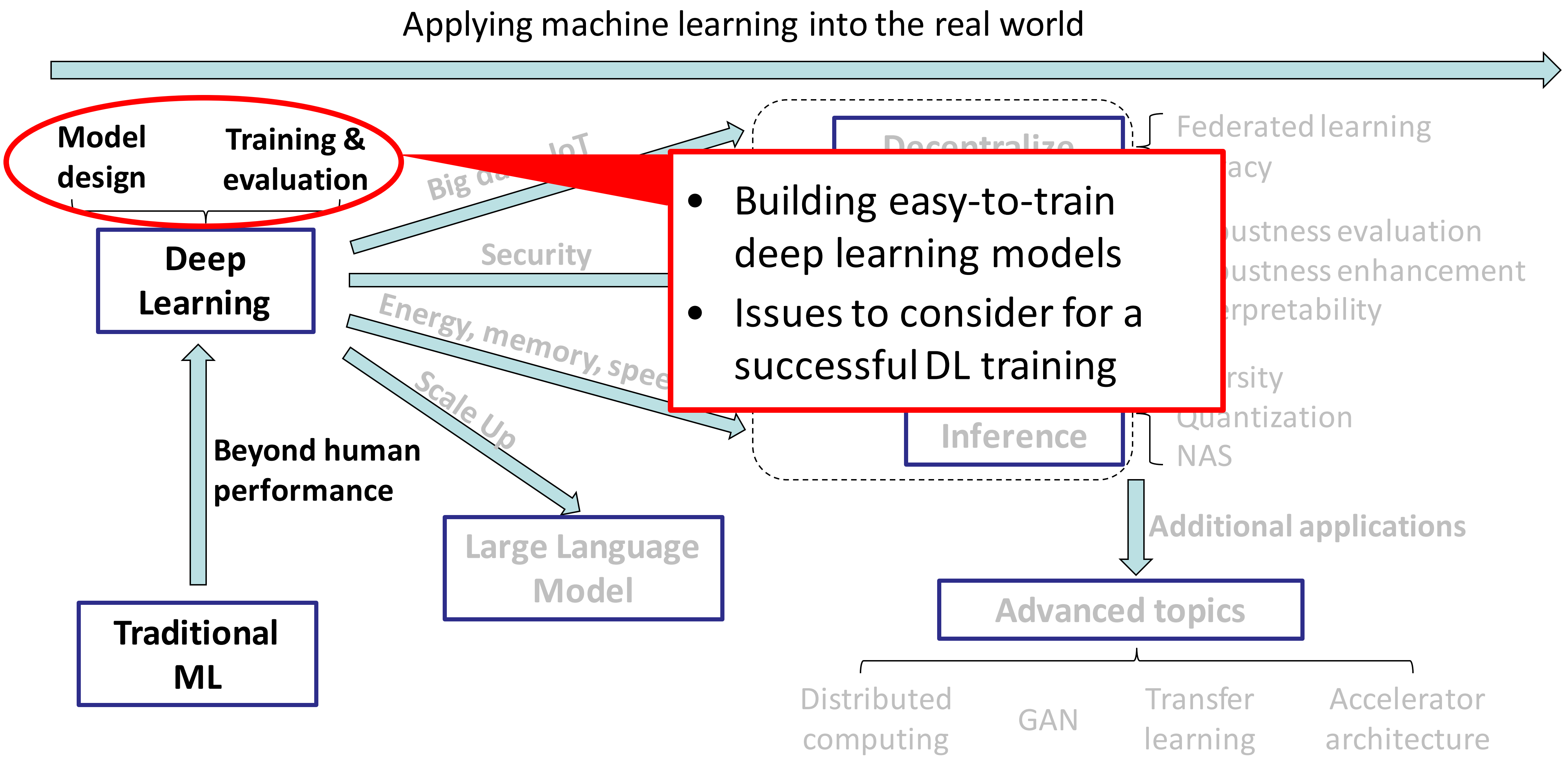
ECE 661 COMP ENG ML & DEEP NEURAL NETS

7. CNN ARCHITECTURES

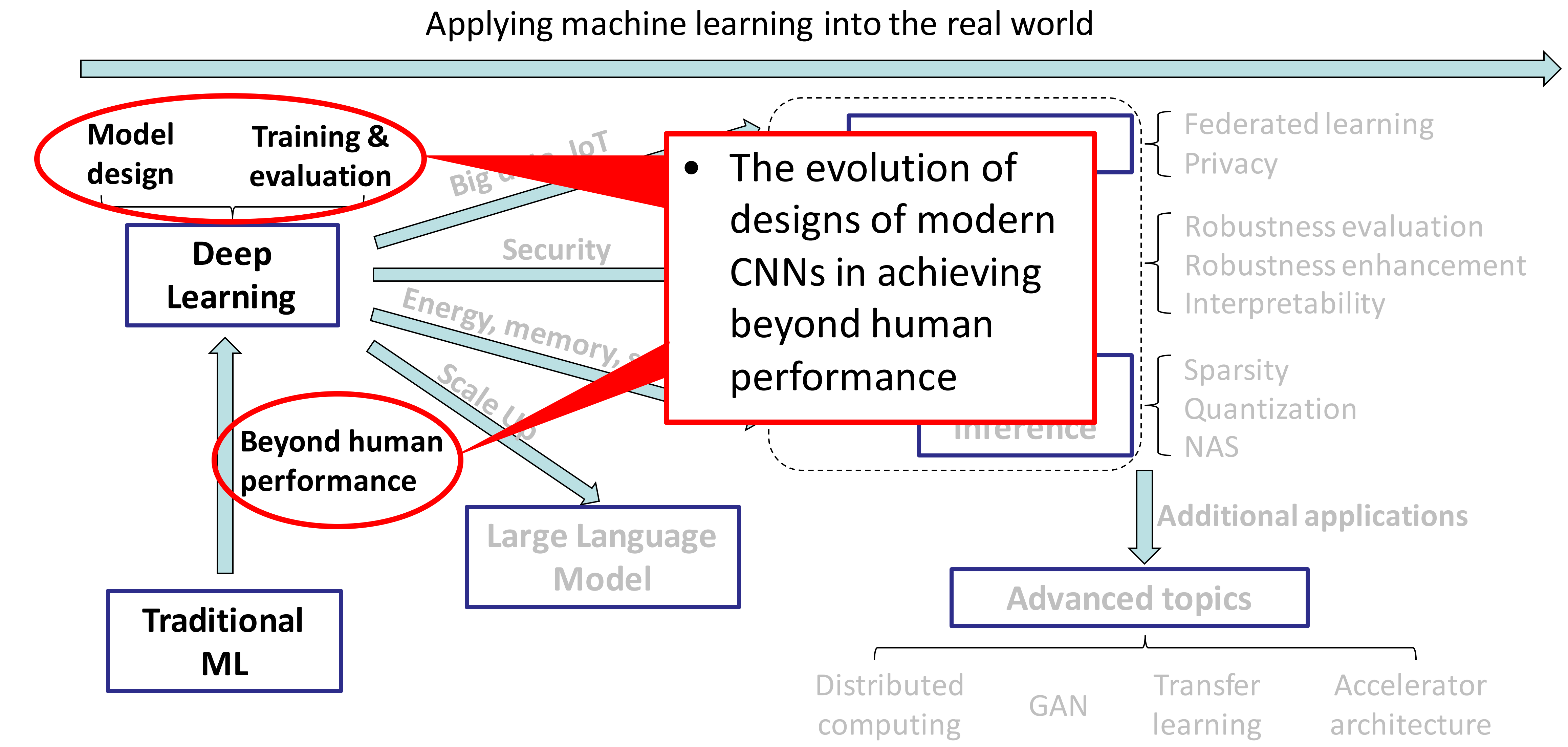
HAI "HELEN" LI , SPRING 2025

The previous lecture

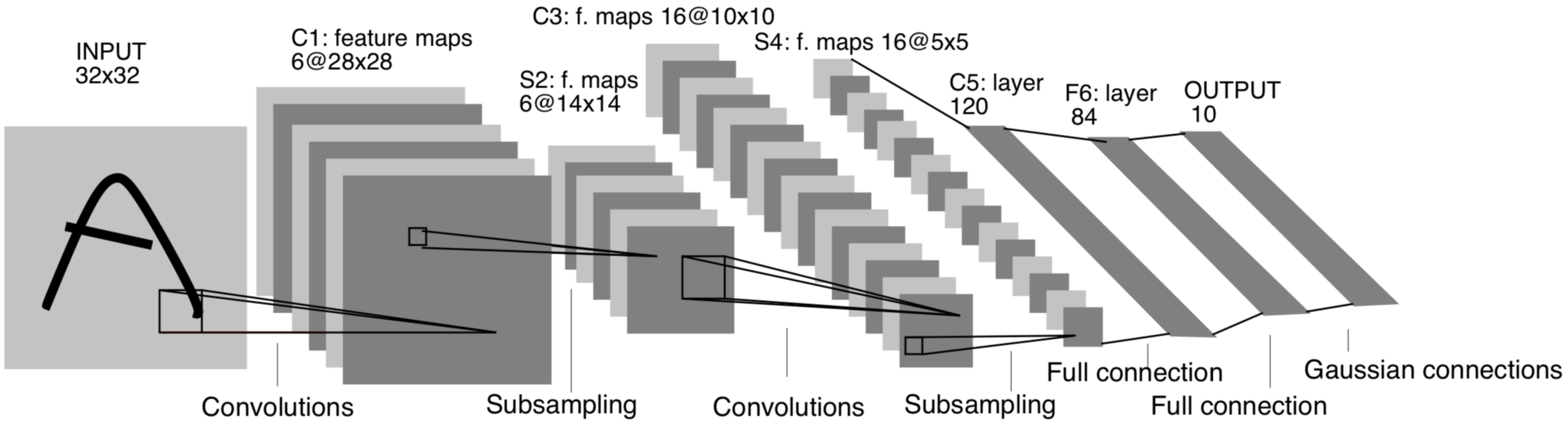


- Building easy-to-train deep learning models
- Issues to consider for a successful DL training

This lecture



Recap: LeNet-5



Features:

- LeNet-5 is the first one to stack CONV-POOL-CONV-POOL structures.
- LeNet-5 achieves good results on MNIST dataset.
- LeNet-5 uses tanh activation, which has serious gradient vanishing problem.

