Computer Vision

Naeemullah Khan

naeemullah.khan@kaust.edu.sa

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

November 19, 2023

Building artificial systems that process, perceive, and reason about visual data

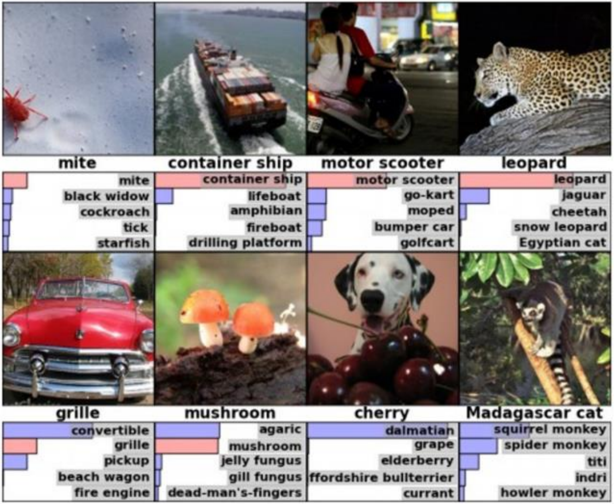
Computer Vision is Everywhere

Left to right:
Image by Roger H Goun is licensed under CC BY 2.0
Image is CC0 1.0 public domain
Image is CC0 1.0 public domain
Image is CC0 1.0 public domain

Left to right:
Image is free to use
Image is CC0 1.0 public domain
Image by NASA is licensed under CC BY 2.0
Image is CC0 1.0 public domain

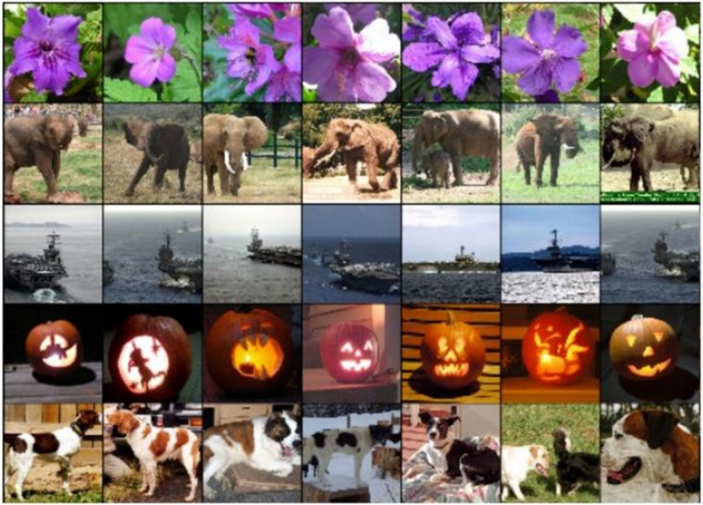
Bottom row, left to right
Image is CC0 1.0 public domain
Image by Derek Keats is licensed under CC BY 2.0; changes made
Image is public domain
Image is licensed under CC-BY 2.0; changes made

Image Classification

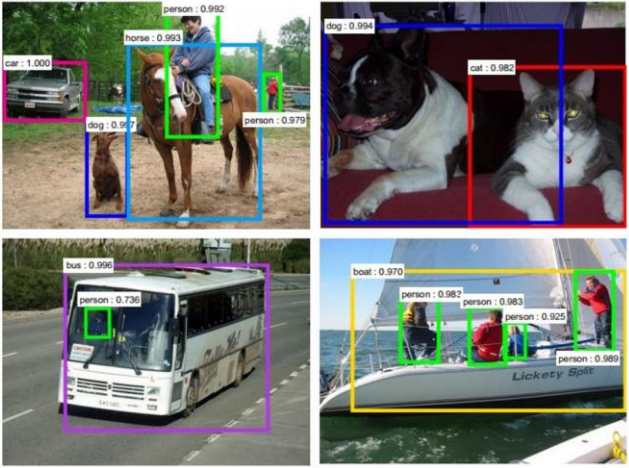


Some Applications (cont.)

Image Retrieval



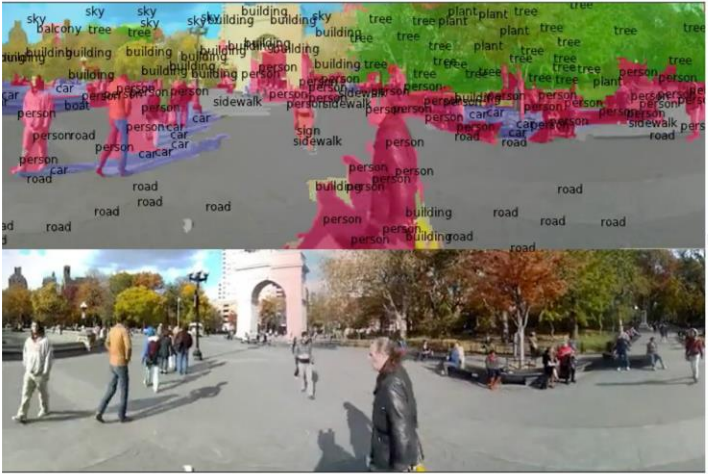
Object Detection



Ren, He, Girshick, and Sun, 2015

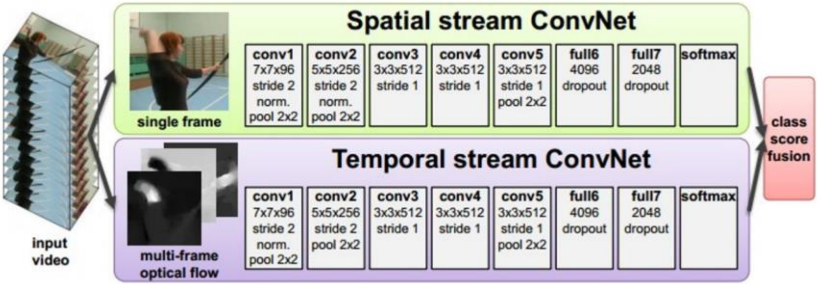
Some Applications (cont.)

