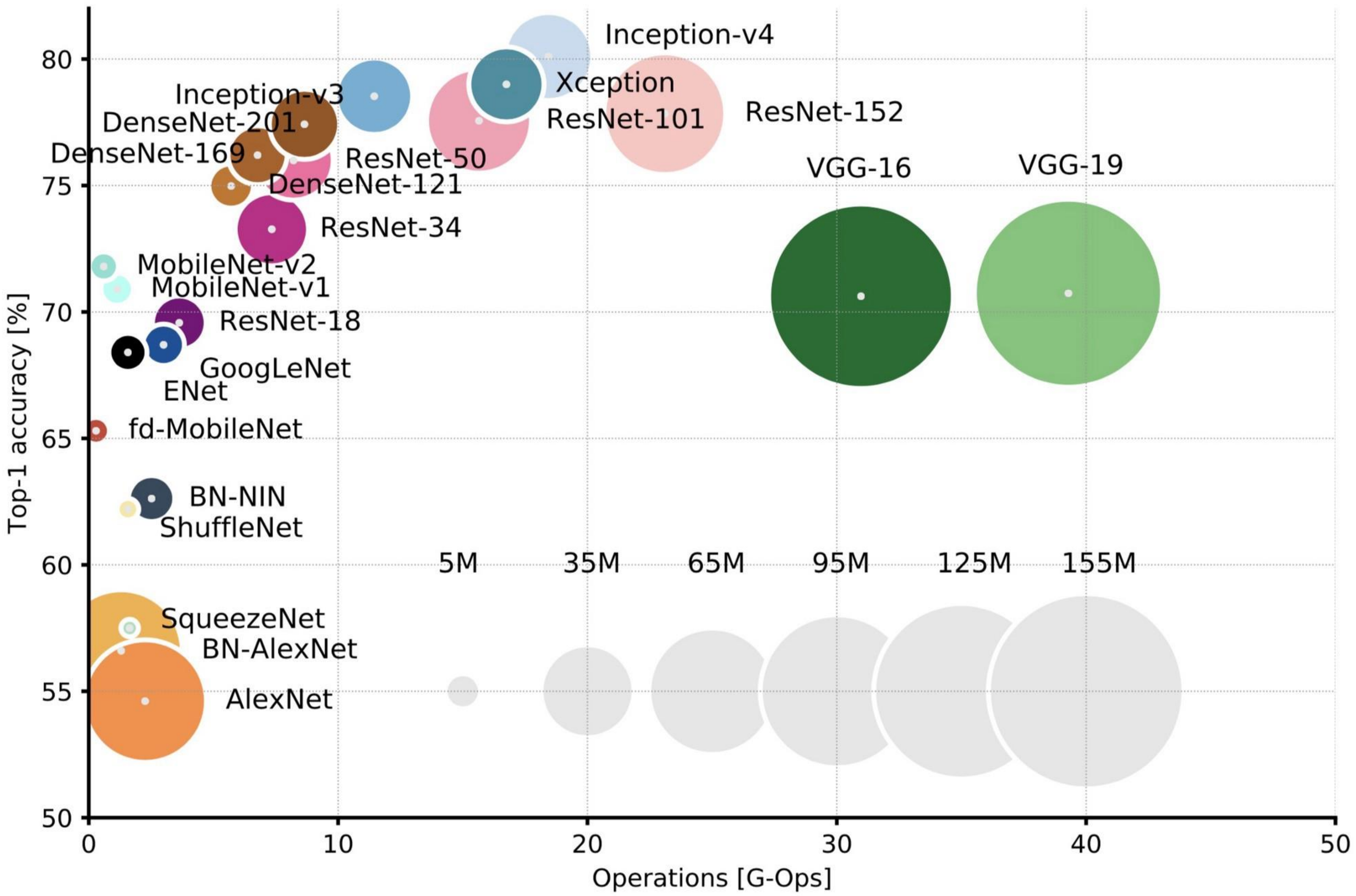
- Human performance = 5.1 %




POSTECH

from Kaiming He slides "Deep residual learning for image recognition." ICML. 2016.