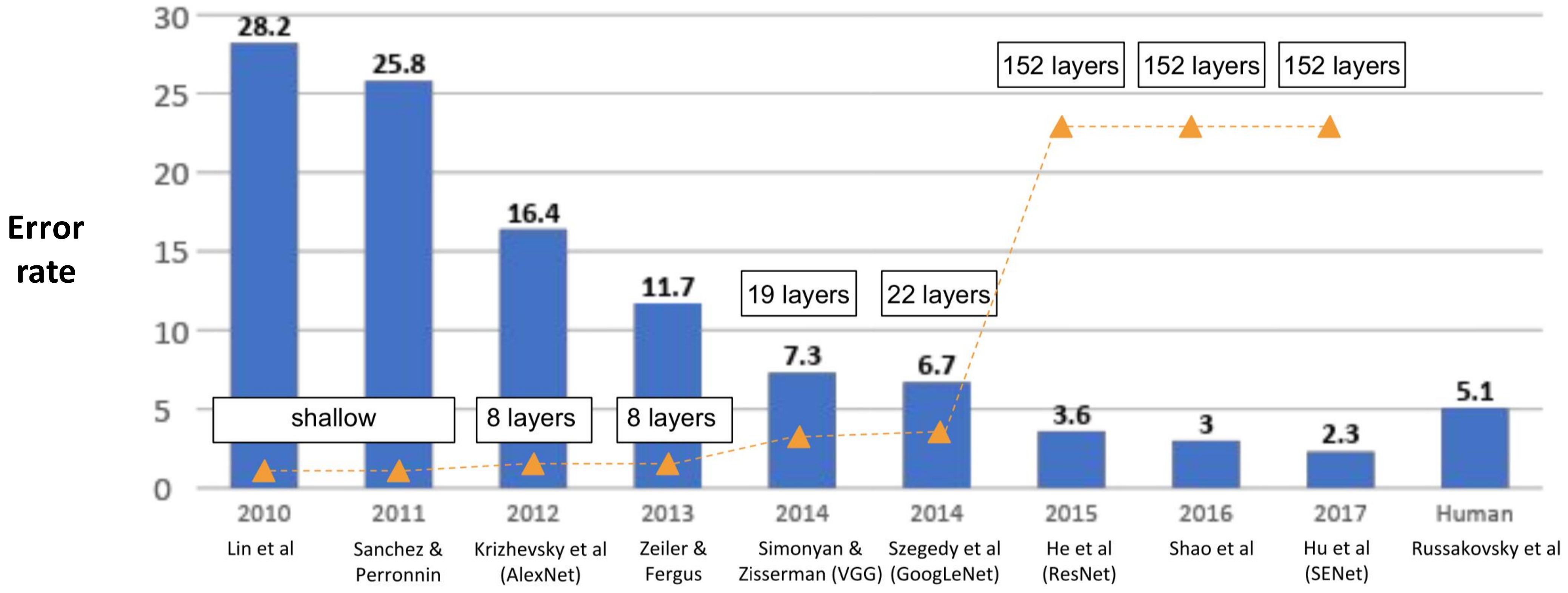
LeCun, Yann, et al. “Gradient-based learning applied to document recognition.” (1998)

CNN architectures

AlexNet:	the **1st** deep convolution neural network for image classification
VGG:	**very deep** convolution neural network for image recognition
GoogLeNet (Inception):	going **deeper** with convolutions
ResNet:	deep **residual** learning for image recognition
DenseNet:	**densely connected** convolutional network

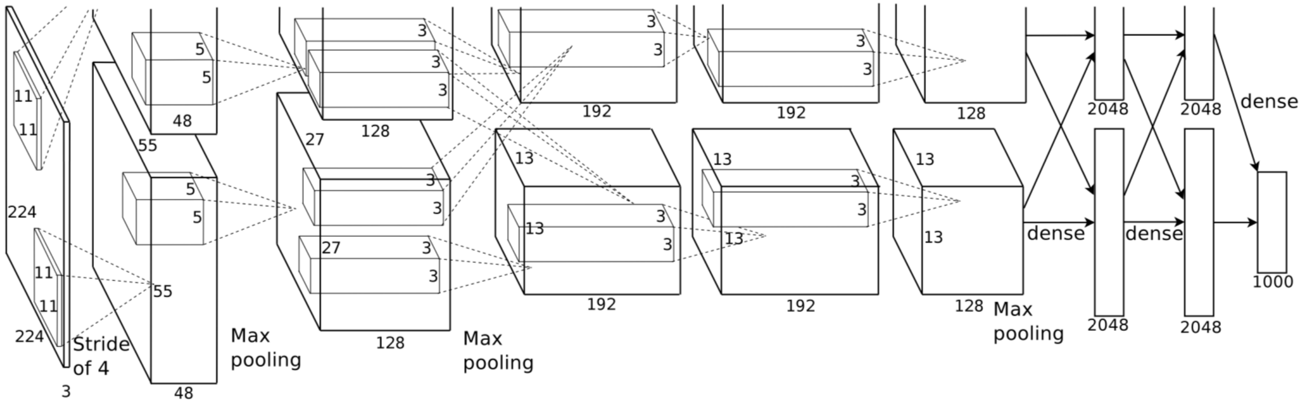
Overview: ILSVRC winners

- Large Scale Visual Recognition Competition (ILSVRC) is a world-famous challenge in image classification.
- CNNs are doing better as new architectures emerge.



AlexNet

- AlexNet is the first CNN model which achieves great success in image classification tasks.
- Due to limited computing power, AlexNet is spread over 2 GPUs with each one containing half of the channels.



Channels only cross-talk at certain layers.

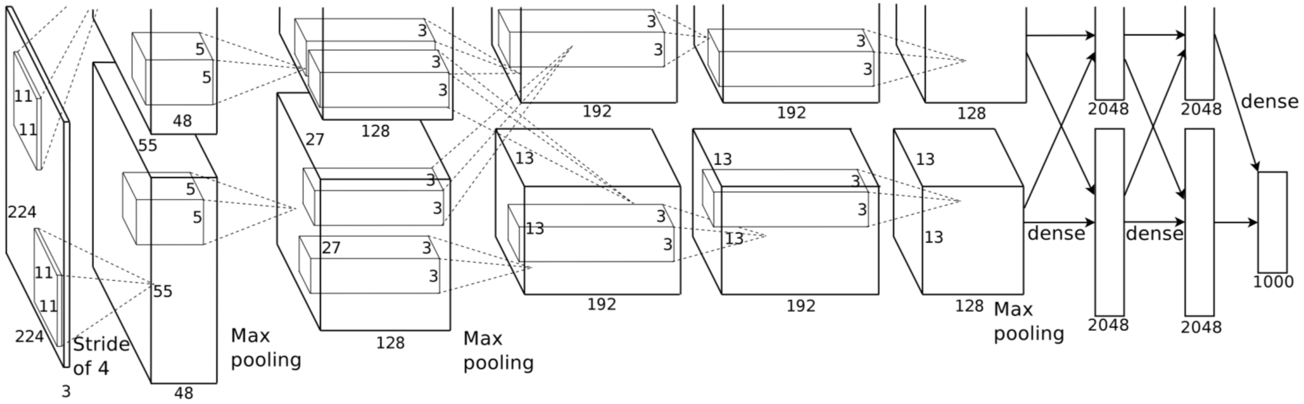
Features:

- Use very large kernels (11x11) in the first layer
- The first architecture using ReLU as non-linear activations
- Use local response normalization (LRN) to speedup training
- ImageNet top-5 error: 16.4%

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." (2012)

AlexNet

- AlexNet is the first CNN model which achieves great success in image classification tasks.
- Due to limited computing power, AlexNet is spread over 2 GPUs with each one containing half of the channels.



Question: What is the computation cost (MACs) for the first layer?