Image Segmentation

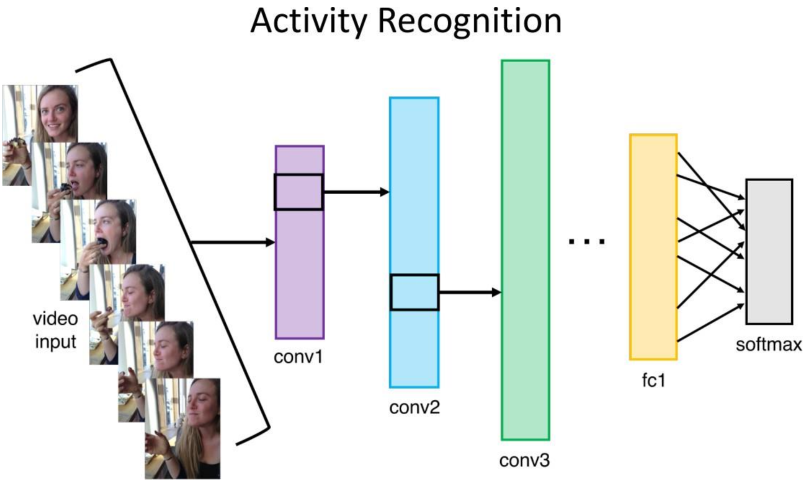


Fabaret et al, 2012

Video Classification

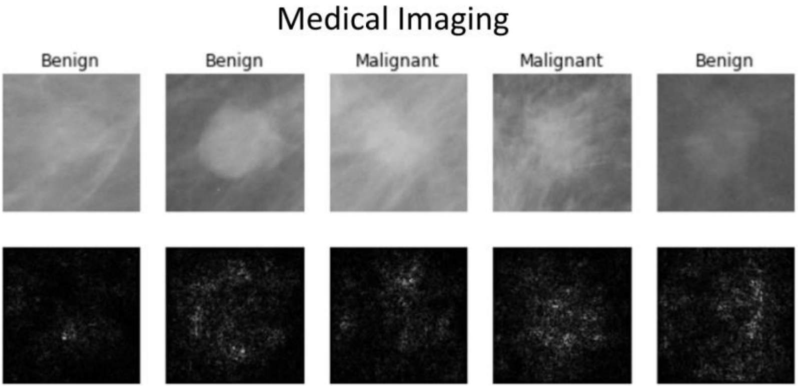


Simonyan et al, 2014


Activity Recognition
video input
conv1
conv2
conv3
fc1
softmax

Some Applications (cont.)

Pose Recognition (Toshev and Szegedy, 2014)



Medical Imaging
Benign
Benign
Malignant
Malignant
Benign

Image Captioning

Vinyals et al, 2015
Karpathy and Fei-Fei, 2015

A white teddy bear sitting in the grass

A man in a baseball uniform throwing a ball

A woman is holding a cat in her hand

A man riding a wave on top of a surfboard

A cat sitting on a suitcase on the floor

A woman standing on a beach holding a surfboard

Image Generation

"Teddy bears working on new AI research underwater with 1990s technology"

DALL-E 2

Some Applications (cont.)

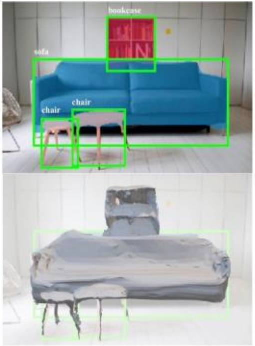
Style Transfer

Some Applications (cont.)

3D Vision

Choy et al., 3D-R2N2: Recurrent Reconstruction Neural Network (2016)

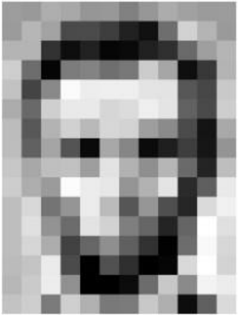
Zhou et al., 3D Shape Generation and Completion through Point-Voxel Diffusion (2021)

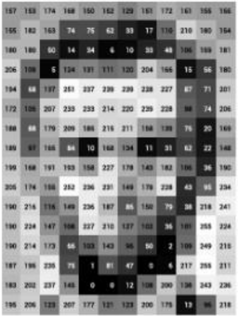


Gkioxari et al., "Mesh R-CNN", ICCV 2019

How to represent an image?

- Images are represented as Matrices with elements in $[0, 255]$
- Grayscale images have one channel while RGB images have 3 channels





157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

[0] https://www.v7labs.com/blog/image-recognition-guide

Fully-Connected Neural Networks

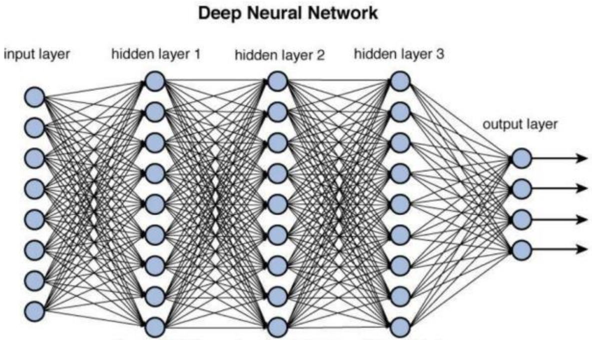
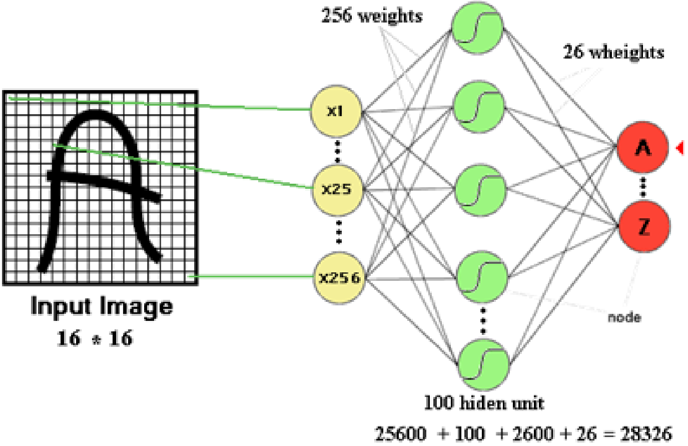


Figure 12.2 Deep network architecture with multiple layers.

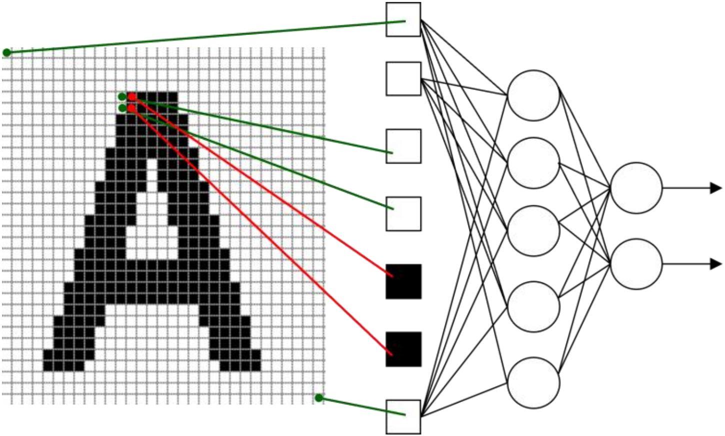
$$z = W_1x_1 + W_2x_2 + \cdots + W_nx_n + b$$

[0] https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964

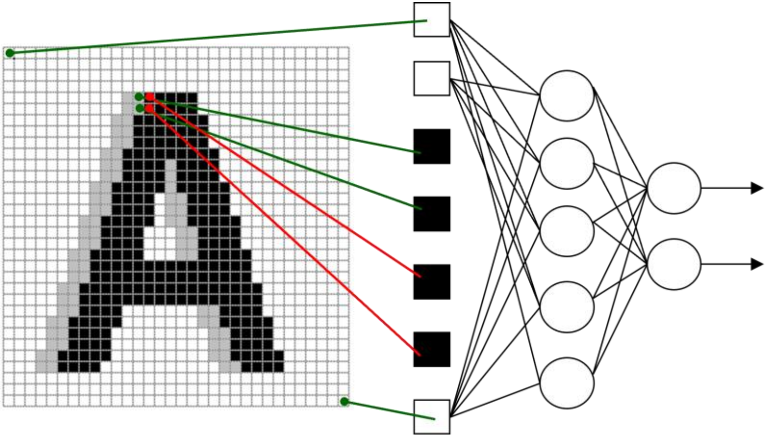
- The number of trainable parameters becomes extremely large