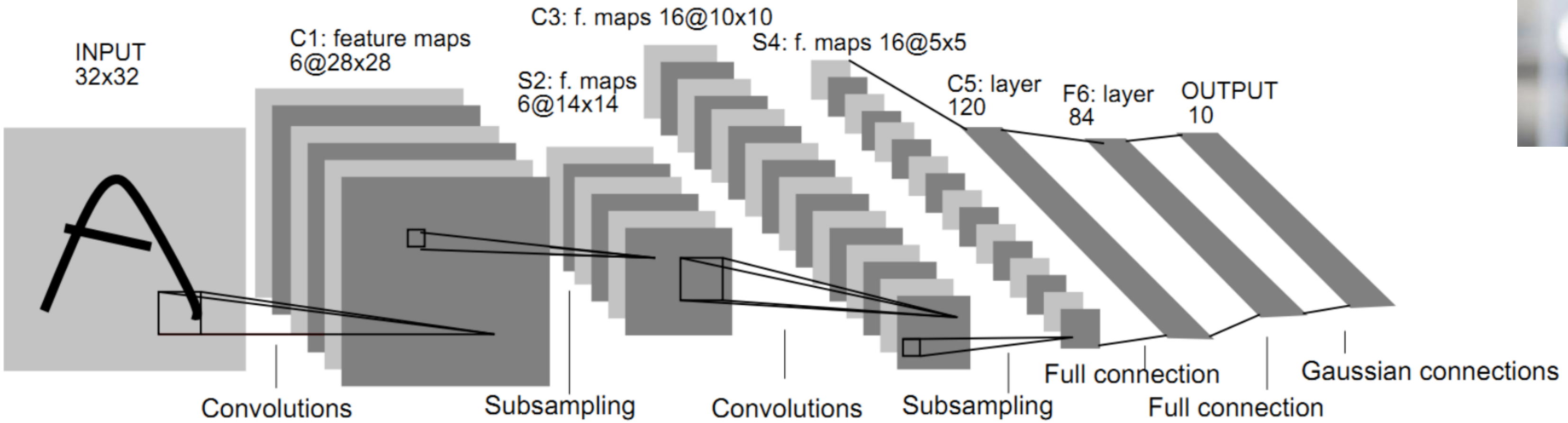
# ImageNet



Inception-v4
Xception
Inception-v3
DenseNet-201
ResNet-101
ResNet-152
DenseNet-169
ResNet-50
VGG-16
VGG-19
DenseNet-121
ResNet-34
MobileNet-v2
MobileNet-v1
ResNet-18
GoogLeNet
ENet
fd-MobileNet
BN-NIN
ShuffleNet
5M
35M
65M
95M
125M
155M
SqueezeNet
BN-AlexNet
AlexNet
Top-1 accuracy [%]
80
75
70
65
60
55
50
0
10
20
30
40
50
Operations [G-Ops]

# LeNet

- CNN = Convolutional Neural Networks = ConvNet
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition.
- All are still the basic components of modern ConvNets!

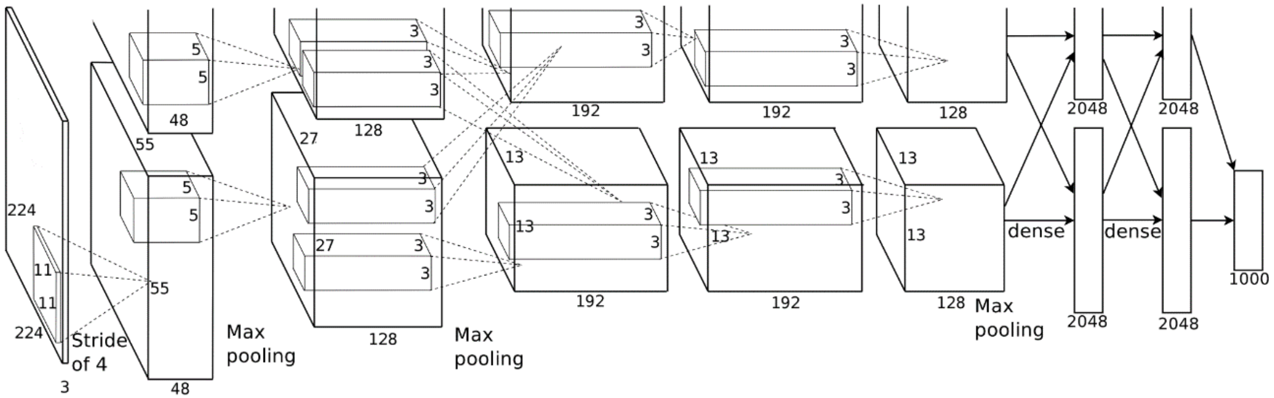



Yann LeCun

# AlexNet

- Simplified version of Krizhevsky, Alex, Sutskever, and Hinton. "Imagenet classification with deep convolutional neural networks." NIPS 2012