Answer: $11 \times 11 \times 3 \times 55 \times 55 \times 96 = 105.4\text{M}$

Features:

- Use very large kernels (11x11) in the first layer
- The first architecture using ReLU as non-linear activations
- Use local response normalization (LRN) to speedup training
- ImageNet top-5 error: 16.4%

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." (2012)

AlexNet

Local response normalization (LRN)

- AlexNet places LRN after the ReLU non-linearity in some layers.

$$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^\beta$$

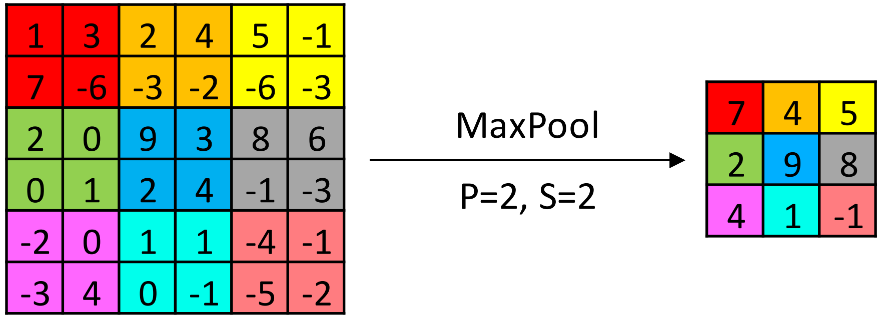
- LRN normalizes the activations to speed up the training of neural networks. It can also be viewed as an early version of batch normalization which is conducted within channels.

- In practice, AlexNet determines the hyperparameters k, n, α, β by running experiments on a validation set.
  - $k = 2, n = 5, \alpha = 10^{-4}$, and $\beta = 0.75$.

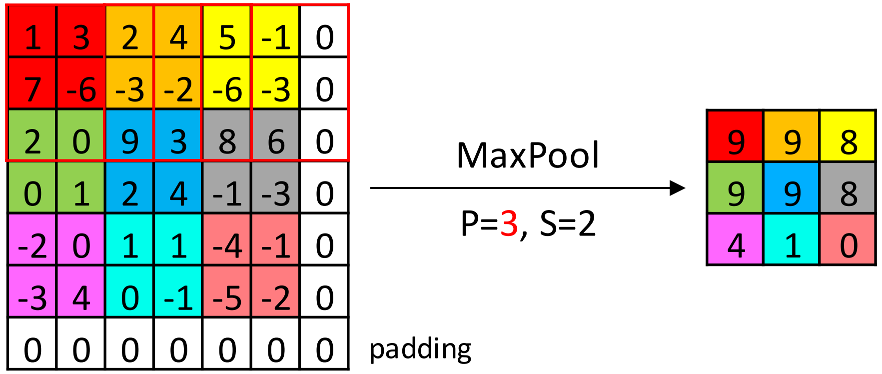
AlexNet

Overlapping Pooling

- AlexNet sets a larger pool size in each pooling layer to conduct overlapping pooling. This means that each pooling window **has a slight overlapping**.



- Overlapping pooling makes the model more difficult to overfit as it considers the magnitude of values over a joint pooling region.



Larger activation will dominate the down-sampled feature map.

AlexNet

Training AlexNet

- Batch size 128, initial learning rate 0.01, weight decay 5e-4
- Use SGD Momentum with momentum 0.9
- Use normalization layers (LRN)
- Dropout 0.5 for FC layers (except for final FC)
- Use aggressive data augmentation (shifting, flipping, etc.)

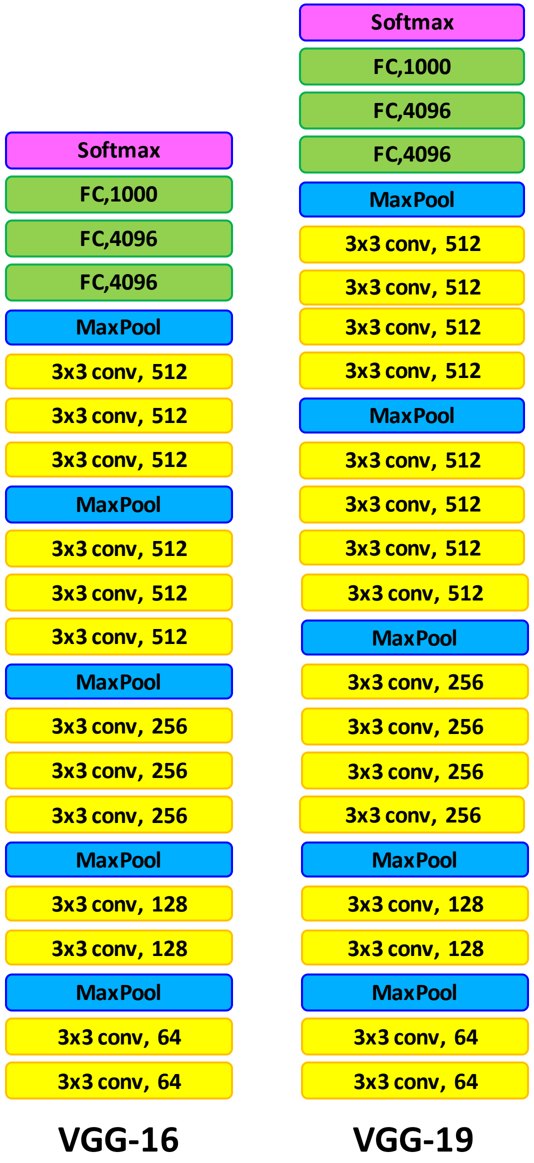
- Later training approaches inherit most of the AlexNet approach.

VGG

VGG creates very deep convolutional neural networks (11-19 layers) for image classification.

Features:

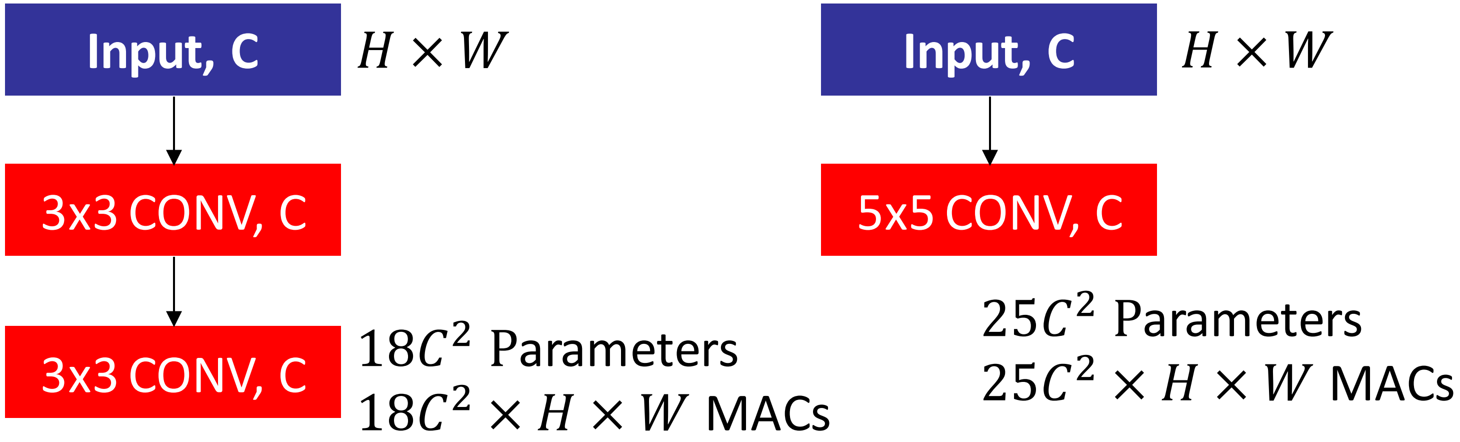
- Use 3×3 convolution to replace larger kernels (e.g., 11×11 in AlexNet)
- ImageNet top-5 error: 7.3%
- Until now, VGG model is still popular among a variety of tasks other than ImageNet (e.g., object detection)
- VGG is very large
  - 140/144 Million parameters and 16G/20G MACs in VGG 16/19
  - Hard to deploy for real-time applications