- Little or no invariance to shifting, scaling, and other forms of distortion

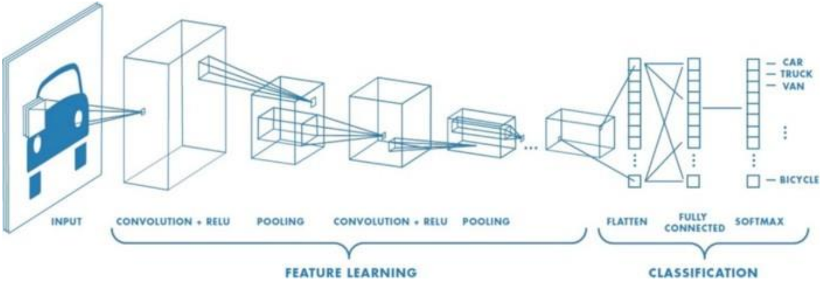


- Little or no invariance to shifting, scaling, and other forms of distortion



- The topology of the input data is completely ignored
- For a 32×32 image, we have
  - Black and white patterns: $2^{32*32} = 2^{1024}$
  - Grayscale patterns: $256^{32*32} = 256^{1024}$

Convolutional Neural Networks (CNNs)

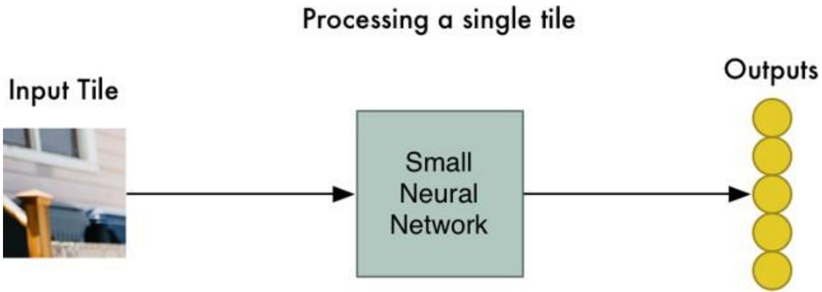


$$z = W * x_{i,j} = \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} W_{ab} x_{(i+a)(j+b)}$$

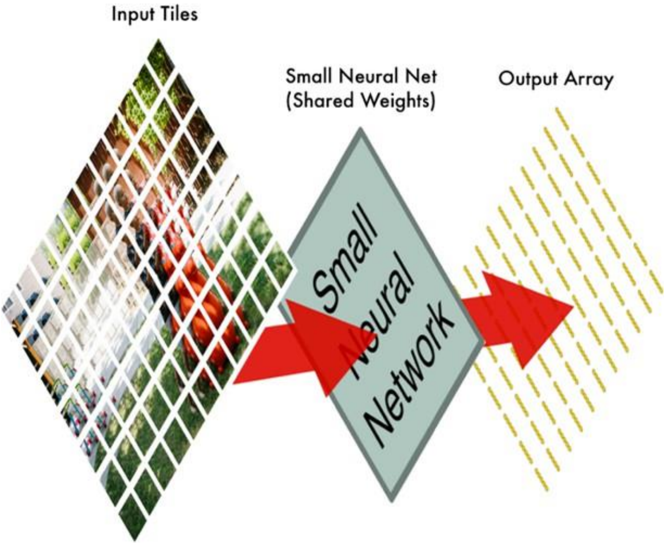
How Convolution Works?

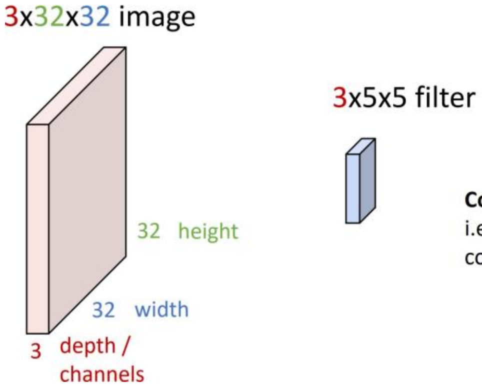
How Convolution Works? (cont.)

How Convolution Works? (cont.)



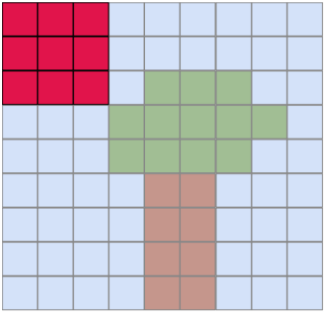
How Convolution Works? (cont.)

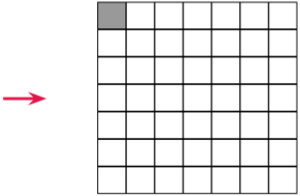




Convolve the filter with the image i.e. "slide over the image spatially, computing dot products"

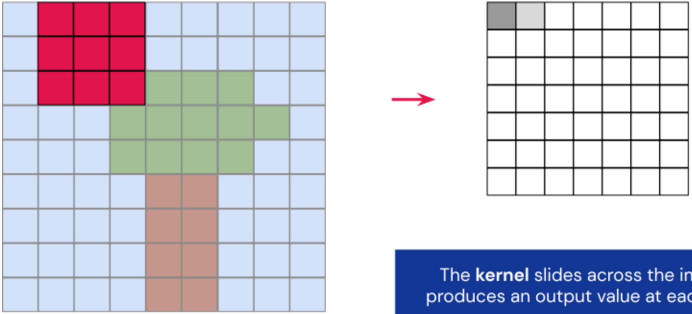
How Convolution Works? (cont.)





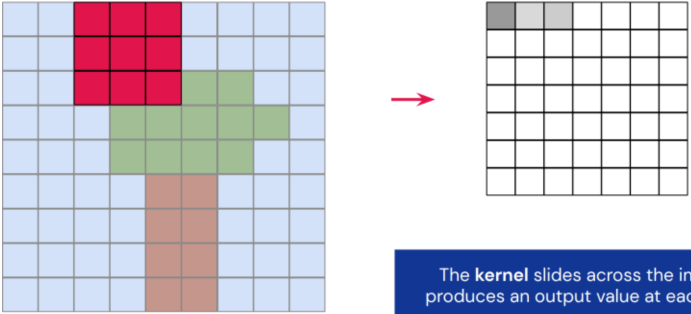
The **kernel** slides across the image and produces an output value at each position

How Convolution Works? (cont.)