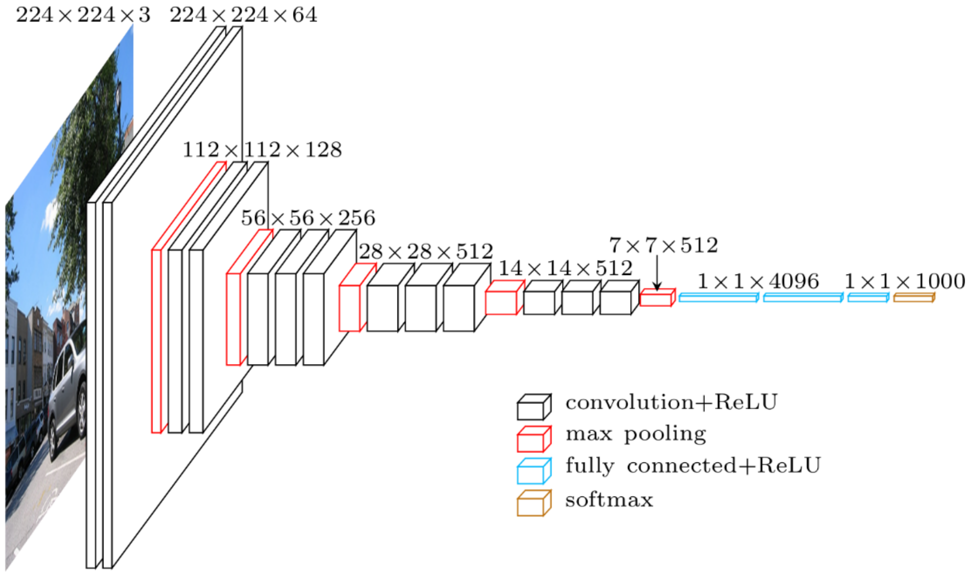
- LeNet-style backbone, plus:
  - ReLU [Nair & Hinton 2010]
    - “RevoLUtion of deep learning”*
    - Accelerate training; better grad prop (vs. tanh)
  - Dropout [Hinton et al 2012]
    - In-network ensembling
    - Reduce overfitting
  - Data augmentation
    - Label-preserving transformation
    - Reduce overfitting

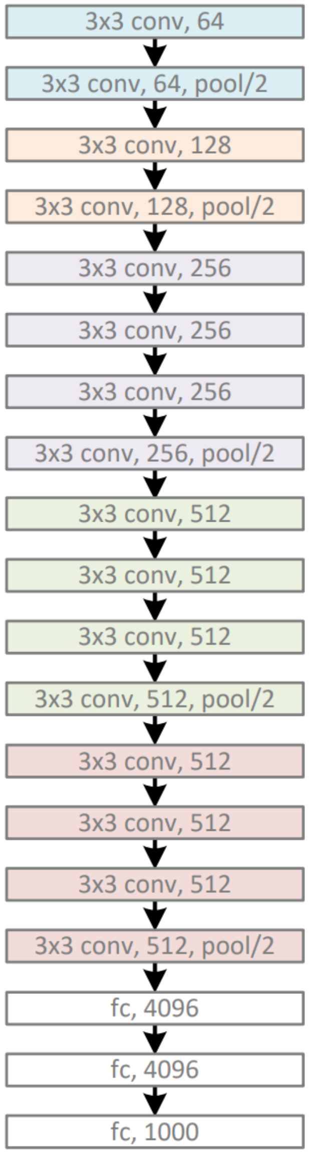


# VGG-16/19

- Simonyan, Karen, and Zisserman. "Very deep convolutional networks for large-scale image recognition." (2014)