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

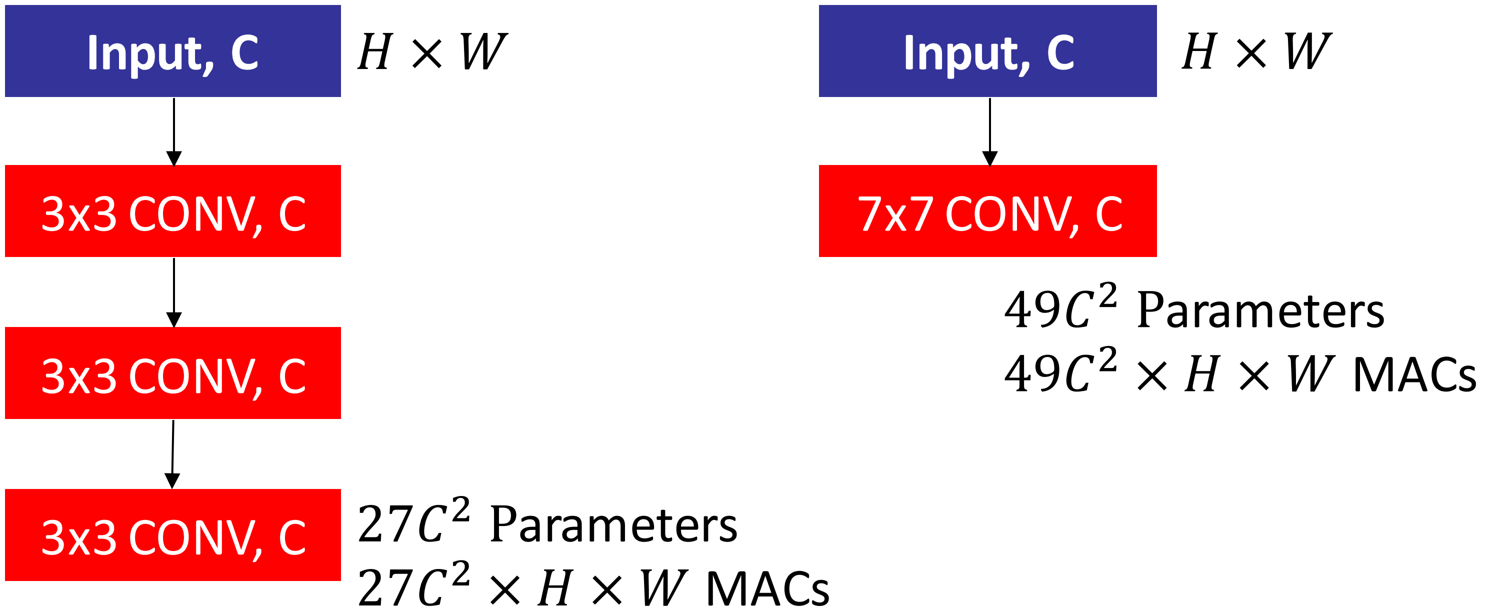
VGG: 3x3 convolution

- Receptive field is the size of the region in the input that produces the feature. VGG enlarges the receptive field by stacking **ONLY** 3x3 convolutions.
  - Two 3x3 convolutions has the same receptive field as one 5x5 convolution, however, with fewer parameters and MACs
  - However, two 3x3 convolutions perform better as it makes the model deeper



VGG: 3x3 convolution

- Receptive field is the size of the region in the input that produces the feature. VGG enlarges the receptive field by stacking **ONLY** 3x3 convolutions.
  - Three 3x3 convolution has the same receptive field as one 7x7 convolution with fewer parameters and MACs
  - However, three 3x3 convolutions perform better as it makes the model deeper



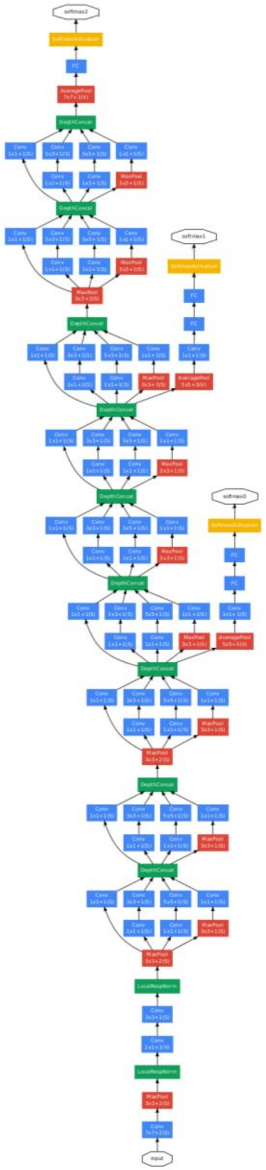
VGG

Training VGG

- Batch size 256, initial learning rate is 0.01, weight decay 5e-4
- Decay the learning rate by a factor of 0.1 if the validation accuracy plateaus
- Use SGD Momentum optimizer with momentum 0.9
- Dropout 0.5 for FC layers (except final FC that produces output)
- Data preprocessing is a bit different from AlexNet. For example, VGG-16 uses **multi-crop** (10-Crop) evaluation for better inference performance. See the VGG paper for more details
- Use **model ensembling** to reach better results. This is common in ILSVRC competition

Designing CNNs in a nutshell.
Fun fact, this meme was referenced in the first inception net paper.

https://knowyourmeme.com/memes/we-need-to-go-deeper



Designing CNNs in a nutshell.
Fun fact, this meme was referenced in the first inception net paper.

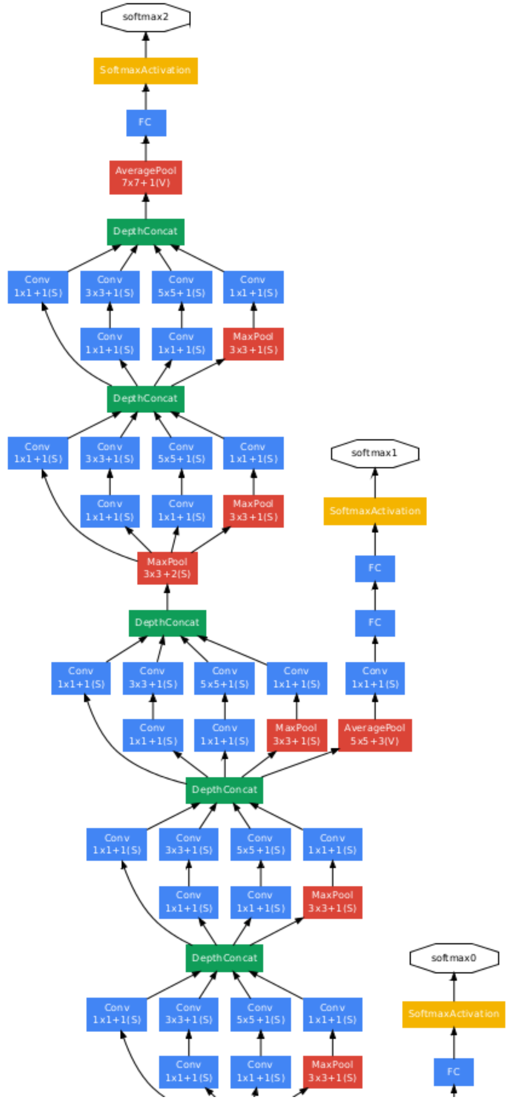
https://knowyourmeme.com/memes/we-need-to-go-deeper

GoogLeNet (Inception)

- From GoogLeNet, CNNs not only go deeper but evolve into multiple branches.

Features:

- Build a deeper model with a mixture of convolutions (3x3, 5x5, 1x1).
- Extract information from different parts of CNNs.
- Uses **local response normalization (LRN)** to speed up training.
- Use **bottleneck** structure to reduce parameters.
  - Yet, Inception is still time and memory-consuming.
  - The training takes about a week to reach convergence using a few high-end GPUs.
- ImageNet top-5 error: 6.7%

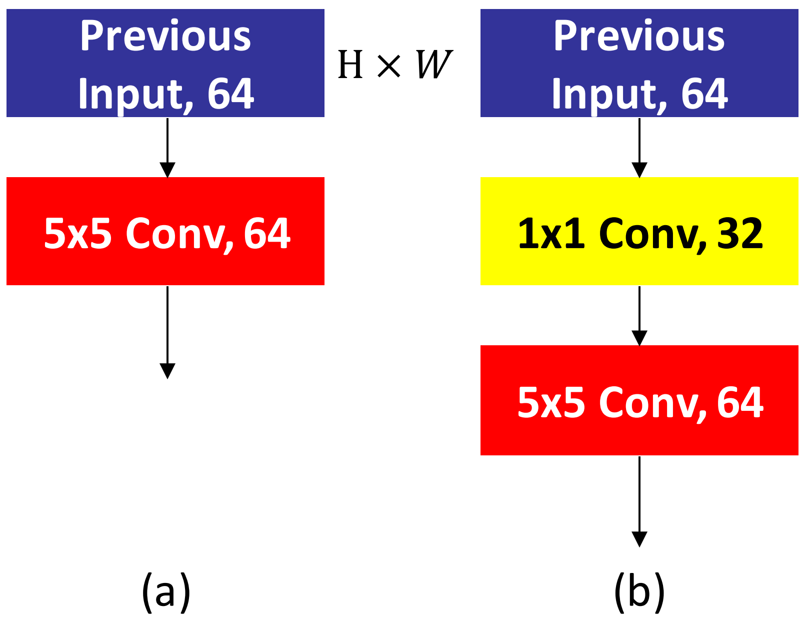


Szegedy, Christian, et al. “Going deeper with convolutions.’ (2015)

GoogLeNet: bottleneck structure

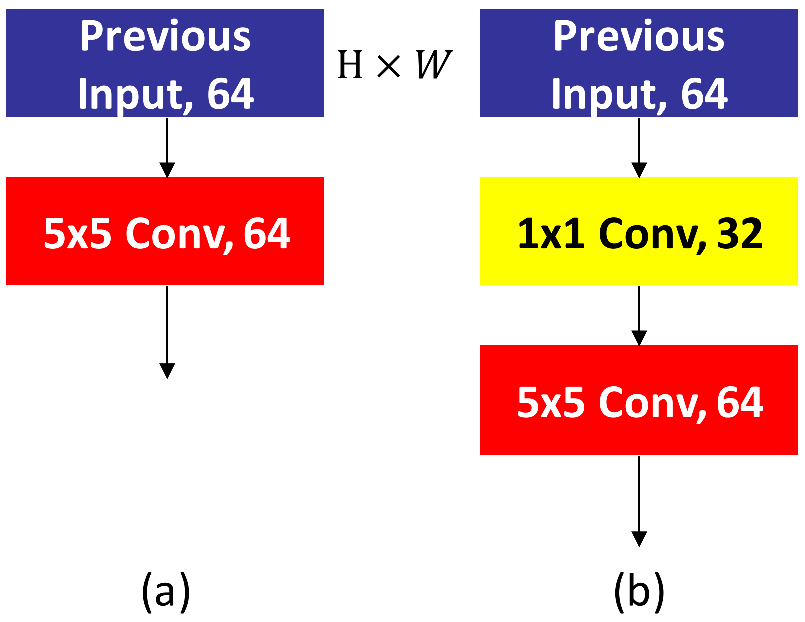
The number of filters is reduced before convolving large filters (e.g., 5x5). This structure is also called **bottleneck** structure.