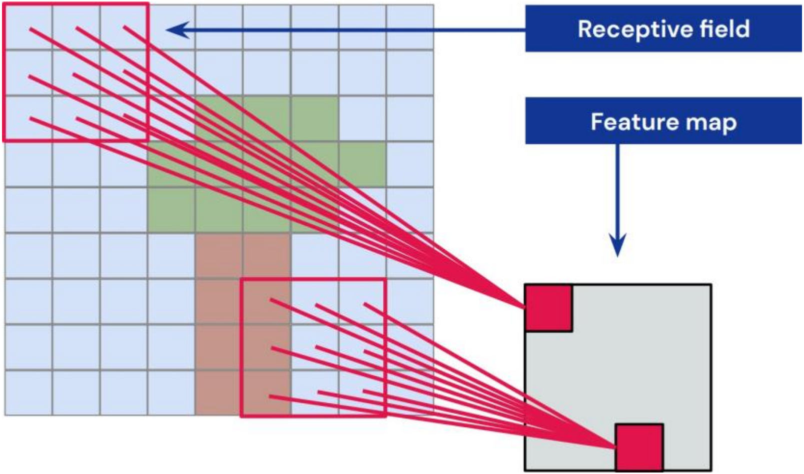
The **kernel** slides across the image and produces an output value at each position

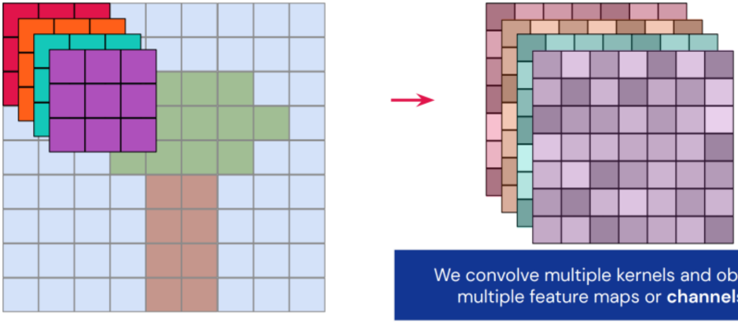
How Convolution Works? (cont.)



The **kernel** slides across the image and produces an output value at each position

How Convolution Works? (cont.)





We convolve multiple kernels and obtain multiple feature maps or **channels**

How Convolution Works? (cont.)

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Vertical edge detection

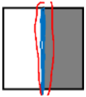
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

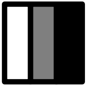
1	0	-1
1	0	-1
1	0	-1

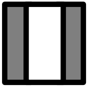
=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

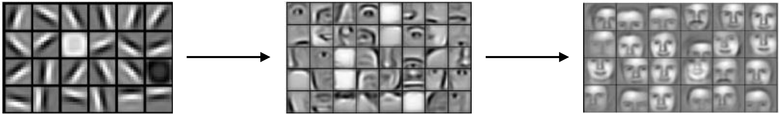


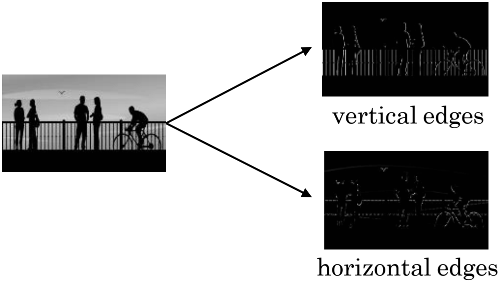
*





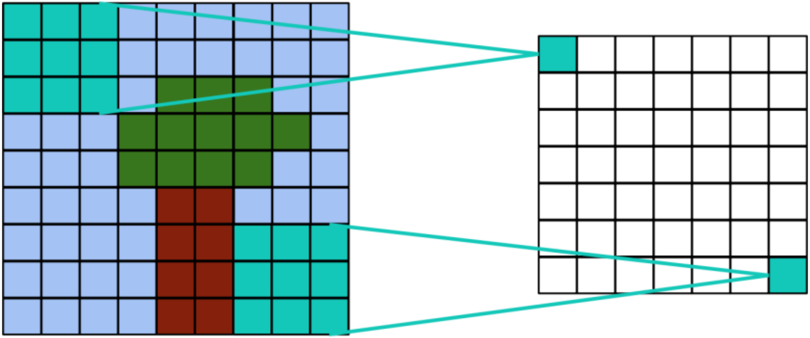
Edge detection example



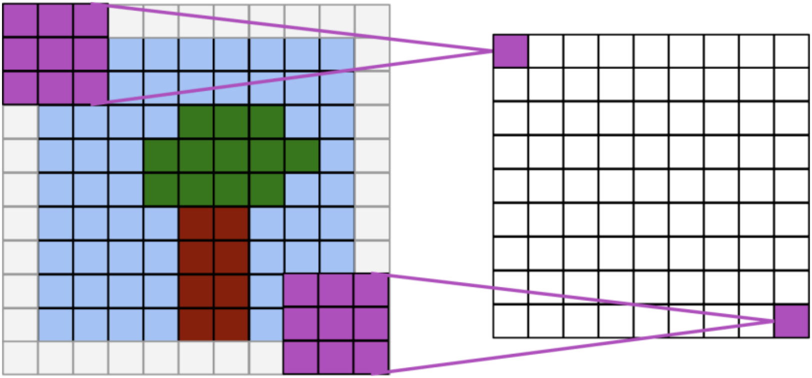


Padding

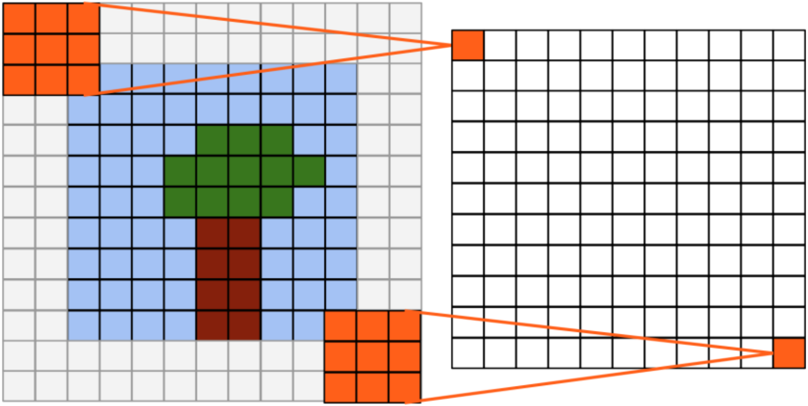
- Applying Convolution as such reduces the size of the borders.
- Sometimes this is not desirable.
- We can pad the border with zeros.



