Deep learning

Convolutional Neural Networks (CNN)

Wouter Gevaert & Marie Dewitte

Overview

CNN applications

Motivation for the use of a CNN

Architecture of a CNN

Advanced CNN architectures

Object detection

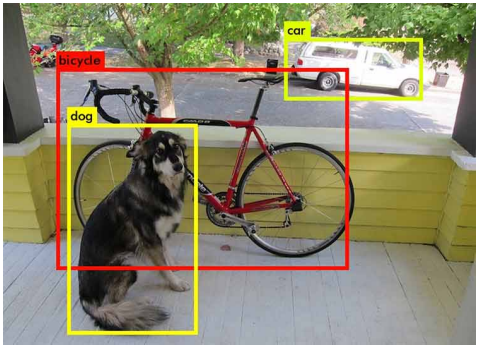
CNN applications

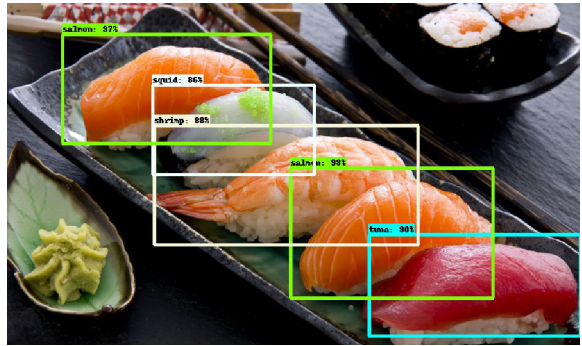
Applications

Classification

Applications

Detection





Applications

Segmentation

Applications

Self-driving cars

https://selfdrivingcars.mit.edu/deeptesla/

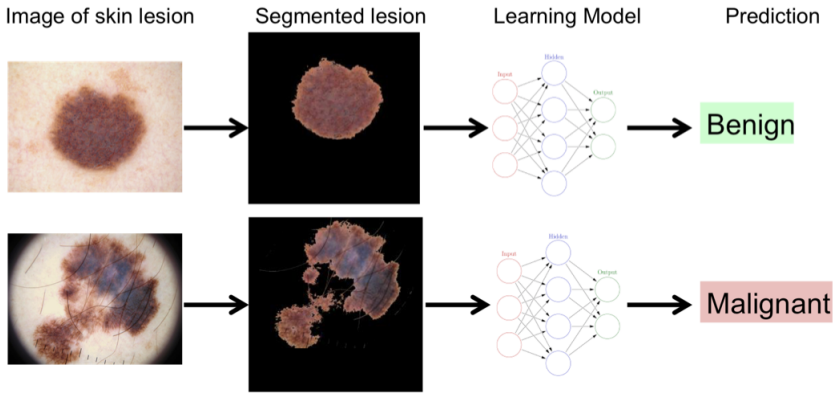
Applications

Pose estimation

https://www.youtube.com/watch?v=pW6nZXeWlGM

Applications

medical diagnoses



https://web.stanford.edu/~kalouche/docs/Vision_Based_Classification_of_Skin_Cancer_using_Deep_Learning_(Kalouche).pdf

Applications

Image captioning

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

"girl in pink dress is jumping in air."

"black and white dog jumps over bar."

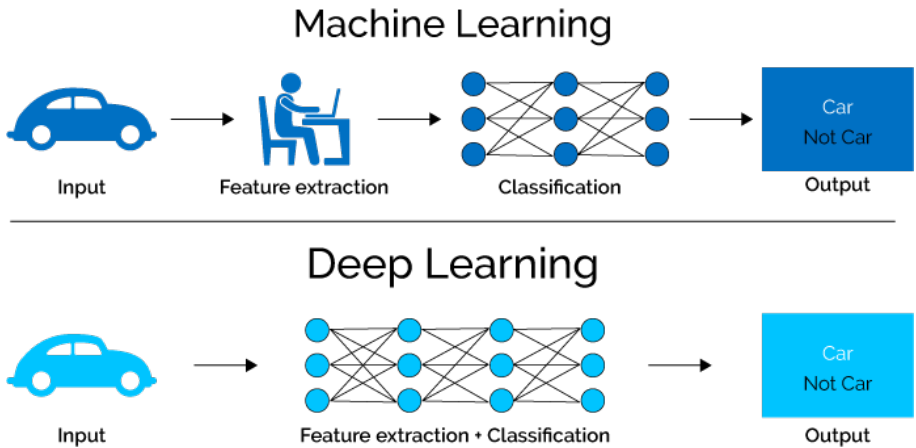
"young girl in pink shirt is swinging on swing."

Applications

Style transfer

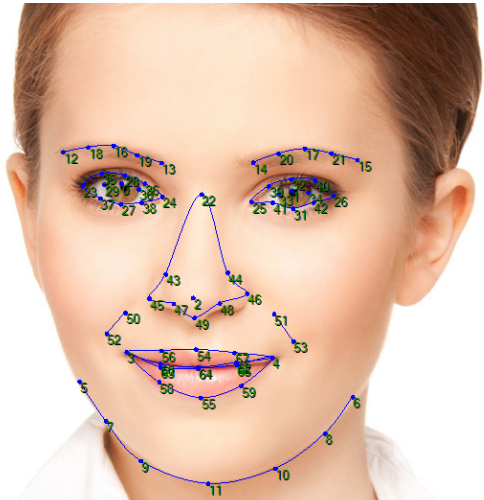
Motivation for the use of a CNN

Feature extraction



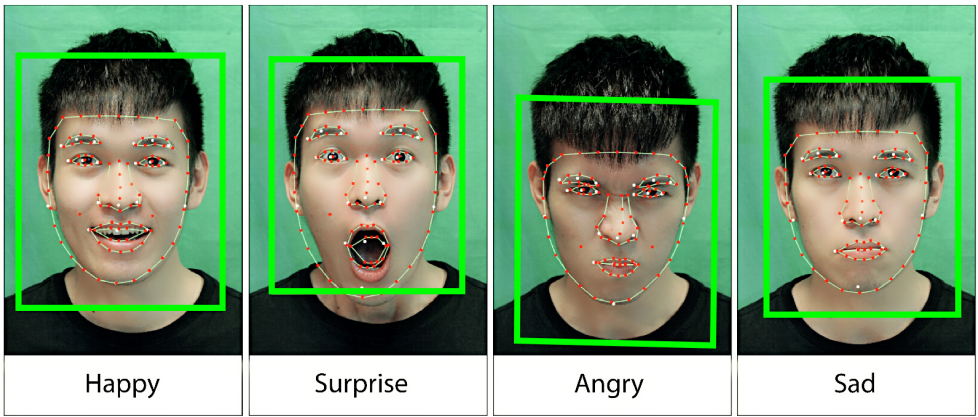
Examples of manual feature extraction

Facial Feature Point Extraction



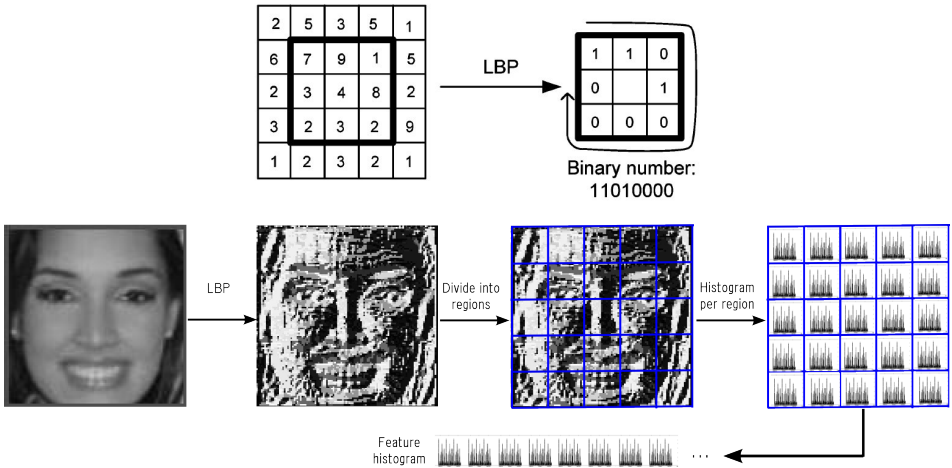
Examples of manual feature extraction

Facial Feature Point Extraction



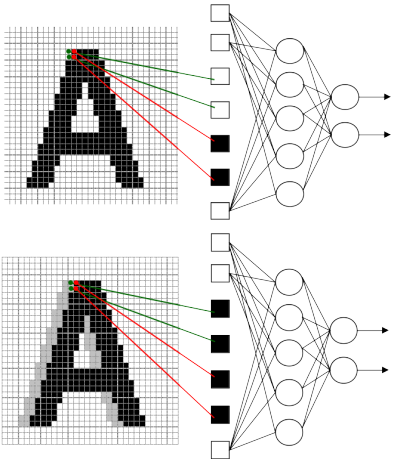
Examples of manual feature extraction

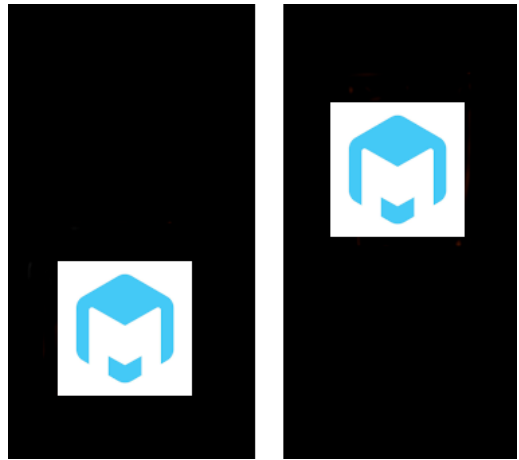
Local Binary Patterns (LBP)