Question: Calculate the weight parameter and MAC numbers for models in (a) & (b).



GoogLeNet: bottleneck structure

The number of filters is reduced before convolving large filters (e.g., 5x5). This structure is also called **bottleneck** structure.



Question: Calculate the weight parameter and MAC numbers for models in (a) & (b).

Answer:

(a) $5 \times 5 \times 64 \times 64 = 102{,}400$ weight parameters

$5 \times 5 \times 64 \times 64 \times H \times W = 102{,}400 \times H \times W$ MACs

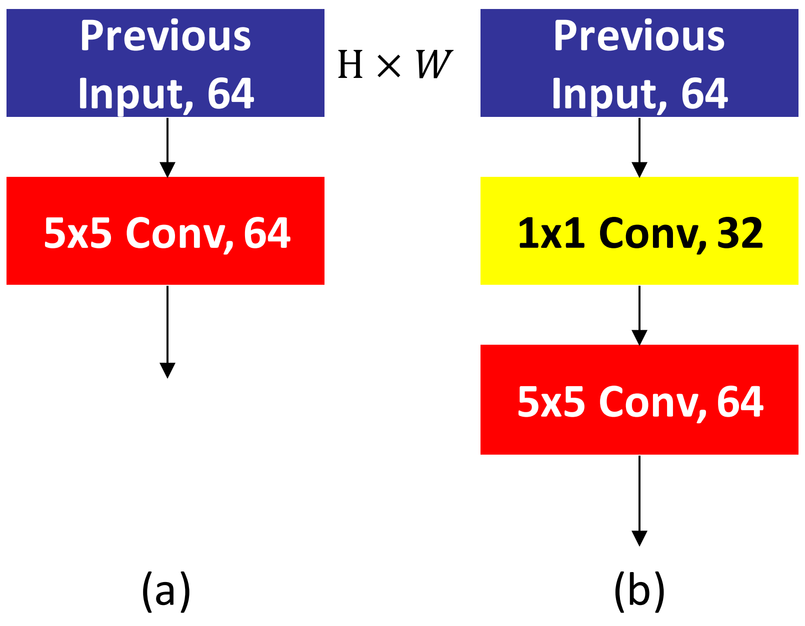
(b) $1 \times 1 \times 64 \times 32 + 5 \times 5 \times 32 \times 64 = 53{,}248$ weight parameters

$(1 \times 1 \times 64 \times 32 + 5 \times 5 \times 32 \times 64) \times H \times W = 53{,}248 \times H \times W$ MACs

GoogLeNet: bottleneck structure

Why is bottleneck structure useful?

- It seems that the bottleneck structures cut down the number of feature maps. Does it affect the performance of GoogLeNet?



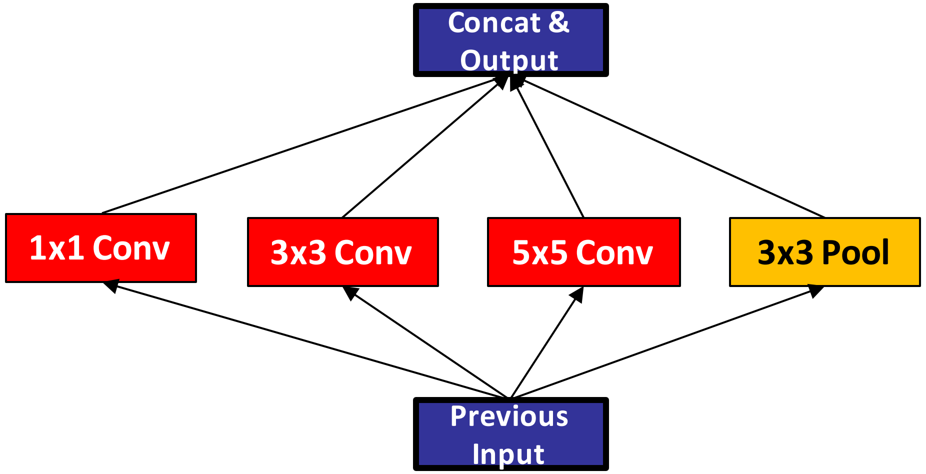
- Bottleneck structures project (compress) high-dimensional inputs into low-dimensional space.

- This 'projection' may even act as a form of regularization to prevent overfitting, which improves the performance of GoogLeNet.

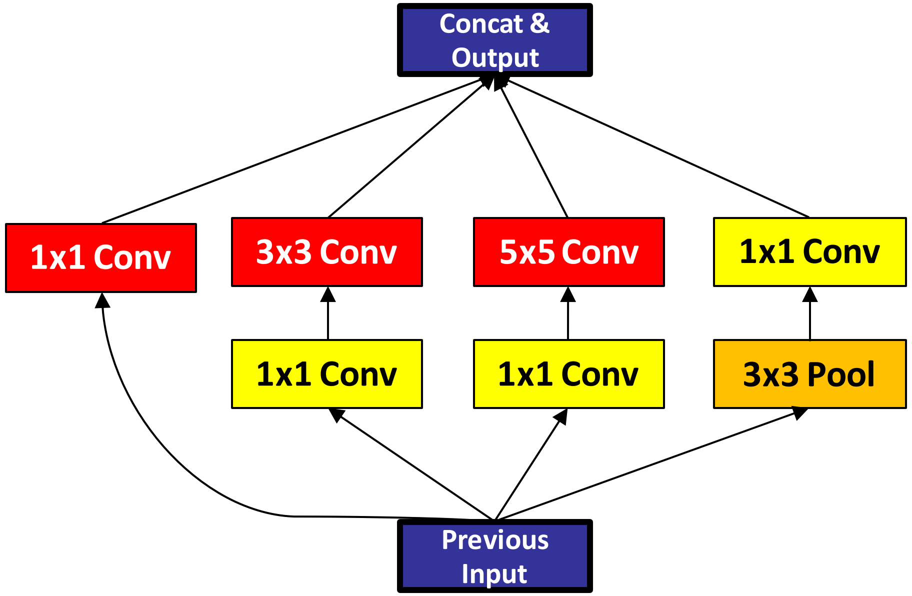
GoogLeNet: bottleneck structure

Where do we usually apply bottleneck structure?

- Bottleneck structure can be applied to any **branch** within the GoogLeNet structure.



Naïve GoogLeNet



GoogLeNet with dimension reduction

GoogLeNet

Train GoogLeNet

- Batch size is 256, initial learning rate is 0.01, decay the learning rate by a factor of 0.96 every 8 epochs
- Use **auxiliary tower** connected to intermediate layers as ways of combatting the overfitting problem. However, this is not common in later training approaches
- 40% dropout before the final FC layer
- Use SGD Momentum optimizer with momentum 0.9
- Preprocessing is a bit different from VGG. See the GoogLeNet-V1 (V2) paper for details
- Average predictions over multiple crops of the same input image

GoogLeNet-V2 (Inception-V2)

- GoogLeNet-V2 introduces **Batch Normalization (BN)** and adds it before each nonlinearity function in the Inception architecture.

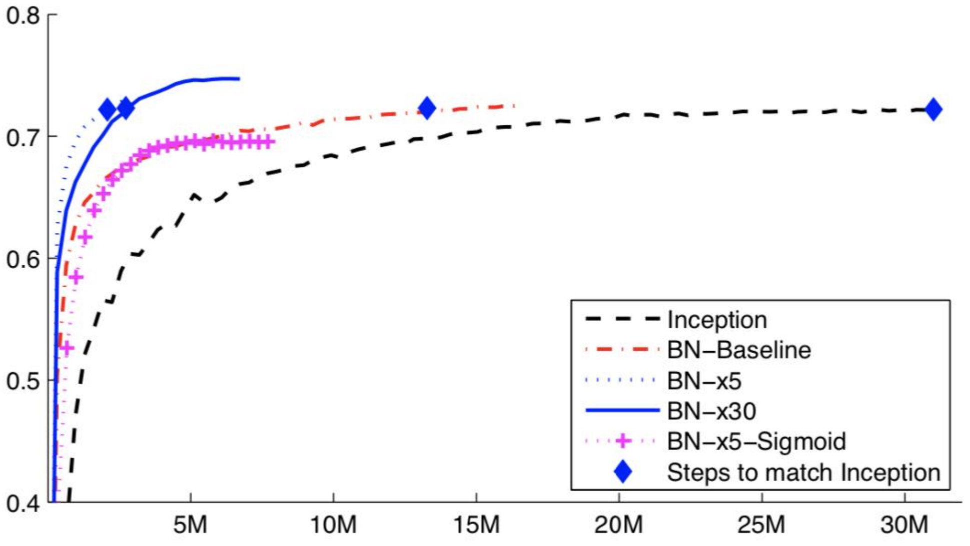


Figure 2: *Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.*

Features:

- Remove local response normalization (LRN) layers in previous architecture
- Introduce Batch Normalization (BN)
- GoogLeNet-V2 is trained faster than its GoogLeNet-V1 counterpart
- ImageNet top-5 error: 4.9%

Sergey Ioffe et al., Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift (2015)

ResNet：deep residual learning

Even with batch normalization, very deep neural networks are difficult to train without **residual learning**.