- Same Convolution: Output is the same size as input

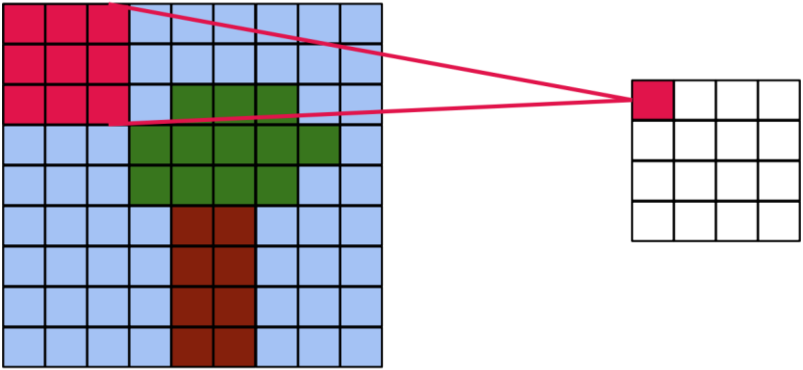


- Full Convolution: output size = input size + kernel size - 1



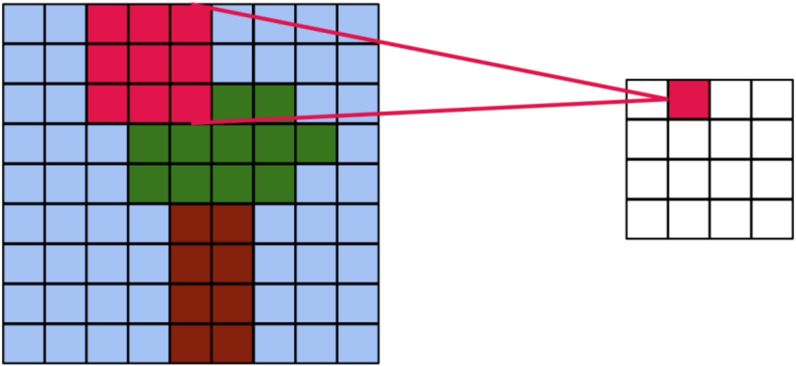
Strided Convolution

- Kernel slides along the image with a step > 1



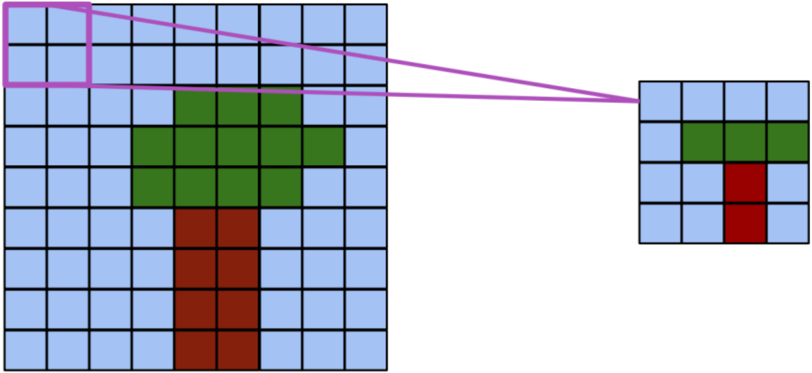
Strided Convolution (cont.)

- Kernel slides along the image with a step > 1

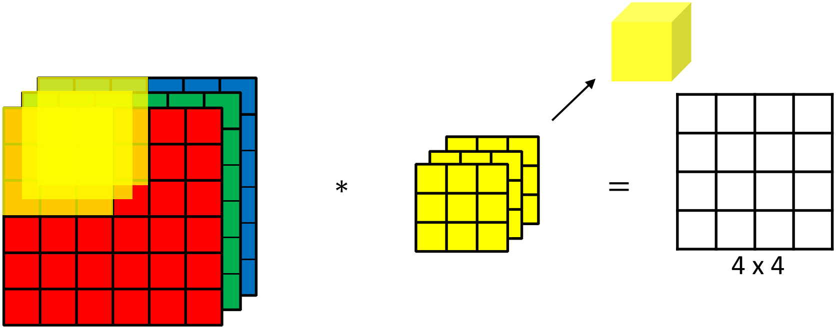


Pooling

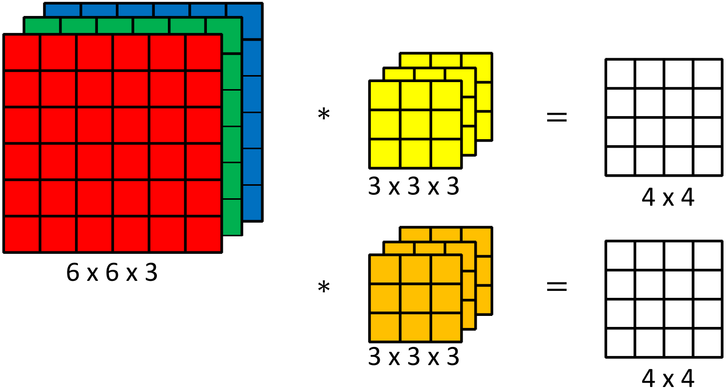
- Compute mean or max over small windows to reduce resolution



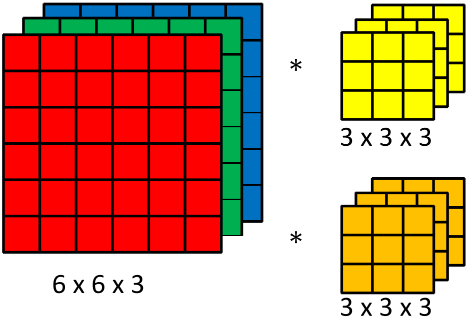
Convolutions on RGB images

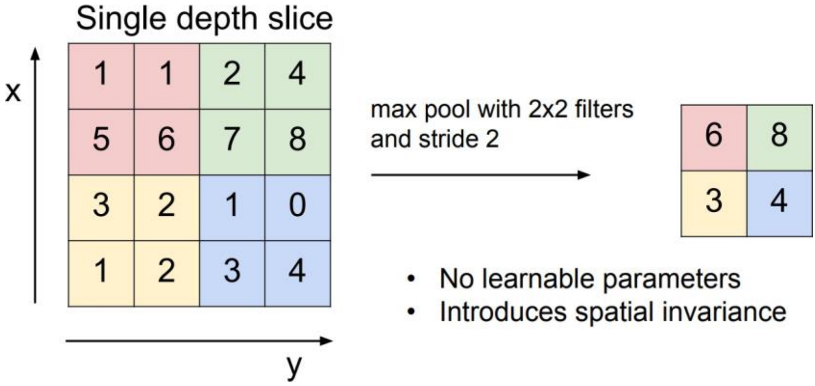


Multiple filters



Example of a conv layer

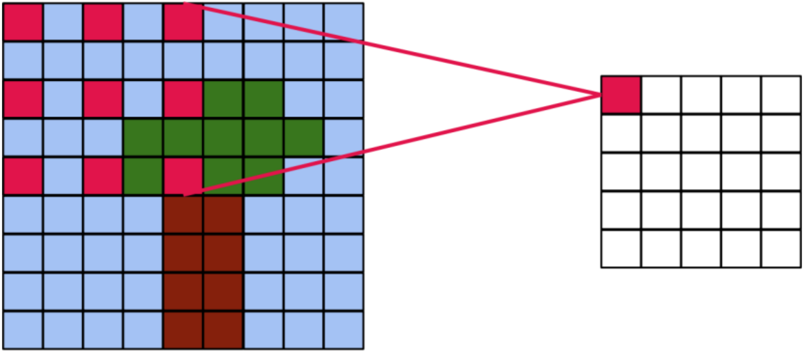




- No learnable parameters
- Introduces spatial invariance

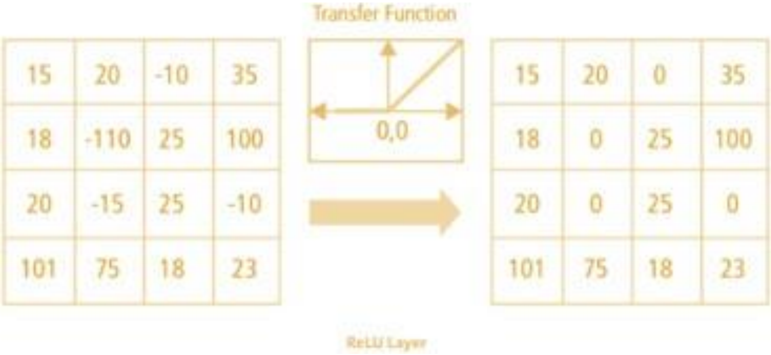
Dilated Convolution

- Kernel is spread out, step > 1 between kernel elements



Activation

- Just like Fully-Connected Neural Networks, we can apply an activation over convolutional layer outputs
- It helps break linearity
- For example, Rectified Linear Unit (ReLU): $\sigma(x) = max(0, x)$