Robustness against (small) variantions in the input

Translations

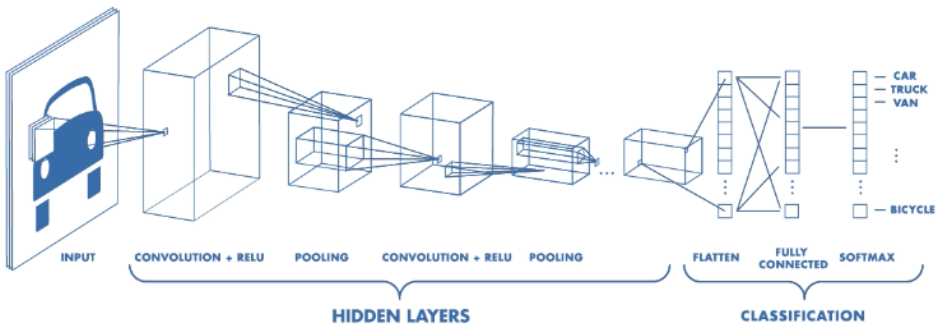




Architecture of a CNN

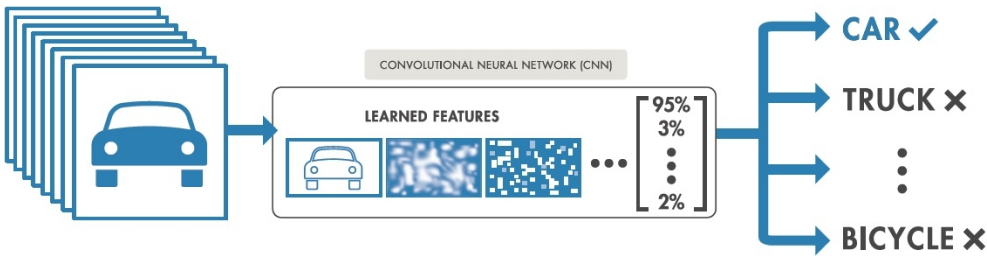
Architecture of a CNN

Overview

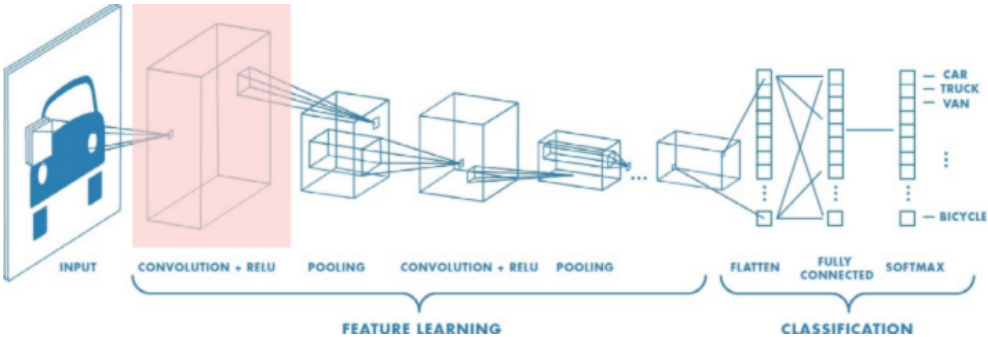


Architecture of a CNN

Feature learning



Convolution



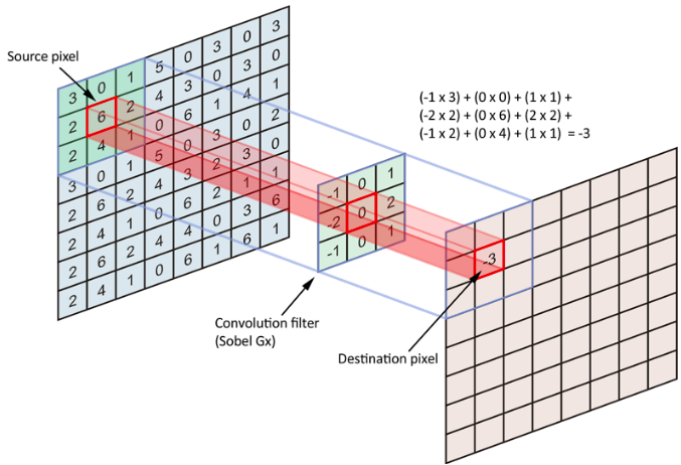
Convolution

Convolution is everywhere

- Signal processing (sound, images, video, ...).
- For advanced systems human designed filters fall short.
- The CNN learns the optimal feature extractor (filter).
- Trained through backprogation.

Convolutions

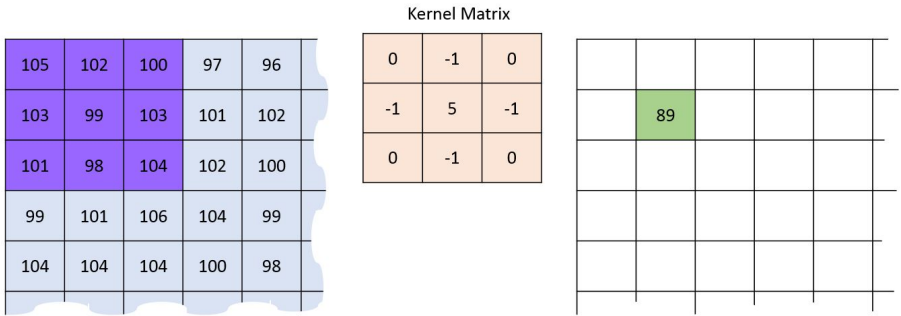
2D convolution



Convolution

2D convolution

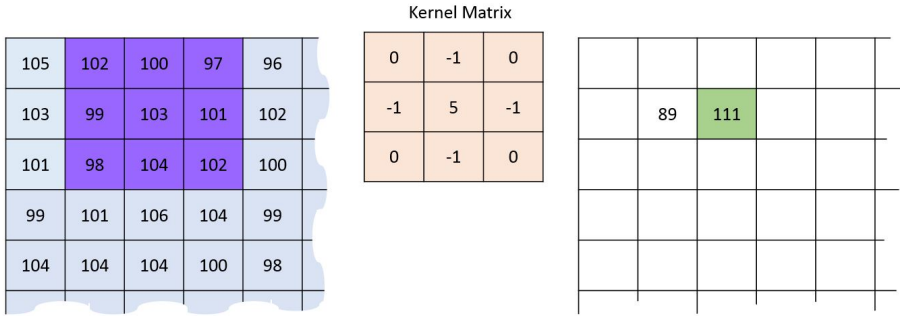
Valid mode: only compute values when the filterkernel and input overlap.



Convolution

2D convolution

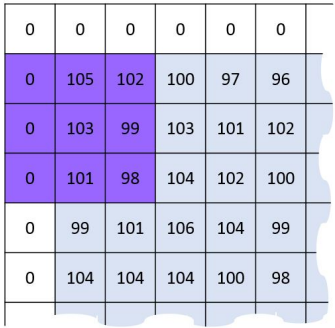
Valid mode: only compute values when the filterkernel and input overlap..



Convolution

2D convolution

Same mode: zero padding such that input length = output length.



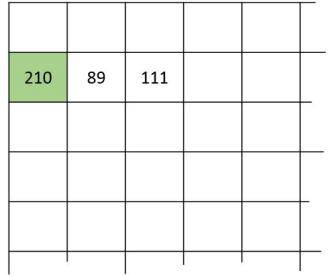
0	0	0	0	0	0
0	105	102	100	97	96
0	103	99	103	101	102
0	101	98	104	102	100
0	99	101	106	104	99
0	104	104	104	100	98

Image Matrix

Kernel Matrix

0	-1	0
-1	5	-1
0	-1	0

$$0*0+105*-1+102*0$$
$$+0*-1+103*5+99*-1$$
$$+0*0+101*-1+98*0 = 210$$



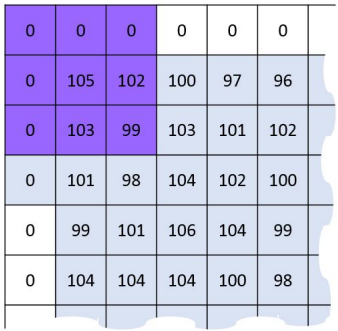
210	89	111			

Output Matrix

Convolution

2D convolution

Same mode: zero padding such that input length = output length.