- **Optimization difficulty** grows with the number of parameters in a deep neural network
- **Gradient explosion/vanishing** problems occur when the networks go deeper
- It is desirable to learn **identity mappings** for generalization

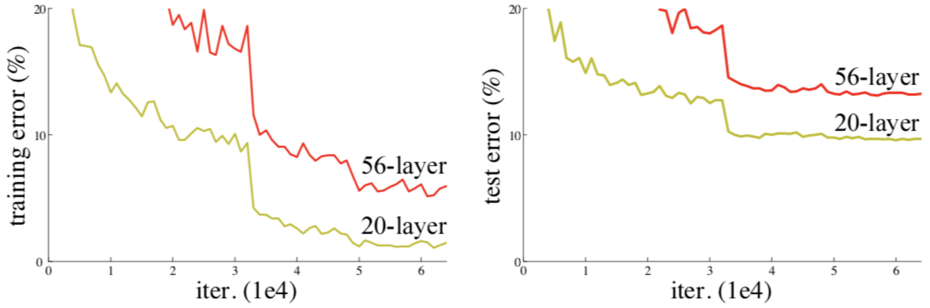


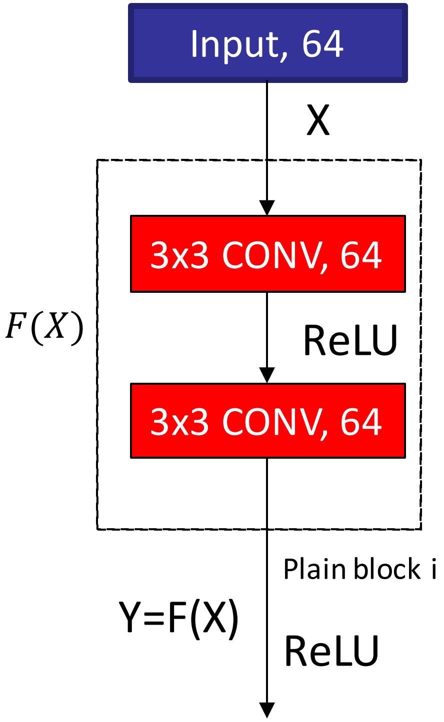
Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Residual learning can do it!

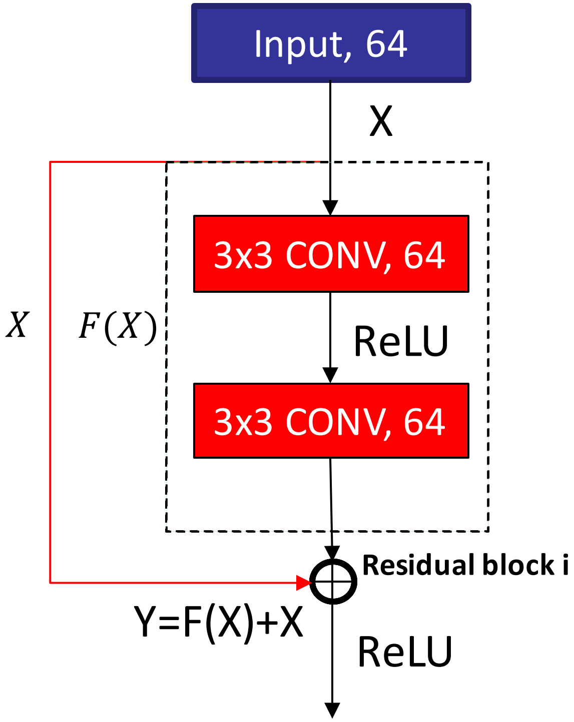
He, Kaiming, et al. "Deep residual learning for image recognition." (2015)

ResNet

- During DNN training, we are trying to learn a function F(X)=Y.
- ResNet adds a shortcut connection to fit a residual mapping F(X)=Y-X. This is called **residual learning**.



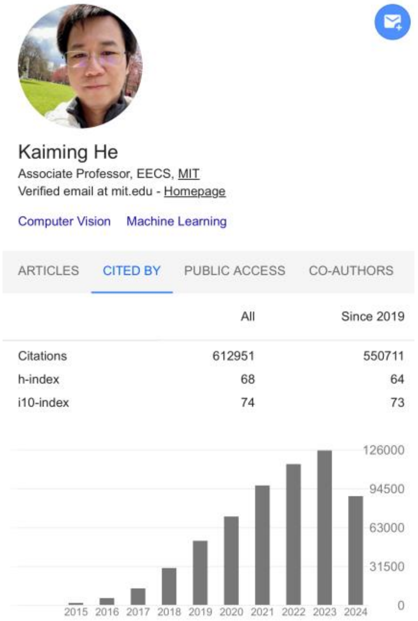
$$Y = F(X)$$



$$Y = F(X) + X$$
$$Y - X = F(X)$$

$$Y = F(x, \{W_i\}) + X$$

Residual mapping   Identity mapping

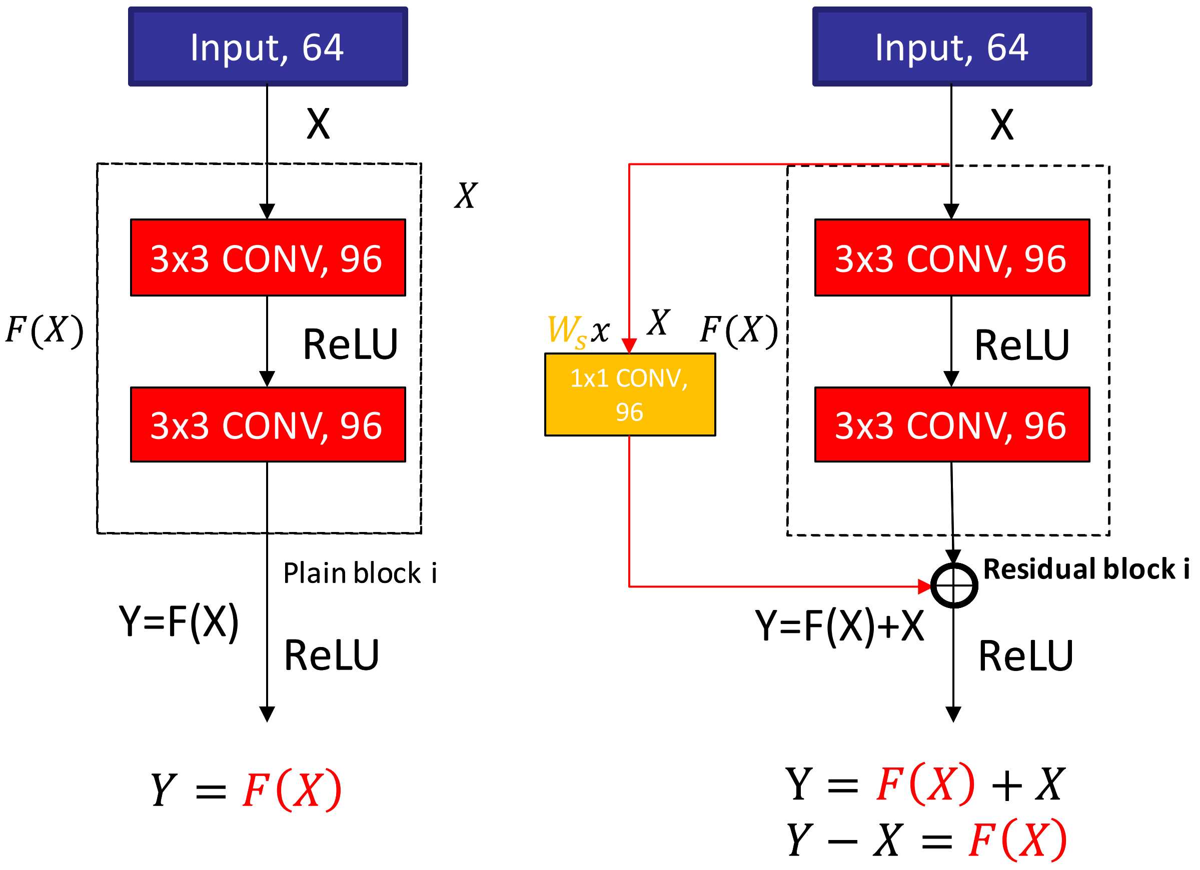


He, Kaiming, et al. "Deep residual learning for image recognition." (2015)

ResNet

What if the number of output filters in Y changes after passing function F(X)?