Why convolutions

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

1	0	-1
1	0	-1
1	0	-1

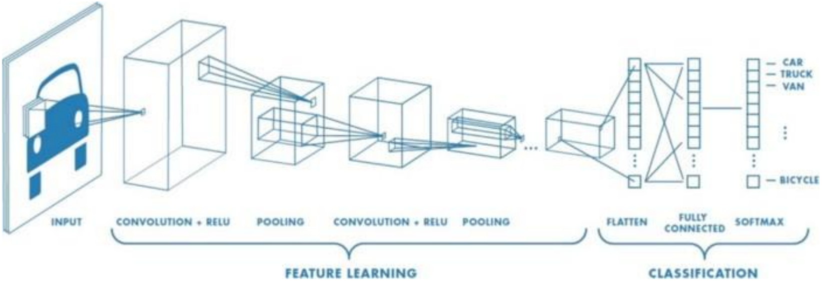
=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

Parameter sharing: A feature detector (such as a vertical edge detector) that's useful in one part of the image is probably useful in another part of the image.

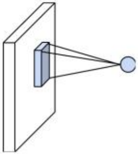
Sparsity of connections: In each layer, each output value depends only on a small number of inputs.

Convolutional Neural Networks

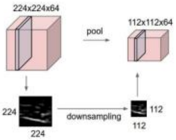


Components of a CNN

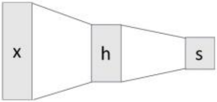
Convolution Layers



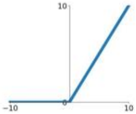
Pooling Layers



Fully-Connected Layers



Activation Function



Normalization

$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$

If you have 10 filters that are 3 x 3 x 3 in one layer of a neural network, how many parameters does that layer have?

Output Shape

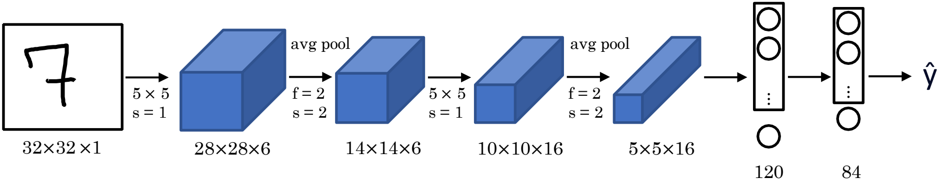
$$\text{Floor}\left(\frac{(W-F+2P)}{S}+1\right)$$

https://androidkt.com/calculate-output-size-convolutional-pooling-layers-cnn/

Most Notable CNNs

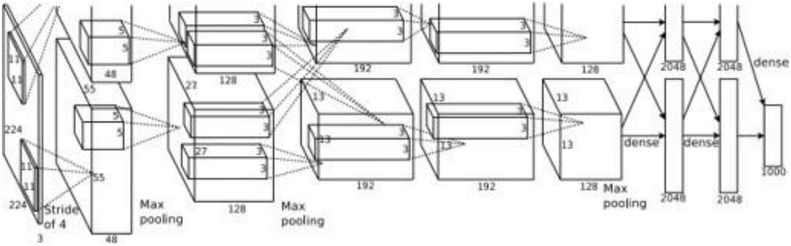
- LeNet
- AlexNet *[Krizhevsky et al. 2012]*
- VGGNet *[Simonyan and Zisserman, 2014]*
- InceptionNet (GoogLeNet) *[Szegedy et al., 2014]*
- ResNet *[He et al., 2015]*
- *MobileNet*

LeNet - 5

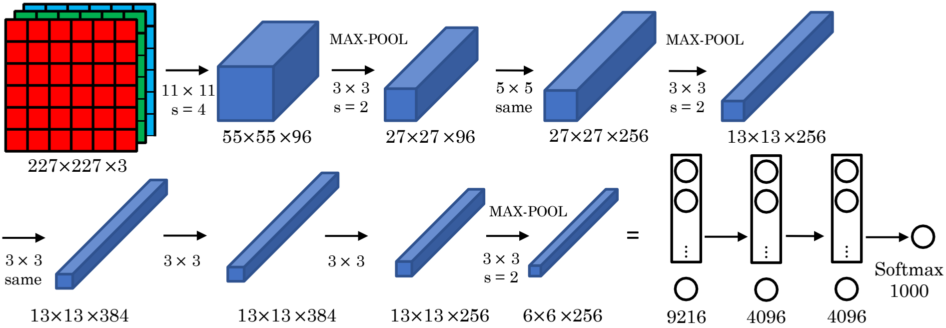


[LeCun et al., 1998. Gradient-based learning applied to document recognition]

- First big improvement in image classification
- Made use of CNN, pooling, dropout, ReLU and training on GPUs.
- 5 convolutional layers, followed by max-pooling layers; with three fully connected layers at the end



AlexNet

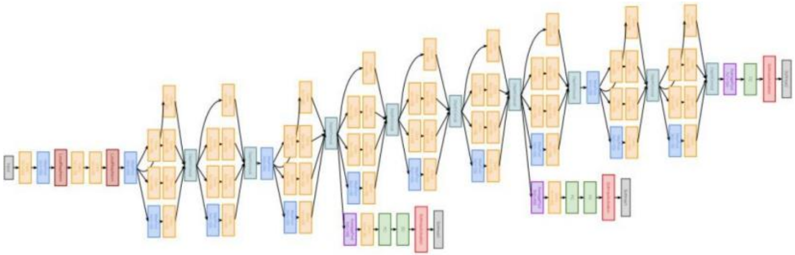


[Krizhevsky et al., 2012. ImageNet classification with deep convolutional neural networks]

VGGNet

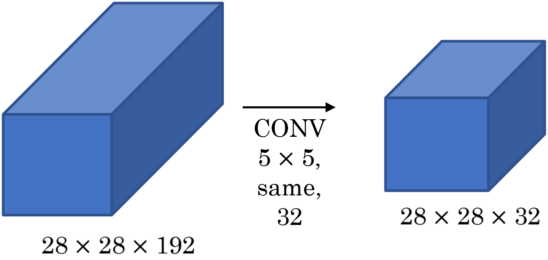
- Stack of three 3x3 conv (stride 1) layers has same effective receptive field as one 7x7 conv layer
- But deeper, more non-linearities and lesser parameters
- 13 or 16 conv layers with 3 fully-connected layers. Most params in the fully connected layer

- Going Deep: 22 layers
- Only 5 million parameters! (12x less than AlexNet and 27x less than VGGNet)
- Introduced efficient "Inception module"
- Introduced "bottleneck" layers that use 1x1 convolutions to reduce feature channel size and computational complexity