0	0	0	0	0	0
0	105	102	100	97	96
0	103	99	103	101	102
0	101	98	104	102	100
0	99	101	106	104	99
0	104	104	104	100	98

Image Matrix

Kernel Matrix

0	-1	0
-1	5	-1
0	-1	0

$$0*0+0*-1+0*0$$
$$+0*-1+105*5+102*-1$$
$$+0*0+103*-1+99*0 = 320$$

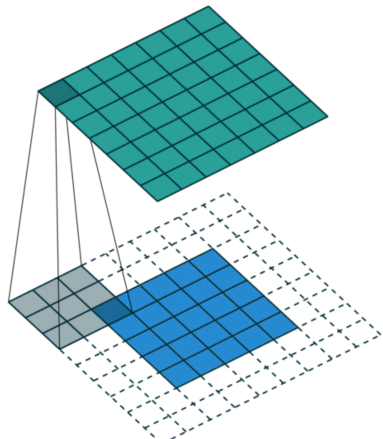
320					
210	89	111			

Output Matrix

Convolution

2D convolution

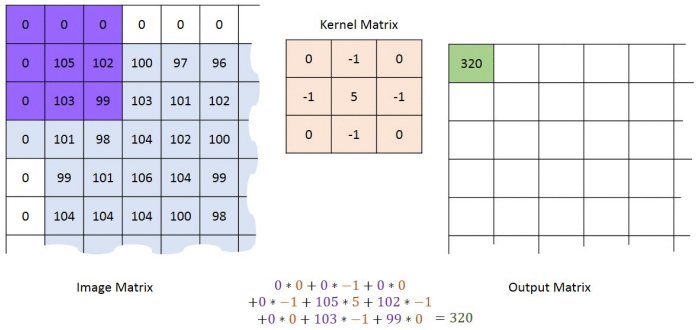
Full mode: zero padding such that filterkernel has a minimum overlap of 1.



Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

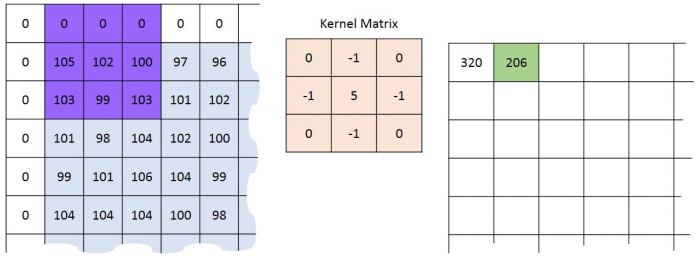


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

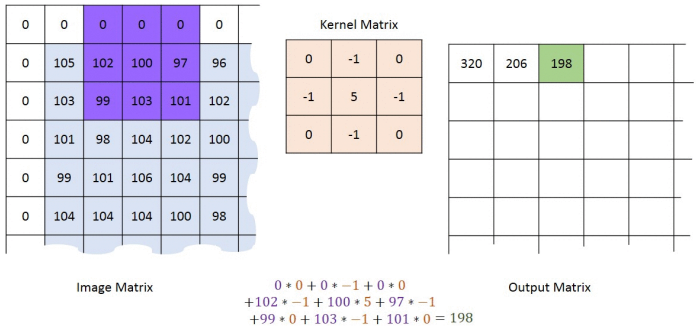


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

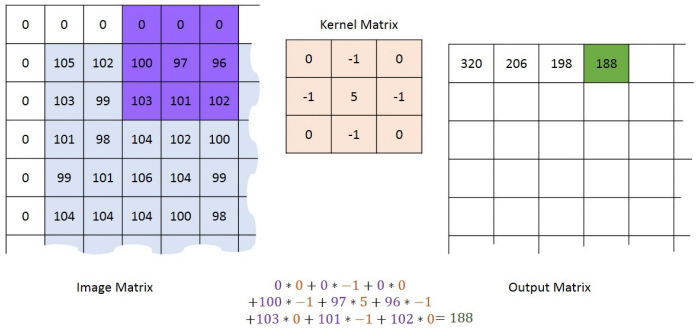


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

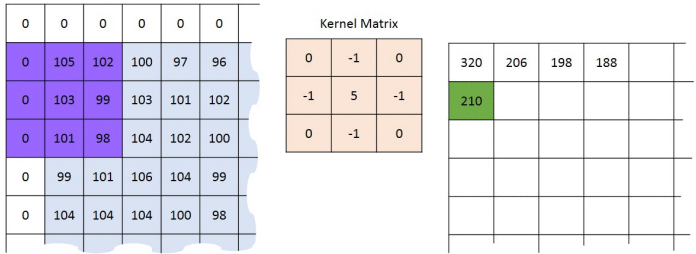


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

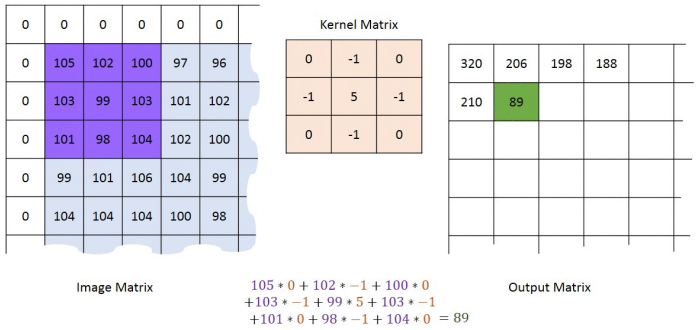


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

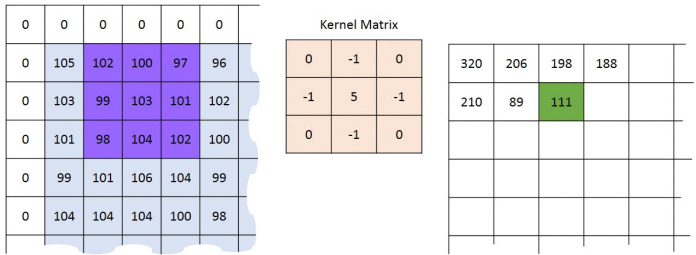


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

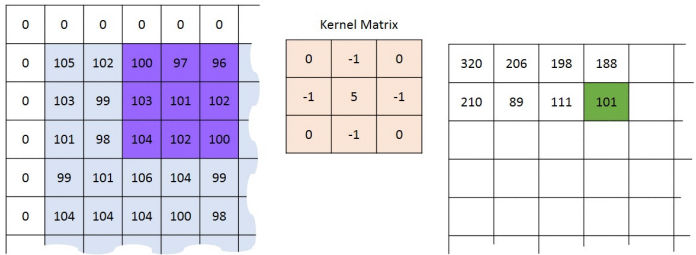


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

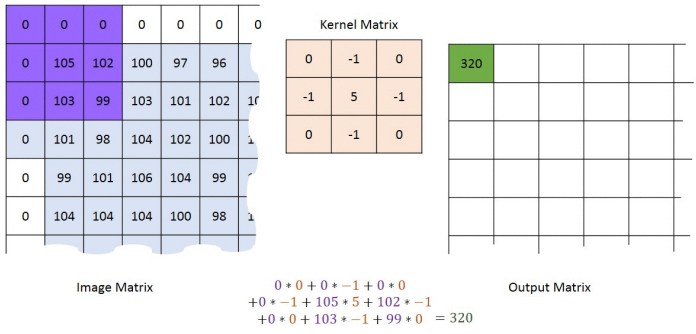


Convolution with horizontal and vertical strides = 1

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

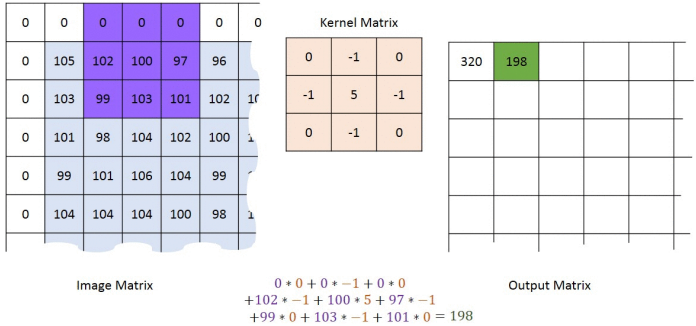


Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

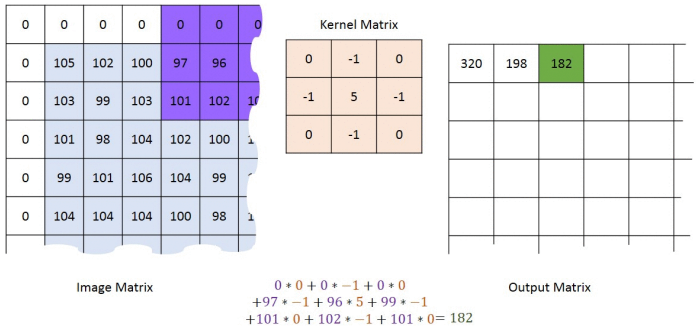


Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

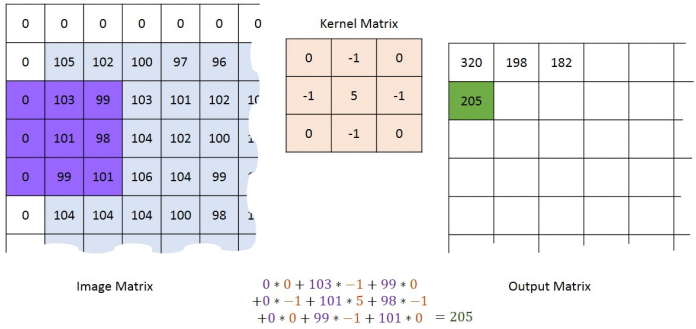


Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

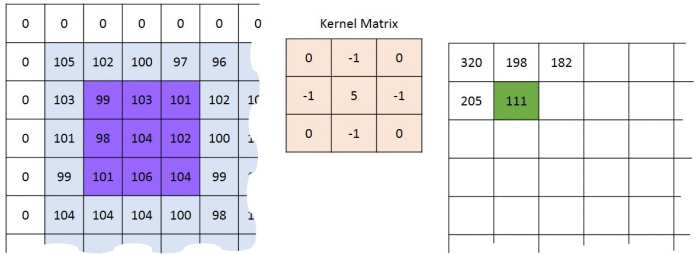


Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

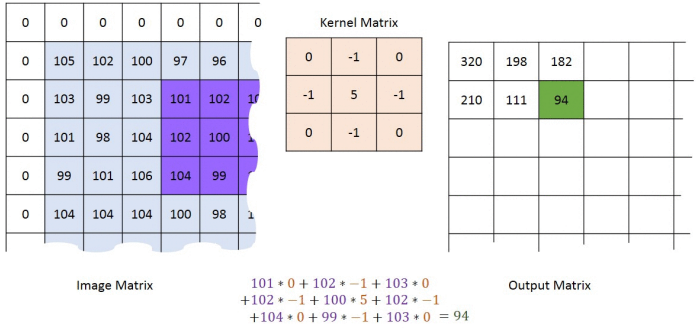


Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

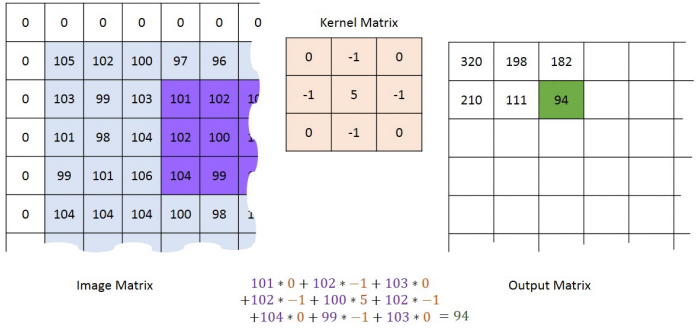


Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.

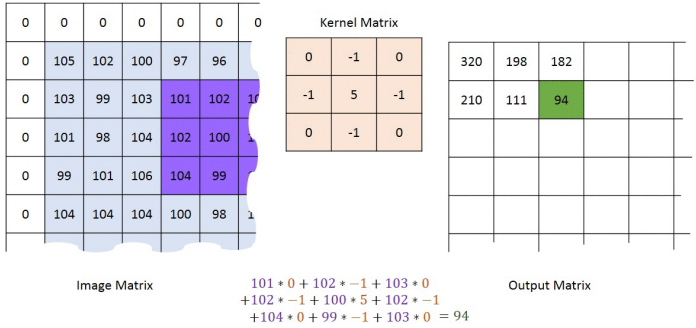


Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Stride: the number of pixels by which the filter shifts.



Convolution with horizontal and vertical strides = 2

Convolution

2D convolution

Dimensions of the output matrix

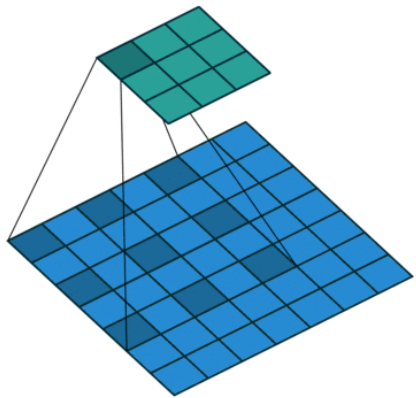
W = input width
H = input height
K_w = kernel width
K_h = kernel height
P = amount of zero padding
S_w = horizontal stride
S_h = vertical stride