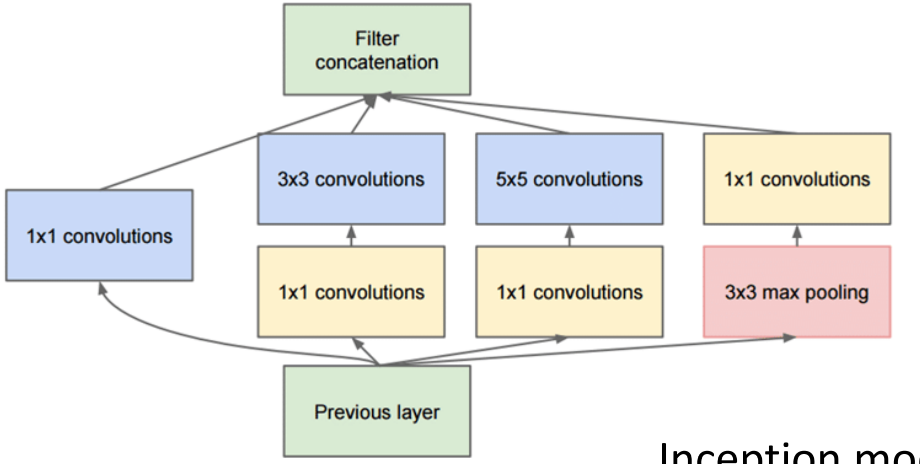
- Simply “Very Deep”!
  - Modularized design
    - 3x3 Conv as the module
    - Stack the same module
    - Same computation for each module
  - Stage-wise training
    - VGG-11 → VGG-13 → VGG-16
    - We need a better initialization…



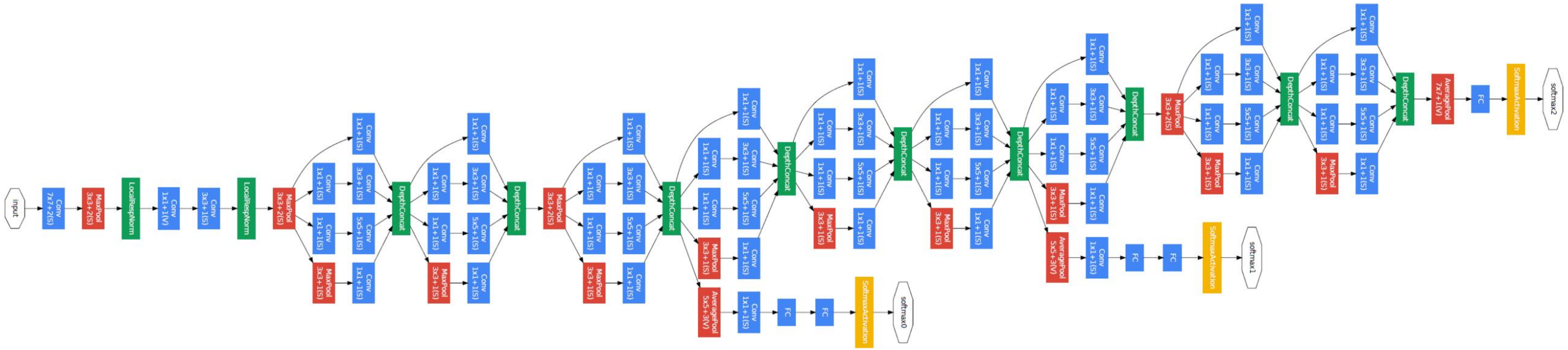



# GoogleNet/Inception

- Multiple branches
  – e.g., 1x1, 3x3, 5x5, pool
- Shortcuts
  – stand-alone 1x1, merged by concat.
- Bottleneck
  – Reduce dim by 1x1 before expensive 3x3/5x5 conv