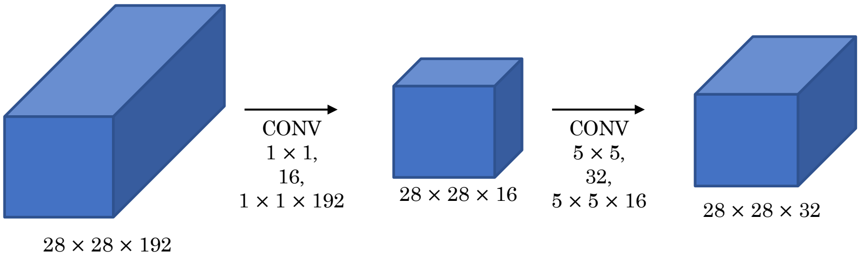


Inception network

The problem of computational cost

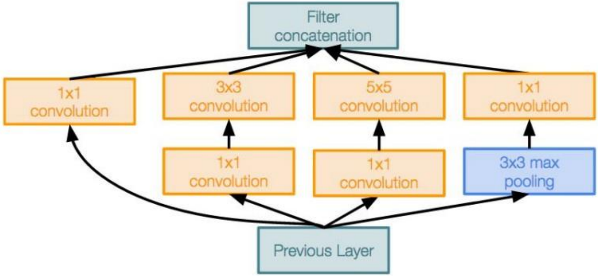


Using 1x1 convolution



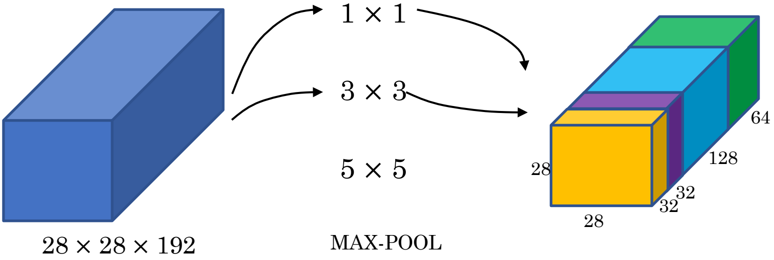
InceptionNet (cont.)

- **Inception module:** design a good local network topology (network within a network) and then stack these modules on top of each other



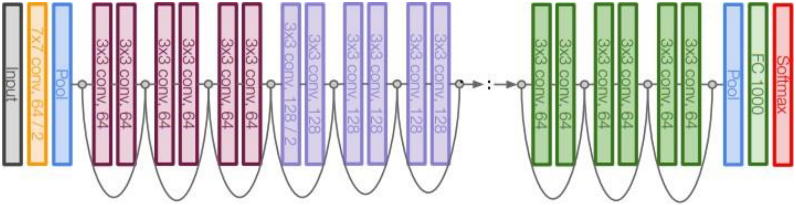
Inception module

Inception network



[Szegedy et al. 2014. Going deeper with convolutions]

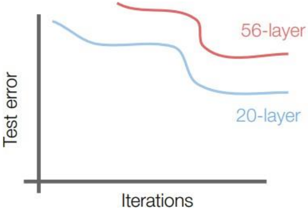
- Very deep networks using residual connections
- 152-layer model for ImageNet
- Stacked Residual Blocks



- What happens when we continue stacking deeper layers on a ”plain” convolutional neural network?

- What happens when we continue stacking deeper layers on a ”plain” convolutional neural network?

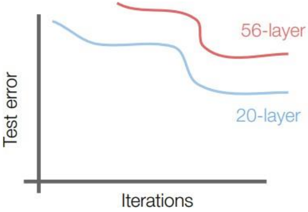
- What happens when we continue stacking deeper layers on a ”plain” convolutional neural network?

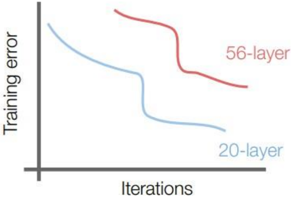


- 56-layer model performs worse on both test and training error

ResNet

- What happens when we continue stacking deeper layers on a ”plain” convolutional neural network?





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

ResNet

- **Fact:** Deep models have more representation power (more parameters) than shallower models.

ResNet

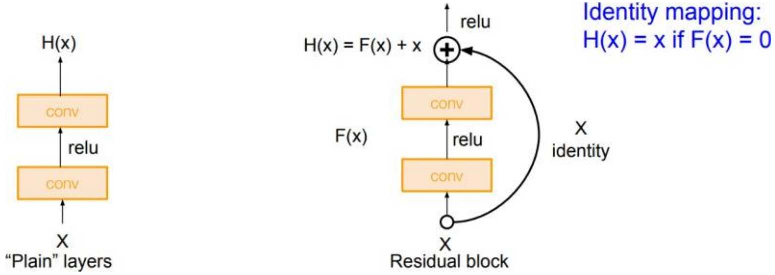
- **Fact:** Deep models have more representation power (more parameters) than shallower models.
- **Hypothesis:** The problem is an optimization problem, deeper models are harder to optimize

ResNet

- **Fact:** Deep models have more representation power (more parameters) than shallower models.
- **Hypothesis:** The problem is an optimization problem, deeper models are harder to optimize
- **Solution:** Use network layers to fit a residual mapping instead of directly trying to fit a desired underlying mapping

ResNet

- **Fact:** Deep models have more representation power (more parameters) than shallower models.
- **Hypothesis:** The problem is an optimization problem, deeper models are harder to optimize
- **Solution:** Use network layers to fit a residual mapping instead of directly trying to fit a desired underlying mapping

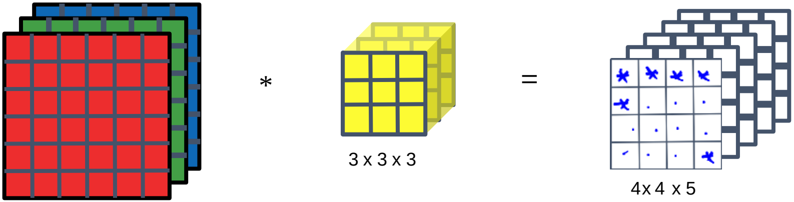


Motivation for MobileNets

أكاديمية كاوست
KAUST ACADEMY

- Low computational cost at deployment
- Useful for mobile and embedded vision applications
- Key idea: Normal vs. depthwise-separable convolutions

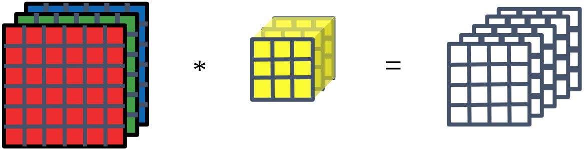
Normal Convlution



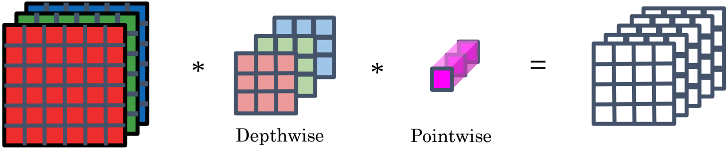
Computational cost = #filter params x # filter positions x # of filters