- Solution: Use 1x1 convolution to match dimension.



$$Y = F(x, \{W_i\}) + W_s x$$

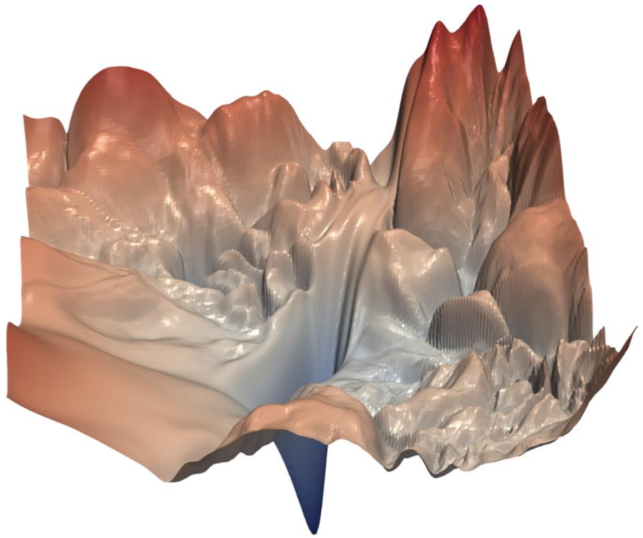
Original mapping   Residual mapping

$W_s \in \mathrm{R}^{1\times1\times64\times96}$ is used for identity mapping.

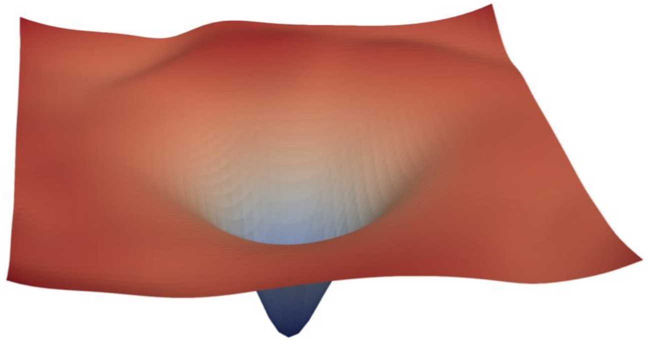
ResNet: visualization of loss surface

Residual learning make the optimization process easier!

- Shortcut connections make the loss surface smoother and easier to optimize.



Loss surface **without** shortcut connections. Noisy & curvy. Optimization process is easy to get stuck into local minimum.

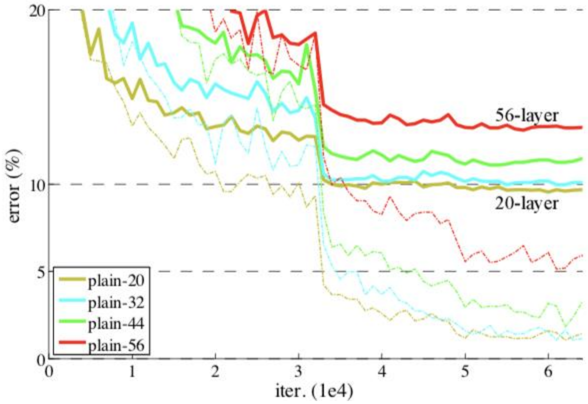


Loss surface **with** shortcut connections. Smooth & easy to optimize. Optimization process is easy to find sharp minima.

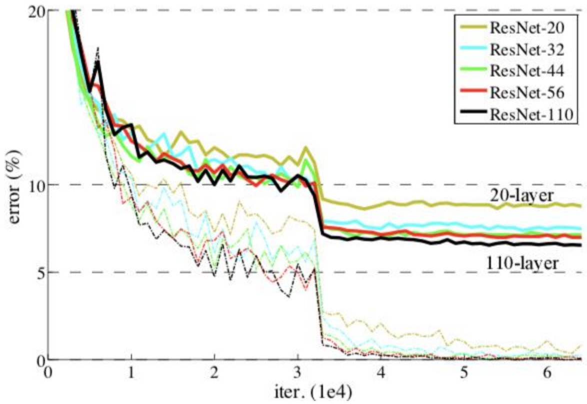
Li Hao et al., Visualizing the Loss Landscape of Neural Nets

ResNet: visualization of loss values

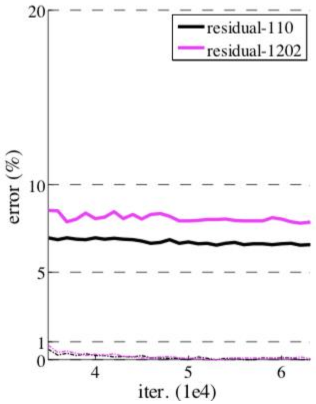
- The accuracy of plain networks is upper-bounded by the total number of layers.
- The accuracy of ResNet grows with the number of residual layers.



Plain networks



ResNets

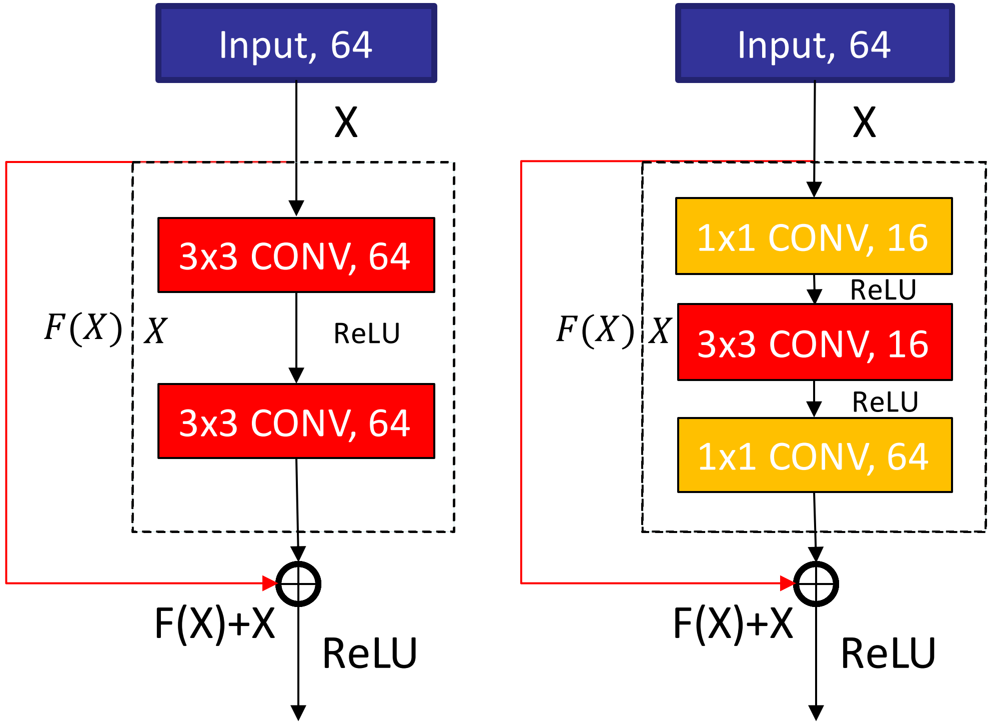


Very deep ResNets

- However, we cannot indefinitely add the depth of neural networks: a ResNet with more than 1000 layers is still difficult to optimize!

ResNet: residual block with bottleneck

- Similar to GoogLeNet, ResNet incorporates bottleneck layers into residual blocks to improve the efficiency of the model. This is called **ResNet Bottleneck**.
- Bottleneck layers make residual blocks deeper and more efficient.



16 1x1 filters **project** to 16 feature maps.

64 1x1 filters **expand** back to 64 feature maps.

ResNet: residual block with bottleneck

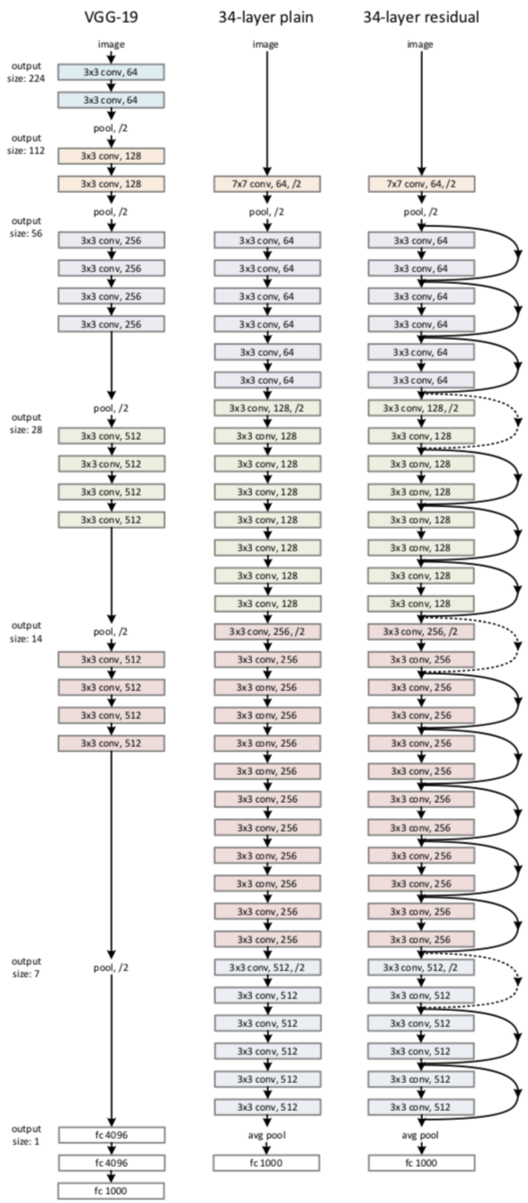
- Similar to GoogLeNet, ResNet incorporates bottleneck layers into residual blocks to improve the efficiency of the model. This is called **ResNet Bottleneck**.
- Bottleneck layers make residual blocks deeper and more efficient.