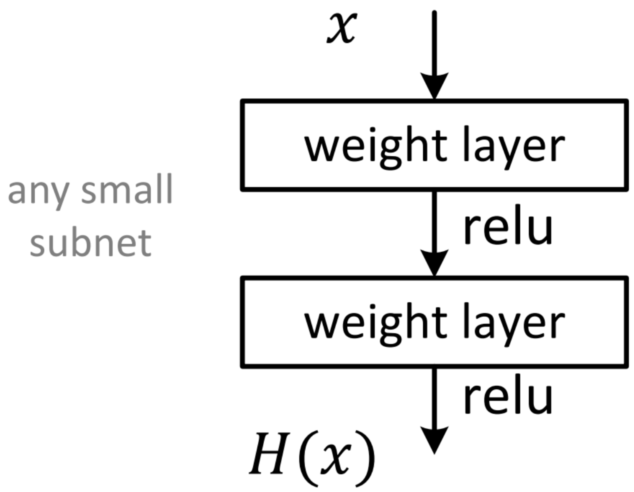



Inception module



# ResNet (Deep Residual Learning)

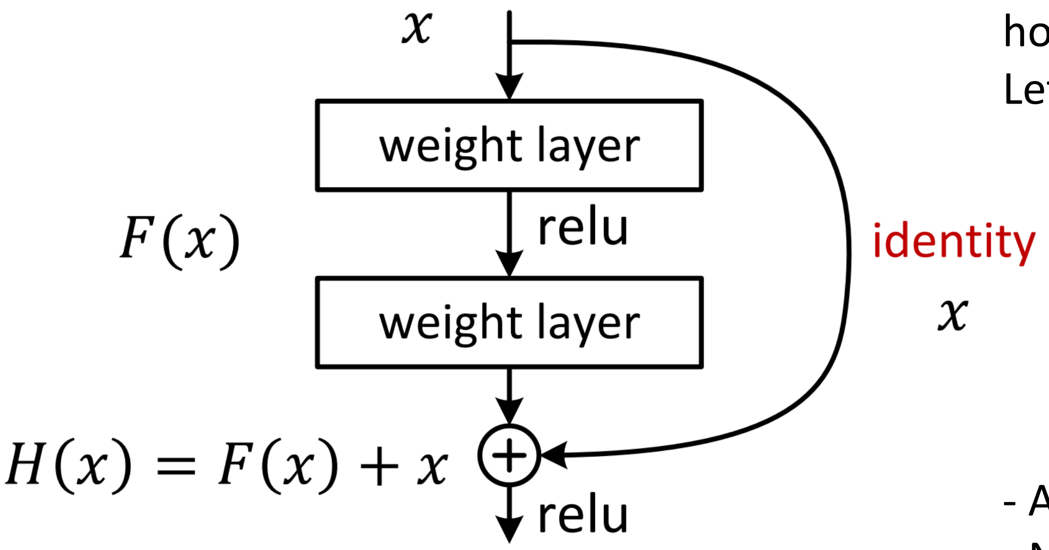
- He, Kaiming, et al. "Deep residual learning for image recognition." CVPR. 2016.
- Plane net




$H(x)$ is any desired mapping,
hope the small subnet fit $H(x)$

# ResNet (Deep Residual Learning)

- He, Kaiming, et al. "Deep residual learning for image recognition." CVPR. 2016.
- Residual net
- Skip connection

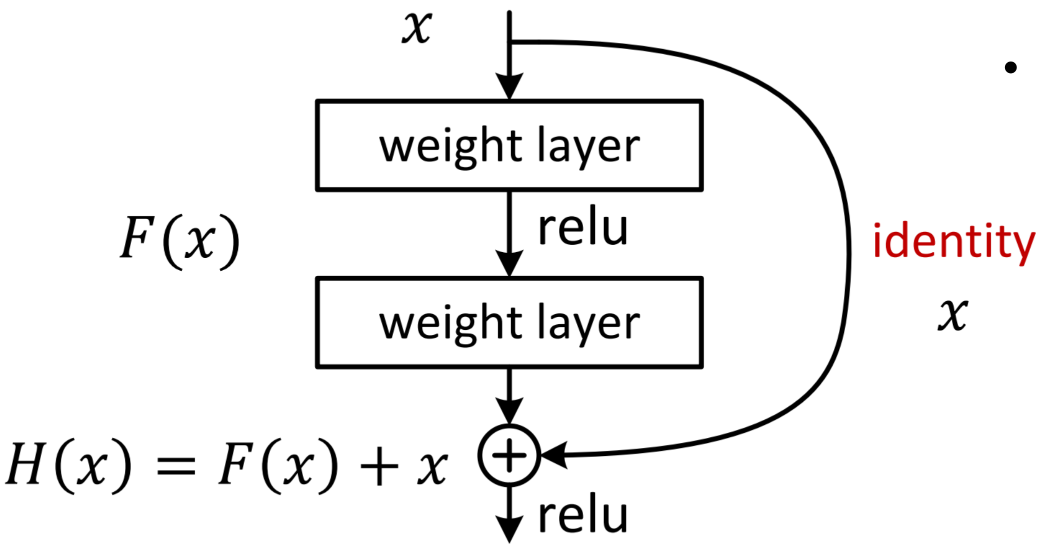



$H(x)$ is any desired mapping,
~~hope the small subnet fit $H(x)$~~
hope the small subnet fit $F(x)$
Let $H(x) = F(x) + x$