Depthwise separable convolution

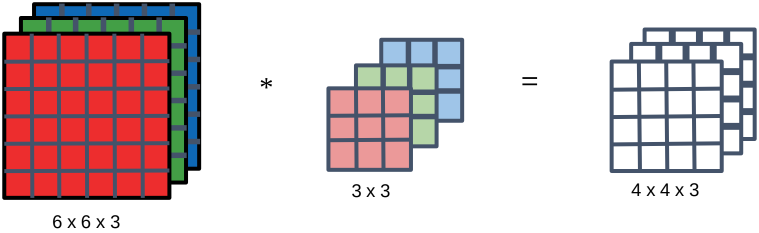
Normal Convolution



Depthwise Separable Convolution

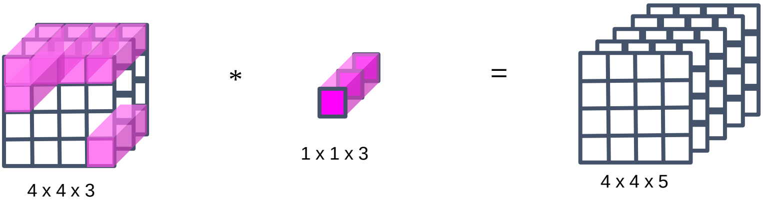


Depthwise convolution



Computational cost = #filter params x # filter positions x # of filters

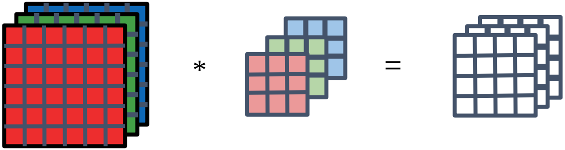
Pointwise convolution



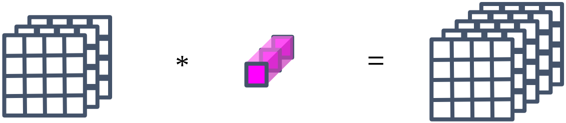
Computational cost = #filter params x # filter positions x # of filters

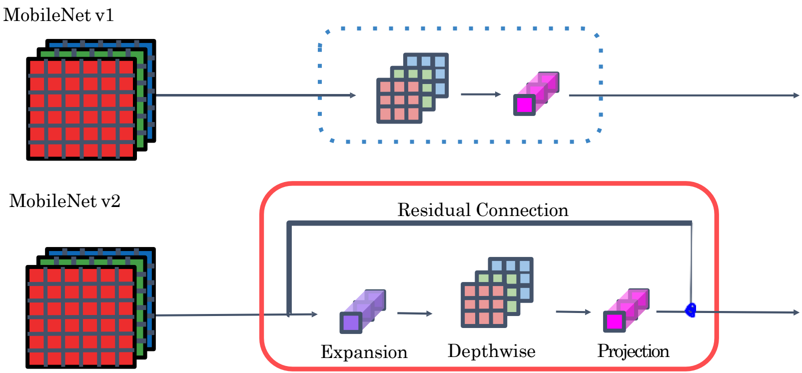
Depthwise separable convolution

Depthwise Convolution



Pointwise Convolution



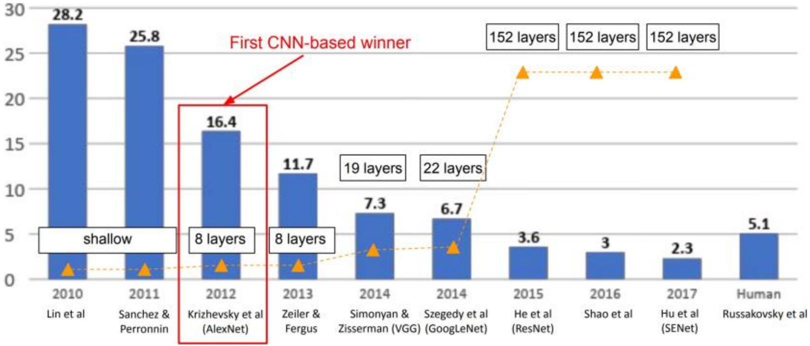


[Sandler et al. 2019, MobileNetV2: Inverted Residuals and Linear Bottlenecks]

ImageNet

- The most extensive data for Image Classification
- 3 RGB channels from 0 to 255
- 14,197,122 images
- 1000 classes

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

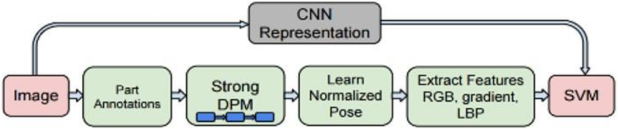


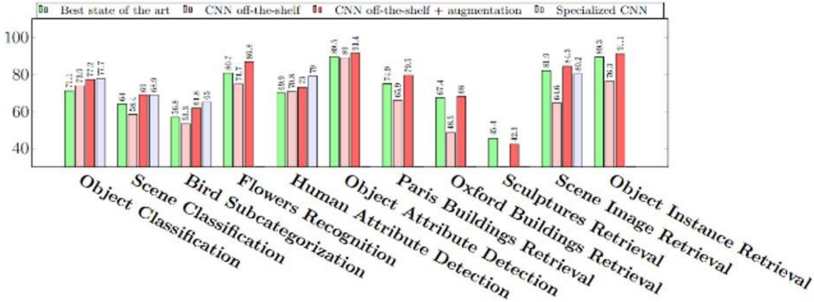
Transfer Learning

- Improvement of learning in a **new** task through the transfer of knowledge from a **related** task that has already been learned.
- We will look at one strategy of transfer learning called Fine-Tuning

When to fine-tune your model?

- New dataset is small with distribution similar to original dataset.
  - Keep the feature extraction part fixed and fine-tune the classifier part of the network
- New dataset is large with similar distribution to the original dataset
  - Fine tune both the feature extractor and the classifier part of the network
- New dataset is small but different distribution from the original dataset
  - Use SVM classifier on the features extracted from the feature extractor part of the Network
- New dataset is large and different distribution from the original dataset
  - Fine tune both the feature extractor and the classifier part of the network





[0] Razavian et al. 2014

Finetuning

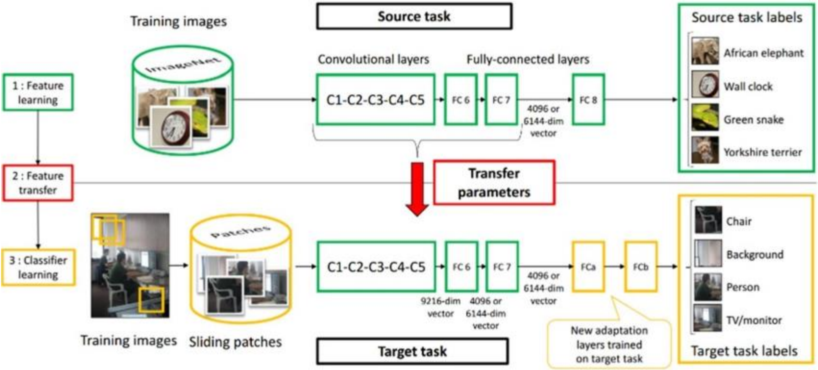


Figure 2: Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks

[0] Oquab et al. CVPR 2014

References

These slides have been adapted from

- Fei-Fei Li, Yunzhu Li & Ruohan Gao, Stanford CS231n: Deep Learning for Computer Vision
- Assaf Shocher, Shai Bagon, Meirav Galun & Tali Dekel, WAIC DL4CV Deep Learning for Computer Vision: Fundamentals and Applications
- Justin Johnson, UMich EECS 498.008/598.008: Deep Learning for Computer Vision
- Sander Dieleman, Deepmind: Deep Learning Lecture Series 2020