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	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\boxed{\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6}$	$\boxed{\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23}$	$\boxed{\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36}$
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

ResNet

Features

- ResNet constructs the neural network by stacking residual blocks. Each residual block has 2 3×3 convolutions.
- ResNet enables the design of very deep convolutional networks (from ResNet-18 to ResNet-152).
- By using bottleneck architectures, ResNet can be deeper and more efficient.
- ResNet achieves 3.57% top-5 error on ImageNet dataset.



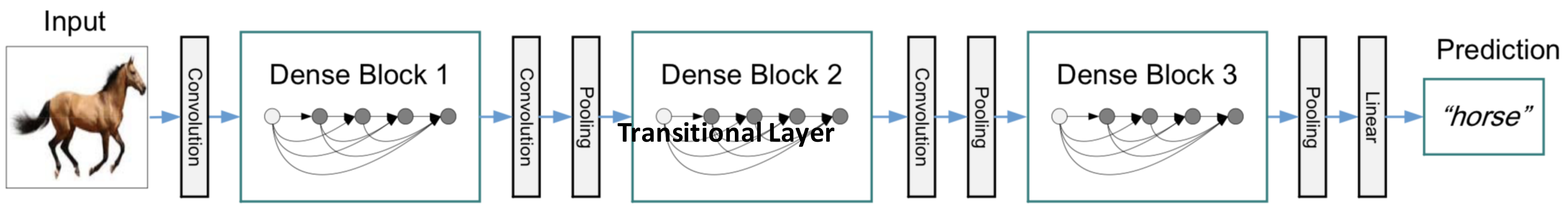
ResNet

Training ResNet

- Batch size is 256, use learning rate 0.01 to train for a few warmup epochs, then switch to initial learning rate 0.1
- Decay the learning rate by 0.1 when the validation accuracy plateaus
- Use SGD momentum with momentum 0.9
- Use 1e-5 L2 regularization
- Use similar preprocessing methods as inception

DenseNet

- **DenseNet** formulates **dense connectivity** between layers.



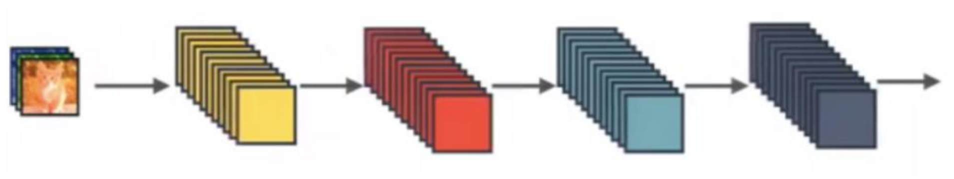
Features

- DenseNet encourages feature reuse, sharing, and interaction between **different positions (layers)** of the network.
- DenseNet uses a **transition** layer to do down-sampling on images.
- With respect to feature maps of similar depth, DenseNet reduces the weight parameter by using **filter concatenation**. However, DenseNet can lead to high memory consumption.
- ImageNet top-5 error: 5.3%

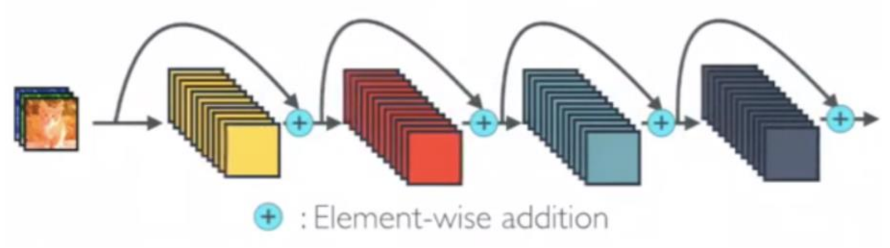
Huang, Gao, et al. "Densely connected convolutional networks." (2017)

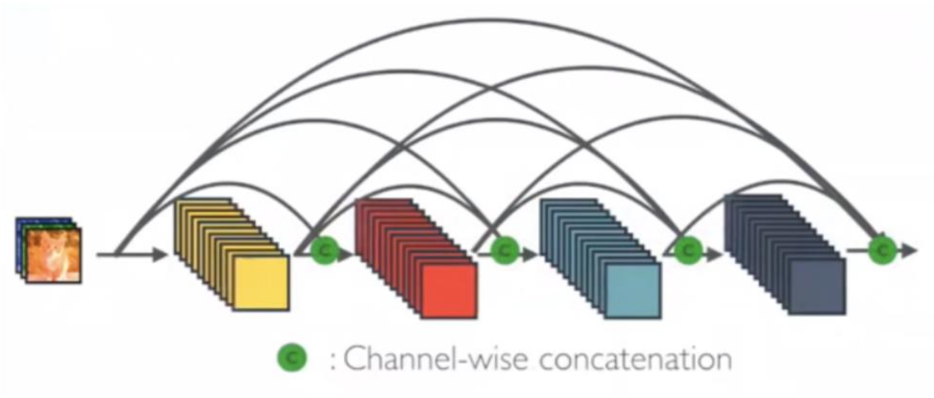
DenseNet: dense connections

Dense connections is achieved by **filter concatenation**.



Standard ConvNet Concept





One Dense Block in DenseNet

Standard ConvNet:
The output of the current layer is directly passed to the next layer.

ResNet:
The output of the current layer is first added to the identity of the current residual block, then passed to the next layer.

DenseNet:
The output of the previous layers in the current dense block is first concatenated and then passed to the next layer.

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

DenseNet

Training DenseNet

- Similar to ResNet
- The authors adopted the same training methodology as ResNet for fair comparison

Summary of CNN architectures

Architecture	Single-Path?	Use small (<=3) convolutional kernel?	Depth?
AlexNet	☑	✗	Shallow (8 layers)
VGG	☑	☑	Deep (16-19 layers)
GoogLeNet	✗	☑ (V3) ✗ (V1,V2,V4)	Deeper (22 layers)
ResNet	✗	☑	Very deep (34-152 layers)
DenseNet	✗	☑	Very deep (121-264 layers)

Architecture	Input Size	Parameter Memory	MACs
AlexNet	224x224	233 MB	727 M
VGG-16/19	224x224	528 MB / 548 MB	16 G /20 G
GoogLeNet	224x224	51 MB	2 G
ResNet-18/34	224x224	45 MB / 83 MB	2 G / 4 G
DenseNet-121	224x224	31 MB	3 G

Summary of CNN architectures

Metrics	LeNet 5	AlexNet	Overfeat fast	VGG 16	GoogLeNet v1	ResNet 50
Top-5 error†	n/a	16.4	14.2	7.4	6.7	5.3
Top-5 error (single crop)†	n/a	19.8	17.0	8.8	10.7	7.0
Input Size	28×28	227×227	231×231	224×224	224×224	224×224
# of CONV Layers	2	5	5	13	57	53
Depth in # of CONV Layers	2	5	5	13	21	49
Filter Sizes	5	3,5,11	3,5,11	3	1,3,5,7	1,3,7
# of Channels	1, 20	3-256	3-1024	3-512	3-832	3-2048
# of Filters	20, 50	96-384	96-1024	64-512	16-384	64-2048
Stride	1	1,4	1,4	1	1,2	1,2
Weights	2.6k	2.3M	16M	14.7M	6.0M	23.5M
MACs	283k	666M	2.67G	15.3G	1.43G	3.86G
# of FC Layers	2	3	3	3	1	1
Filter Sizes	1,4	1,6	1,6,12	1,7	1	1
# of Channels	50, 500	256-4096	1024-4096	512-4096	1024	2048
# of Filters	10, 500	1000-4096	1000-4096	1000-4096	1000	1000
Weights	58k	58.6M	130M	124M	1M	2M
MACs	58k	58.6M	130M	124M	1M	2M
Total Weights	60k	61M	146M	138M	7M	25.5M
Total MACs	341k	724M	2.8G	15.5G	1.43G	3.9G
Pretrained Model Website	[56]‡	[57, 58]	n/a	[57–59]	[57–59]	[57–59]

https://arxiv.org/abs/1703.09039

The original training setup for ImageNet

	AlexNet	VGG	GoogLeNet	ResNet
Year	2012	2014	2014	2015
Layer #	8	16-19	22	34-152
Batch size	128	256	256	256
LRN vs. BN	LRN	*	V1: LRN V2: BN	BN
Learning rate	0.01	0.01, decay 0.1	0.01, decay 0.96 every 8 epochs	0.1 (0.01 warmup), decay 0.1
Optimizer	SGD w/o momentum	SGD w/o momentum =0.9		
Weight decay (regularization)	5e-4	5e-4		1e-5
Dropout	0.5	0.5	0.4	No dropout
Data preprocessing	Shifting, flipping	+ Multi-crop	+ Brightness, aspect ratio distortion	
Model ensemble	No	Yes	Yes	Yes

Reading Materials

AlexNet: Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “Imagenet classification with deep convolutional neural networks.” (2012)

VGG: Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition.” (2014)

GoogLeNet (Inception): Christian Szegedy, et al. “Going deeper with convolutions.’ (2015)
Sergey Loffe, et al., Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift (2015)

ResNet: Kaiming He, et al. "Deep residual learning for image recognition.” (2015)

DenseNet: Gao Huang, et al. “Densely connected convolutional networks.” (2017)