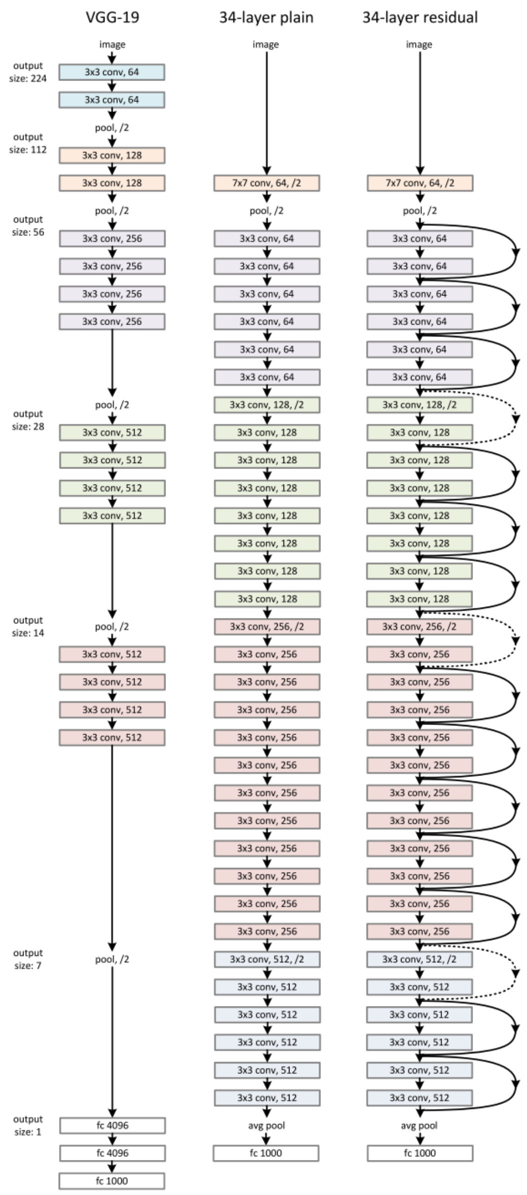
- A direct connection between 2 non-consecutive layers
- No vanishing gradient

# ResNet (Deep Residual Learning)

- Parameters are optimized to learn a residual, that is the difference between the value before the block and the one needed after.
- $F(x)$ is a residual mapping w.r.t. identity




- If identity were optimal, easy to set weights as 0
- If optimal mapping is closer to identity, easier to find small fluctuations

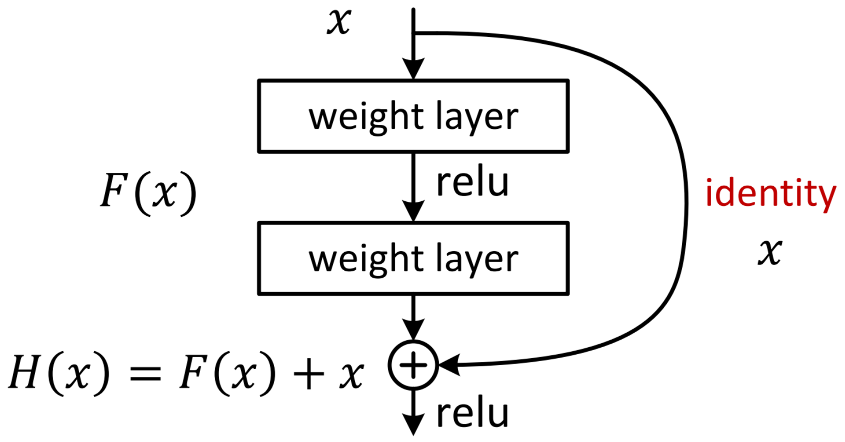


# Skip Connection

- A skip connection is a connection that bypasses at least one layer.
- Here, it is often used to transfer local information by concatenating or summing feature maps from the downsampling path with feature maps from the upsampling path.
- Merging features from various resolution levels helps combining context information with spatial information.