$$\text{output width} = \frac{W - K_w + 2P}{S_w} + 1$$

$$\text{output height} = \frac{H - K_h + 2P}{S_h} + 1$$

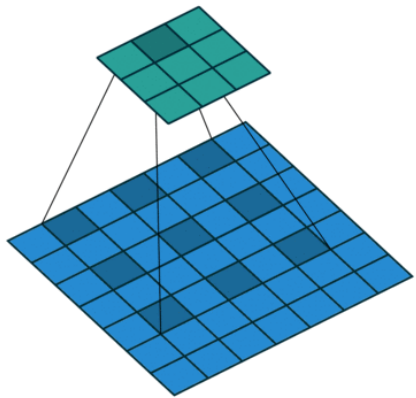
Convolution

Dilated convolution



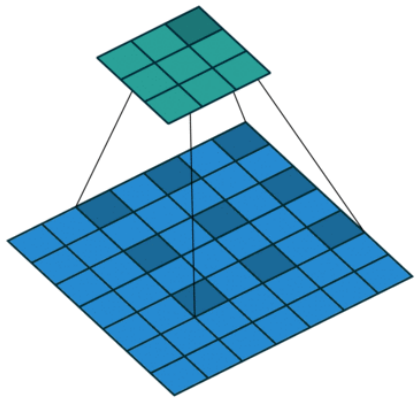
Convolution

Dilated convolution



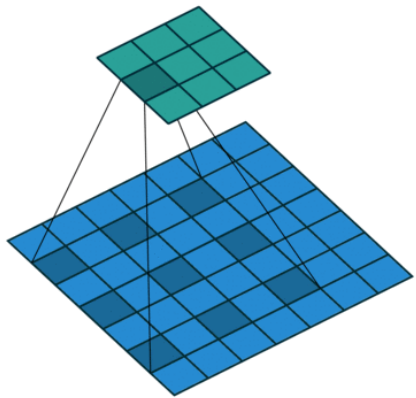
Convolution

Dilated convolution



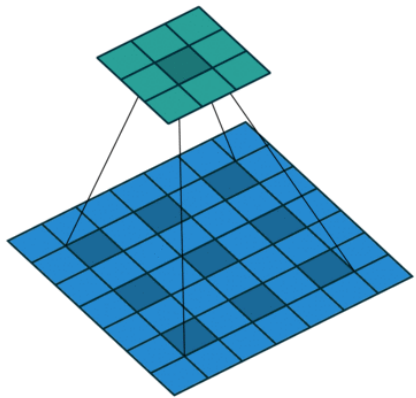
Convolution

Dilated convolution



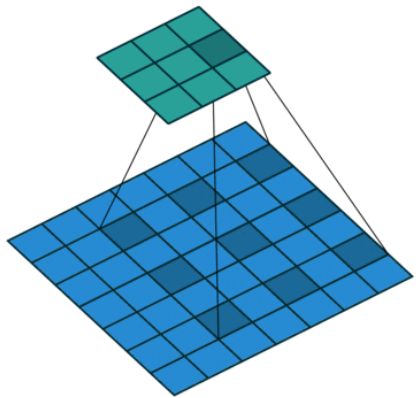
Convolution

Dilated convolution



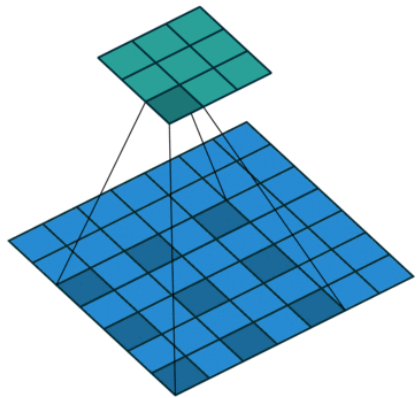
Convolution

Dilated convolution



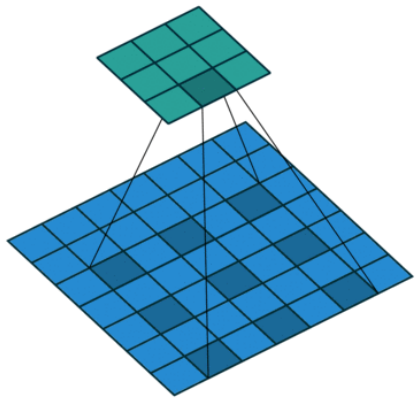
Convolution

Dilated convolution



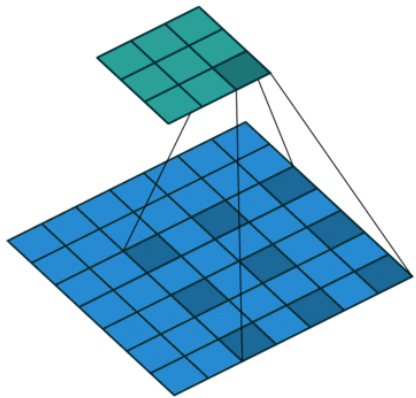
Convolution

Dilated convolution



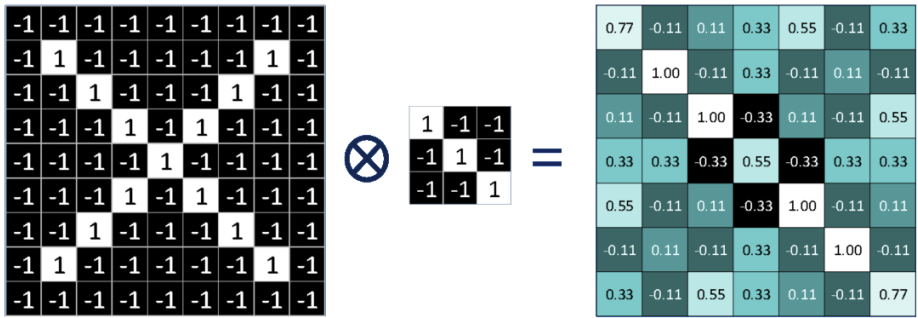
Convolution

Dilated convolution



Convolution

Convolution = pattern machting



https://youtu.be/FmpDIaiMIeA

Convolution

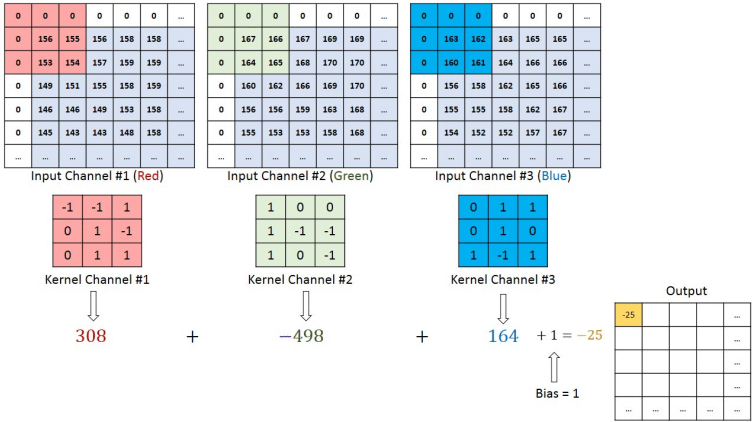
3D convolution

Tensorflow: tensor with a dimension $(H \times W \times C)$

Convolution

3D convolution

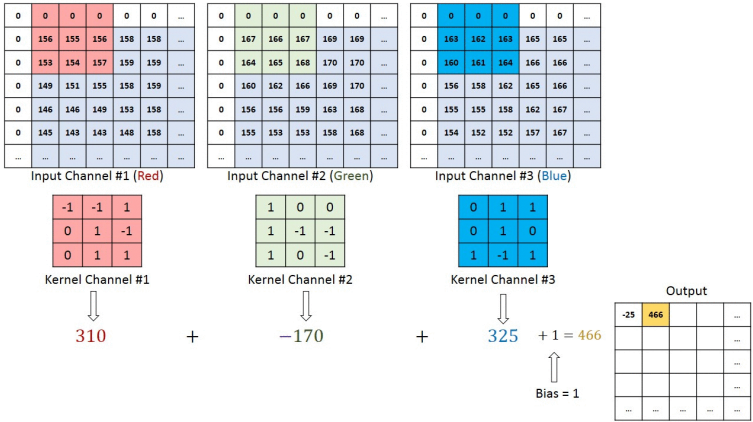
Example:



Convolution

3D convolution

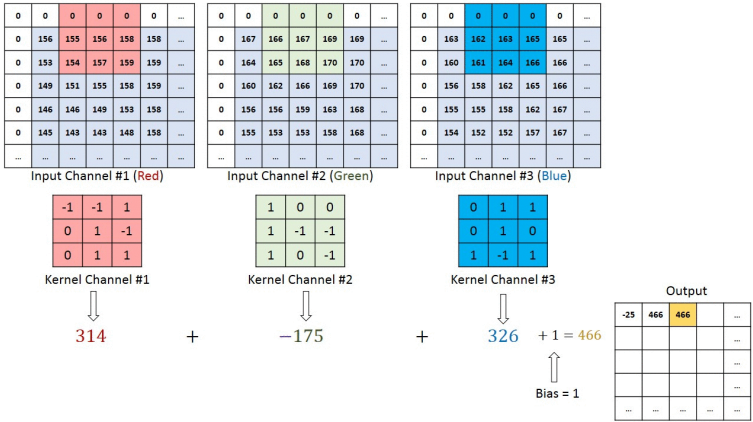
Example:



Convolution

3D convolution

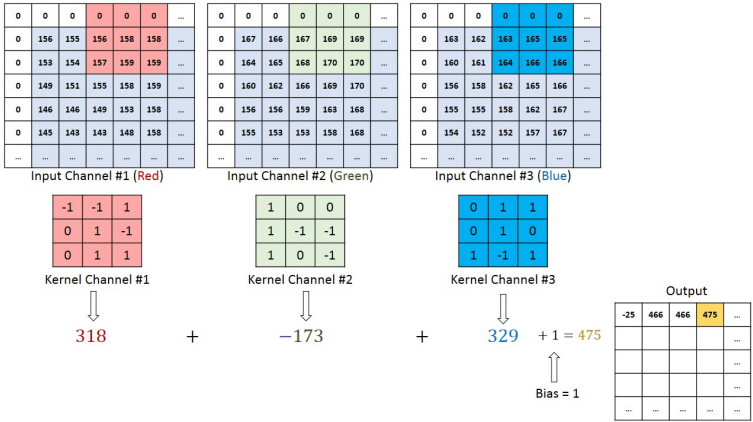
Example:



Convolution

3D convolution

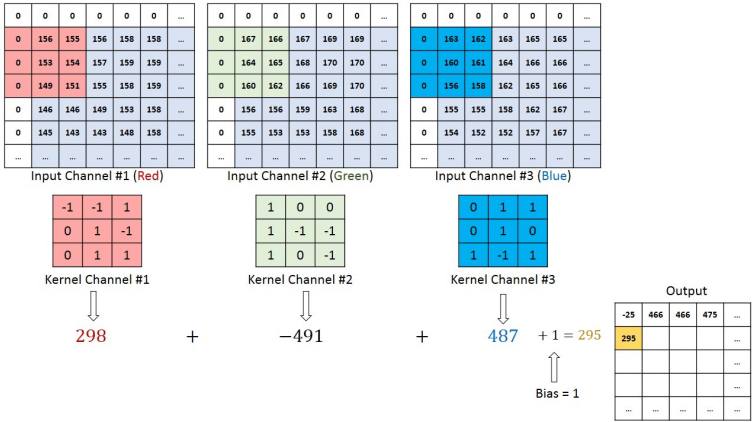
Example:



Convolution

3D convolution

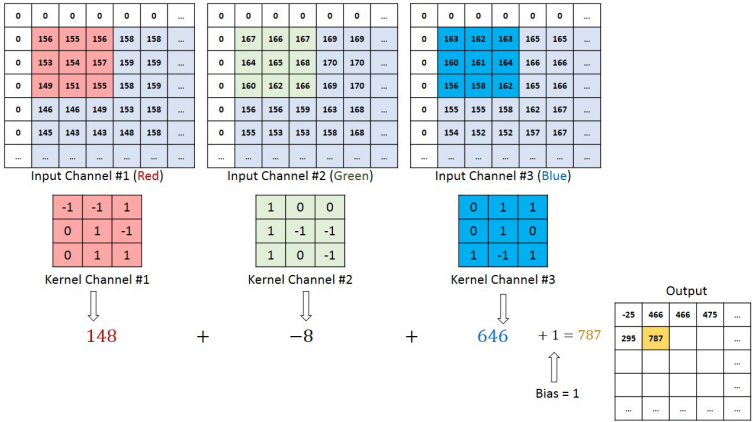
Example:



Convolution

3D convolution

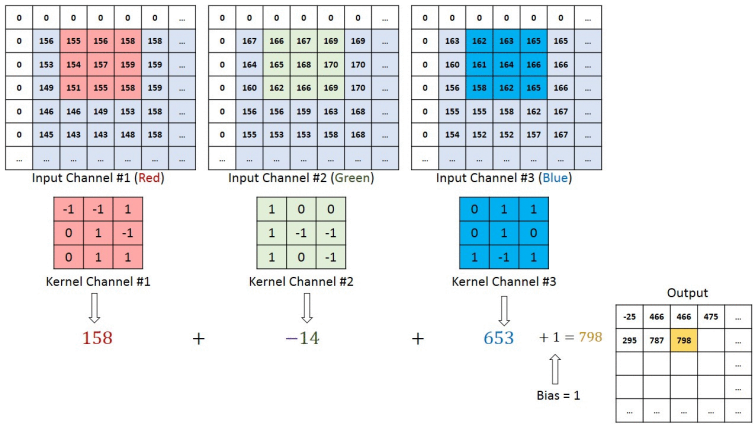
Example:



Convolution

3D convolution

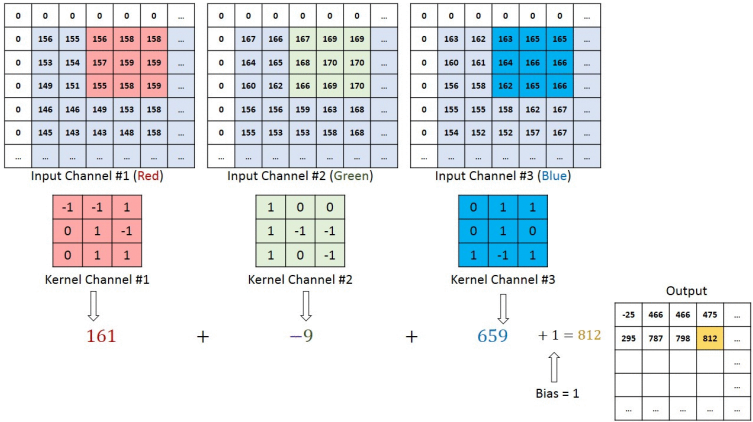
Example:



Convolution

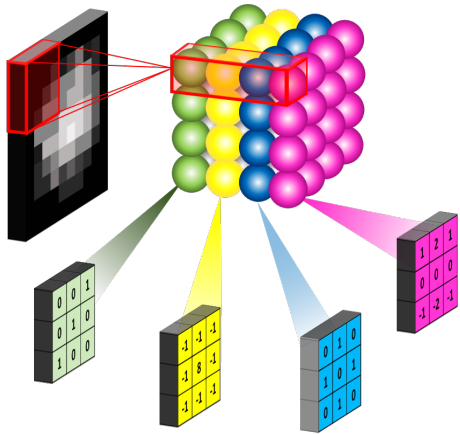
3D convolution

Example:



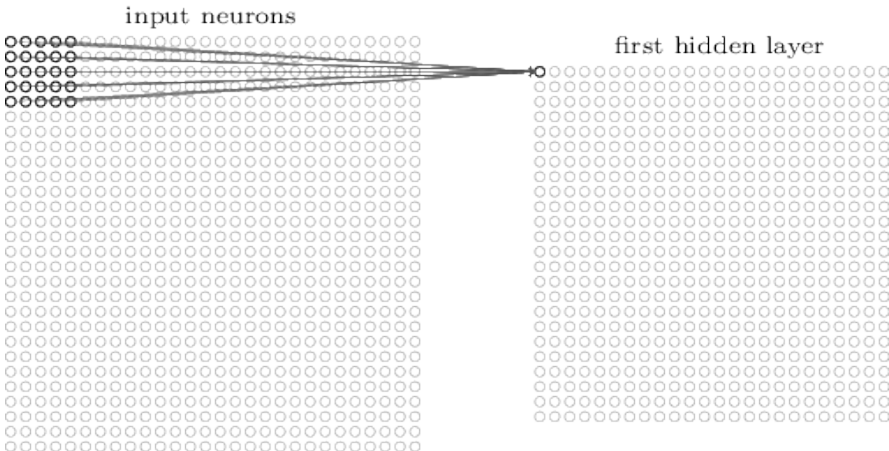
Convolution

3D convolution



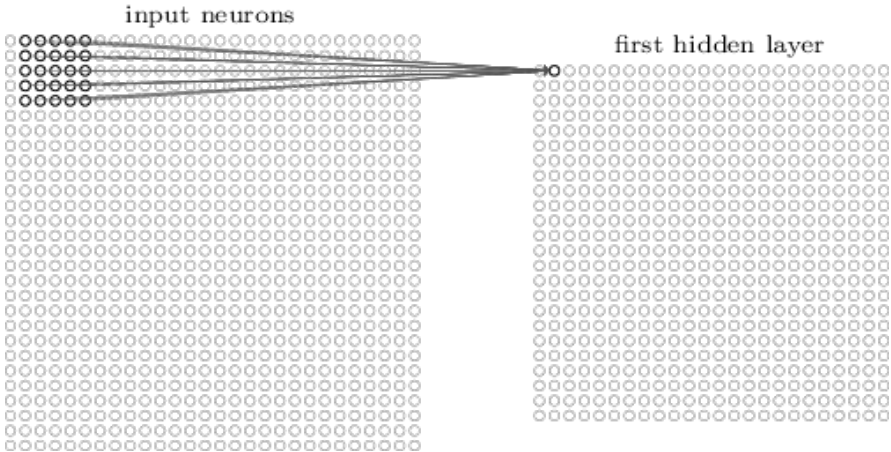
Convolutional hidden layer

Local receptive fields and weight sharing



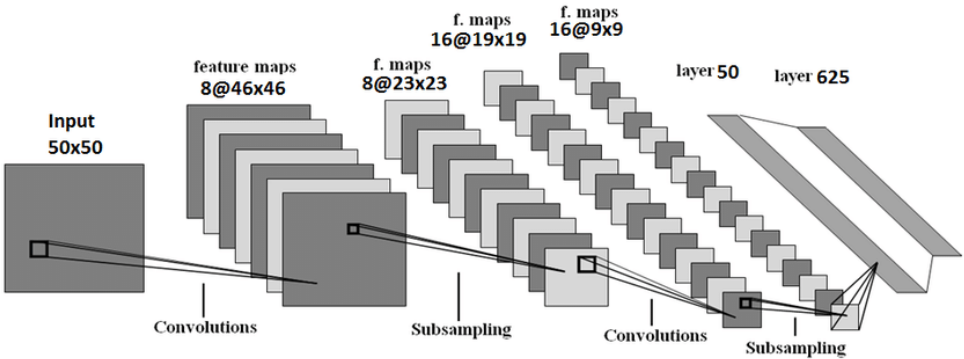
Convolutional hidden layer

Local receptive fields and weight sharing



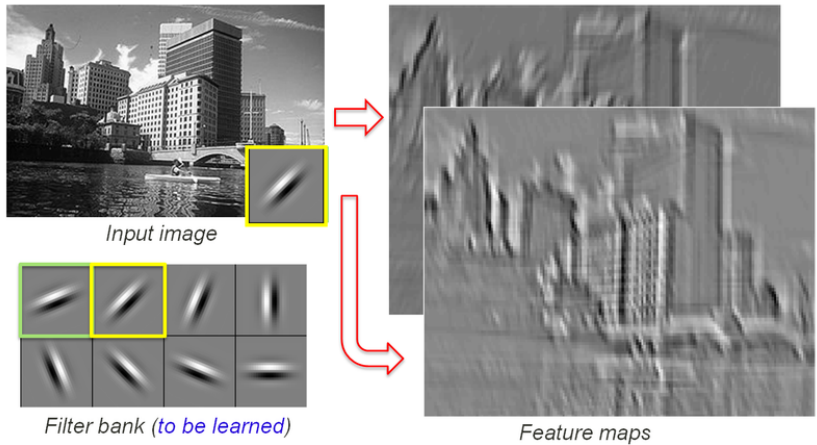
Convolutional hidden layer

Local receptive fields and weight sharing



Convolutional hidden layer

Local receptive fields and weight sharing

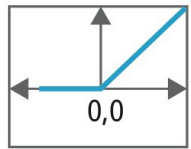


Activation function

ReLu

15	20	-10	35
18	-110	25	100
20	-15	25	-10
101	75	18	23

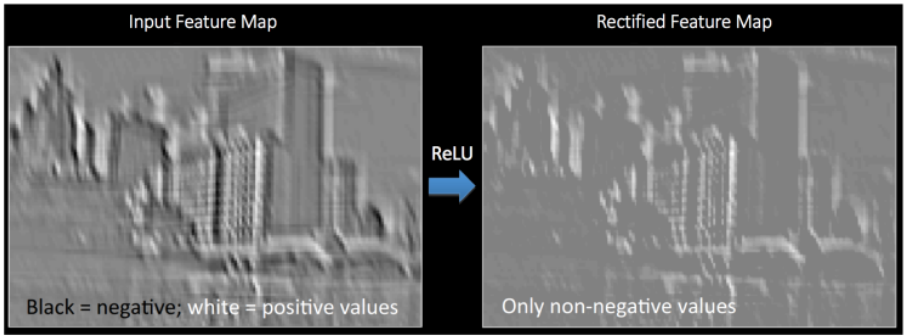
Transfer Function



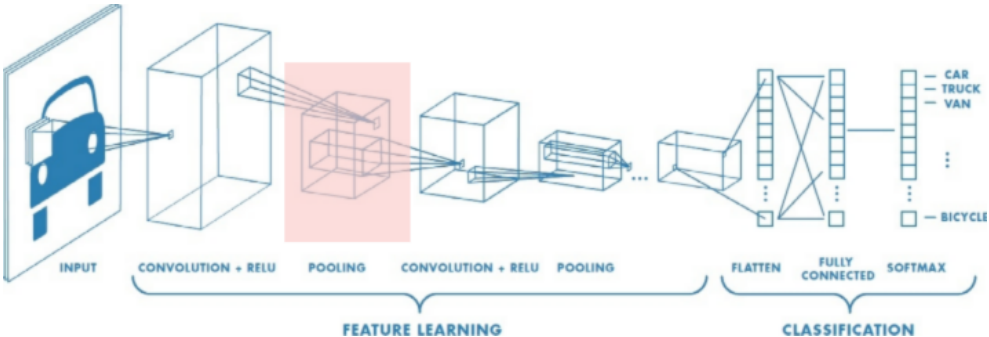
15	20	0	35
18	0	25	100
20	0	25	0
101	75	18	23

Activation function

ReLu

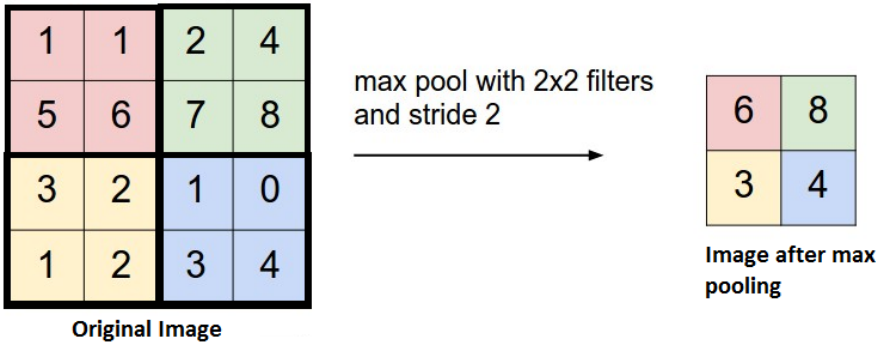


Pooling


CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
FEATURE LEARNING
CLASSIFICATION

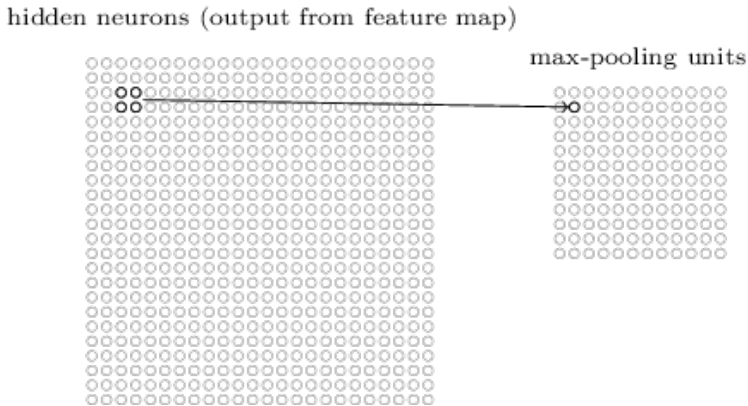
Pooling

Max pooling



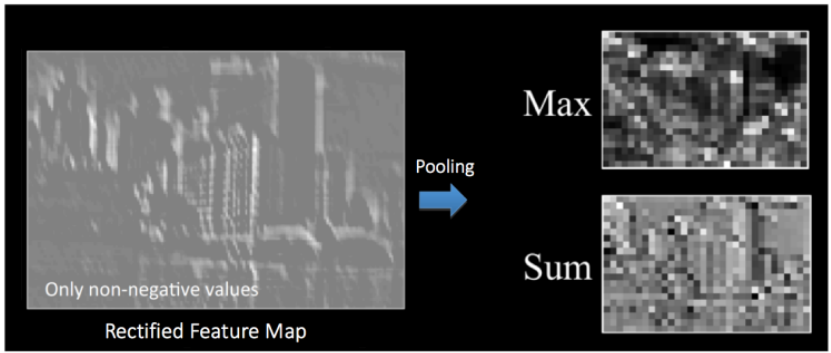
Pooling

Max pooling



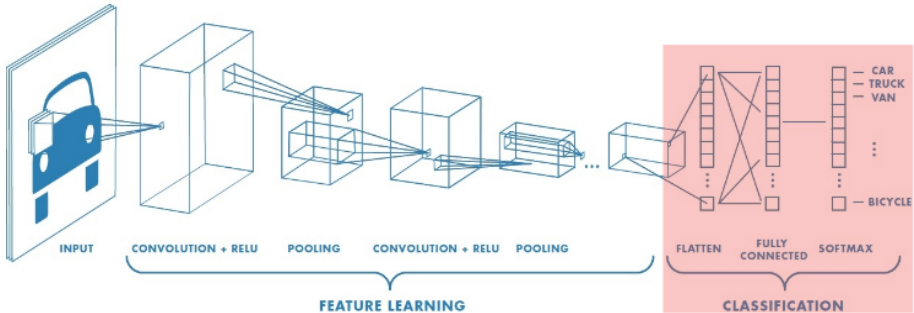
Pooling

Max pooling



Flattening & fully connected layer

Classification of the extracted features



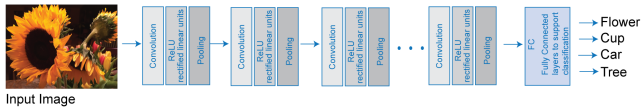
- Flattening: reshape to 1D (vector).
- Very often the hidden layers have the same shape.
- Output is usually softmax.

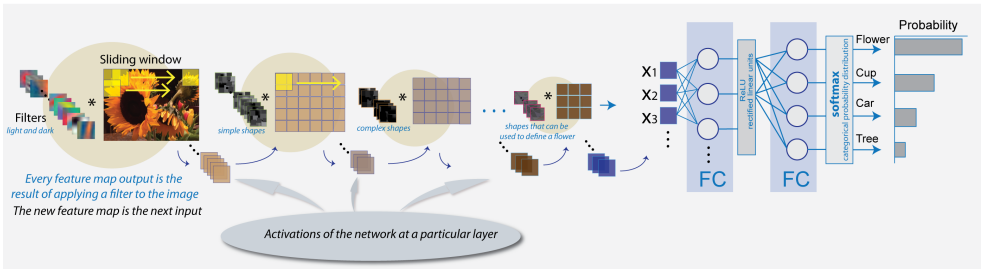
Dropout

Regularization

- Often not necessary as CNN has less weights.
- Dropout not very efficient after convolutional layers.
- `https://arxiv.org/pdf/1506.02158v6.pdf`