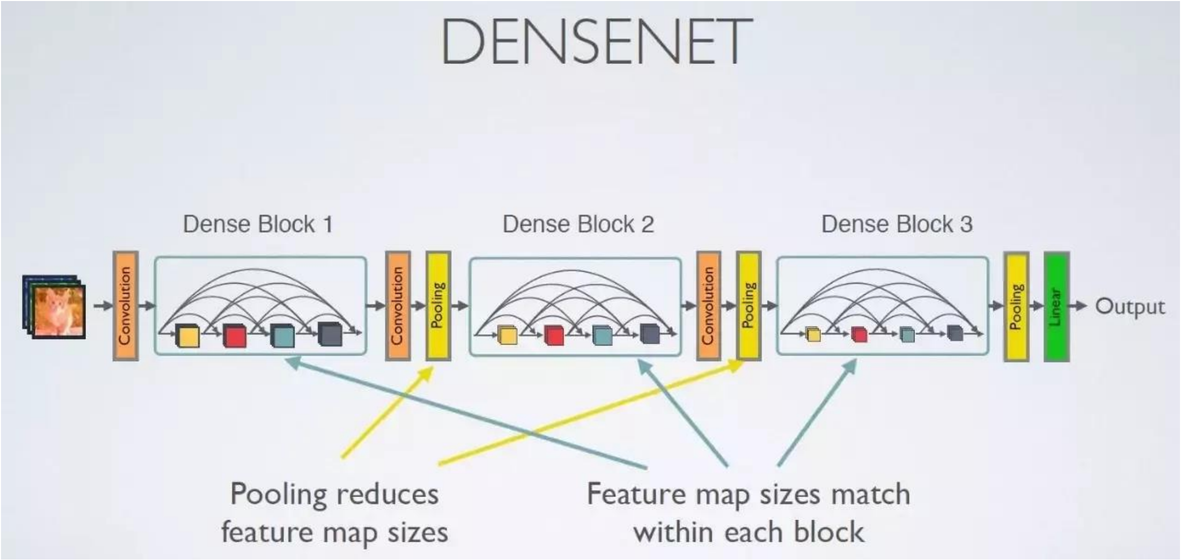
# Residual Net

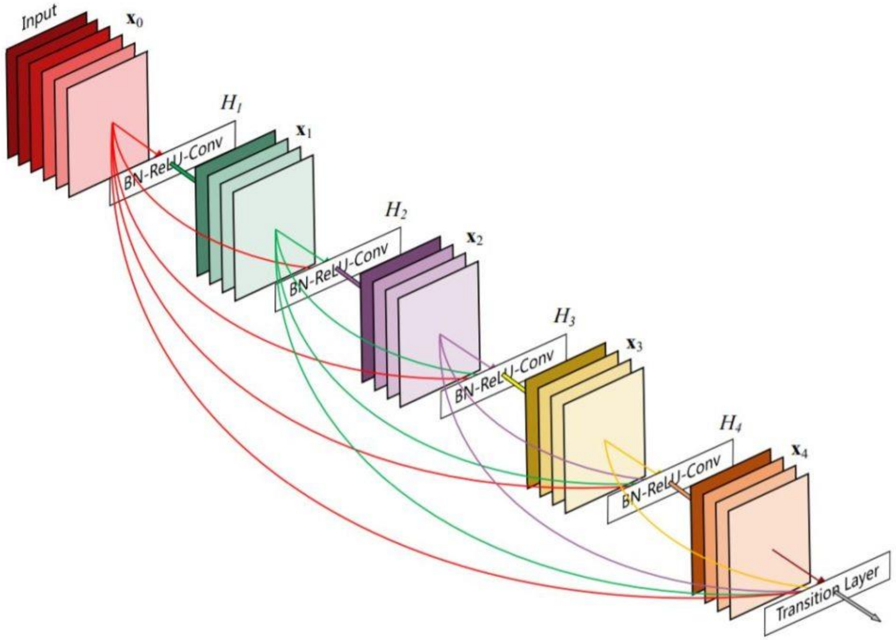
```python
def residual_net(x):
    conv1 = tf.layers.conv2d(inputs = x,
                             filters = 32,
                             kernel_size = [3, 3],
                             padding = "SAME",
                             activation = tf.nn.relu)
    conv2 = tf.layers.conv2d(inputs = conv1,
                             filters = 32,
                             kernel_size = [3, 3],
                             padding = "SAME",
                             activation = tf.nn.relu)

    maxp2 = tf.layers.max_pooling2d(inputs = x + conv2,
                                    pool_size = [2, 2],
                                    strides = 2)

    flat = tf.layers.flatten(maxp2)
    hidden = tf.layers.dense(inputs = flat,
                             units = n_hidden,
                             activation = tf.nn.relu)
    output = tf.layers.dense(inputs = hidden,
                             units = n_output)

    return output
```



# DensNets

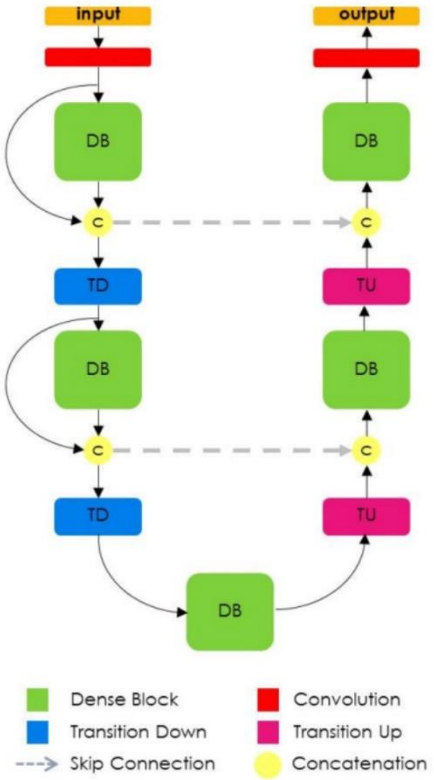
- Densely Connected Convolutional Networks

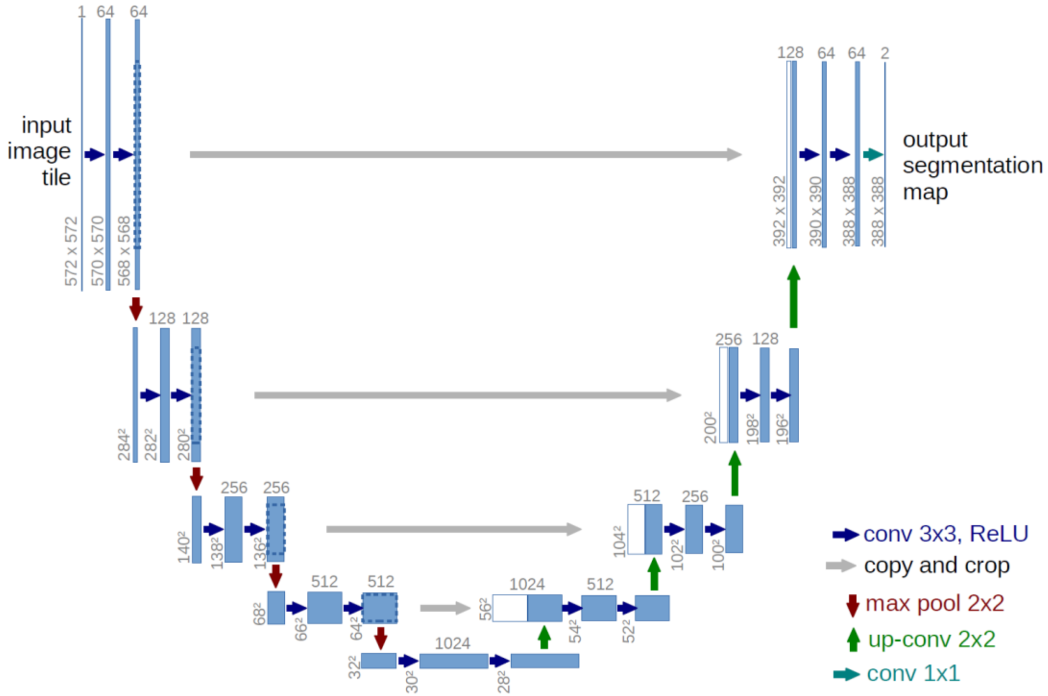



Huang, Gao, et al., "Densely connected convolutional networks"
Proceedings of the IEEE conference on computer vision and pattern recognition. Vol. 1. No. 2. 2017.

# U-Net




Ronneberger, Olaf; Fischer, Philipp; Brox, Thomas (2015),
“U-Net: Convolutional Networks for Biomedical Image Segmentation.” arXiv:1505.04597

# U-Net

- The U-Net owes its name to its symmetric shape

- The U-Net architecture is built upon the Fully Convolutional Network and modified in a way that it yields better segmentation in medical imaging.

- Compared to FCN-8, the two main differences are
  - U-net is symmetric and
  - the skip connections between the downsampling path and the upsampling path apply a concatenation operator instead of a sum.

- These skip connections intend to provide local information to the global information while upsampling. Because of its symmetry, the network has a large number of feature maps in the upsampling path, which allows to transfer information.