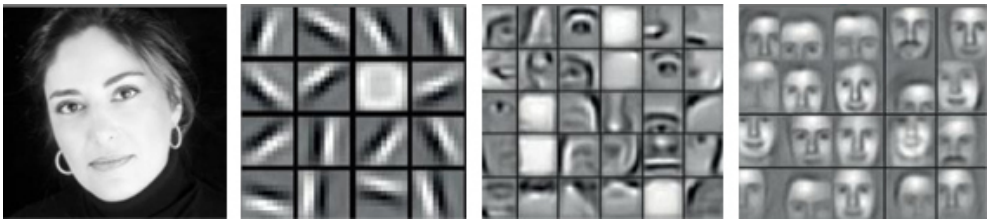
What representations a CNN learns





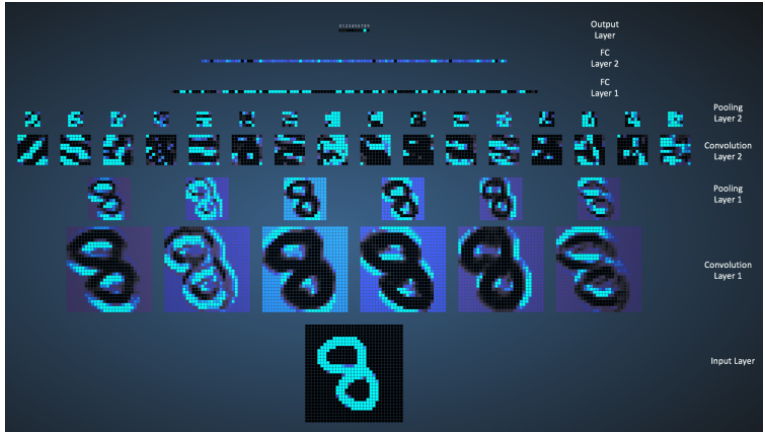
What representations a CNN learns

Face recognition



What representations a CNN learns

MNIST



http://scs.ryerson.ca/~aharley/vis/conv/flat.html

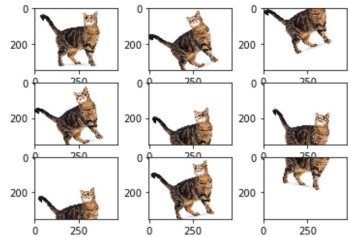
What representations a CNN learns

Deep Tesla

https://selfdrivingcars.mit.edu/deeptesla/

Image augmentation

A popular technique largely used to enhance the training of a CNN

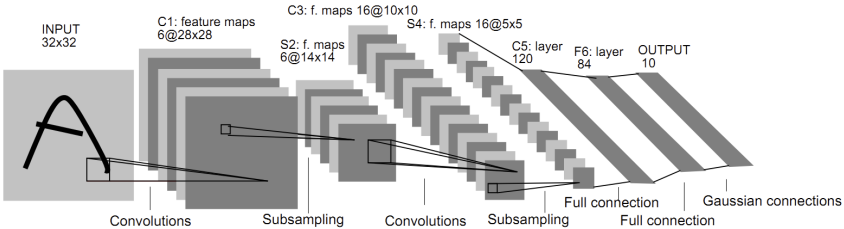


- Translations.
- Rotations.
- Size.
- Color.
- Background
- Noise

Advanced CNN architectures

Advanced CNN architectures

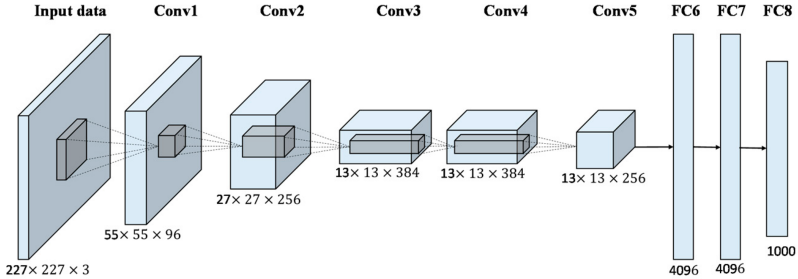
LeNet-5 (1998)



- Invented by Yann LeCun (father of convolutional neural networks).
- Originally intended for handwriting recognition.
- 60000 parameters.
- Paper: `http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf`

Advanced CNN architectures

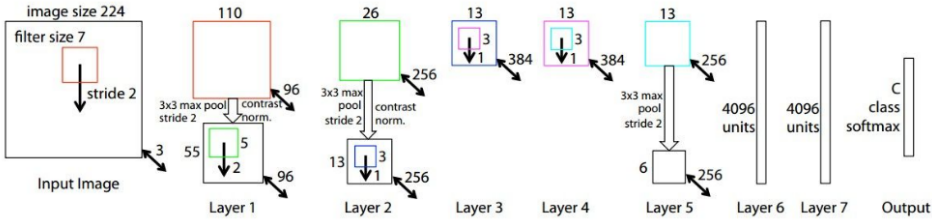
AlexNet (2012)



- Created by Alex Krizhevsky.
- Similar to LeNet but deeper: 60 000 000 parameters
- Paper: `https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf`

Advanced CNN architectures

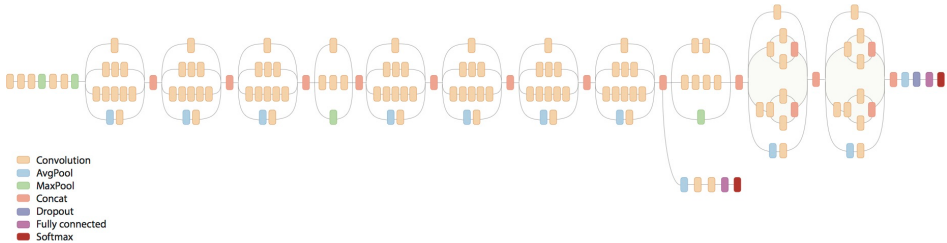
ZFNet (2013)



- Similar to AlexNet with tweaked parameters.
- Smaller filter kernels (in the first layers) than AlexNet.
- AlexNet trained on 15 000 000 images, ZFNet on 1 300 000 afbeeldingen.
- Paper: `https://cs.nyu.edu/~fergus/papers/zeilerECCV2014.pdf`

Advanced CNN architectures

GoogLeNet / Inception (2014)

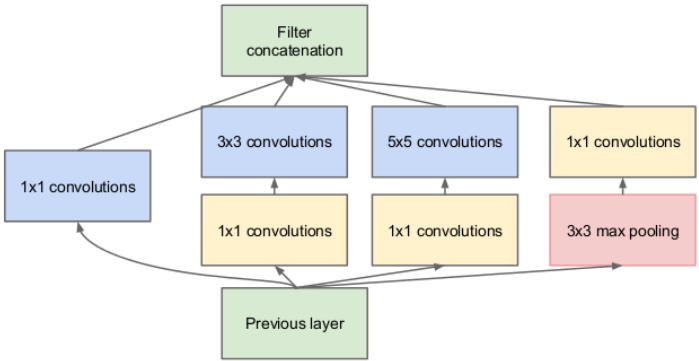


- Uses image augmentation, batch normalization and RMSProp.
- Based on LeNet + **inception module**.
- Combine convolutions with different kernel sizes to extract features with different sizes.
- Paper: `https://arxiv.org/pdf/1409.4842.pdf`

Advanced CNN architectures

GoogLeNet / Inception (2014)

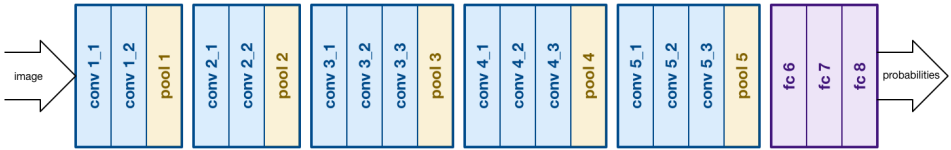
Inception module



- Bron: `https://arxiv.org/abs/1409.4842`

Advanced CNN architectures

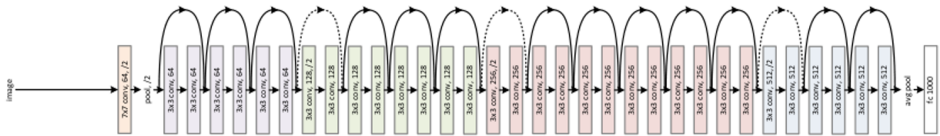
VGG-16 / VGG-19 (2014)



- Visual Geometry Group (Oxford)
- Similar to AlexNet but only 3x3 convolutions.
- Very popular for extracting features from images.
- Weight are publicly available.
- 138 000 000 parameters.
- Paper: `https://arxiv.org/pdf/1409.1556.pdf`

Advanced CNN architectures

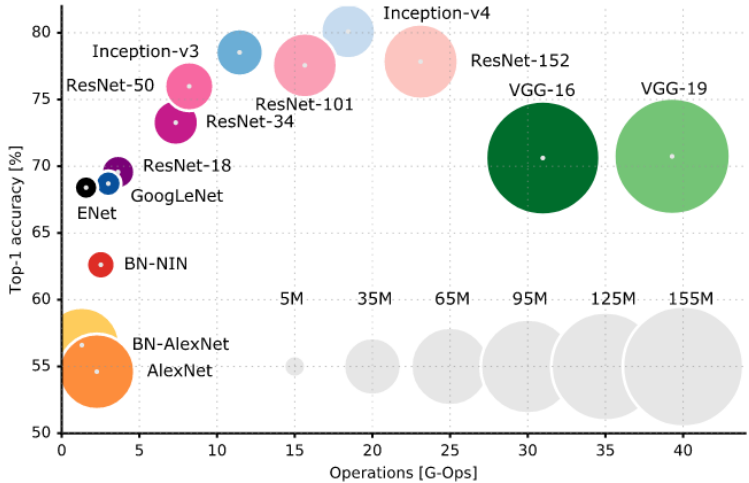
ResNet (2015)



- Residual Neural Network.
- Skip connections (gate units).
- 152 layers (or more) but lower complexity than VGGNet.
- Surpassed human performance on the ImageNet dataset.
- Paper: `https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf`

Advanced CNN architectures

Performance on ImageNet vs Operations



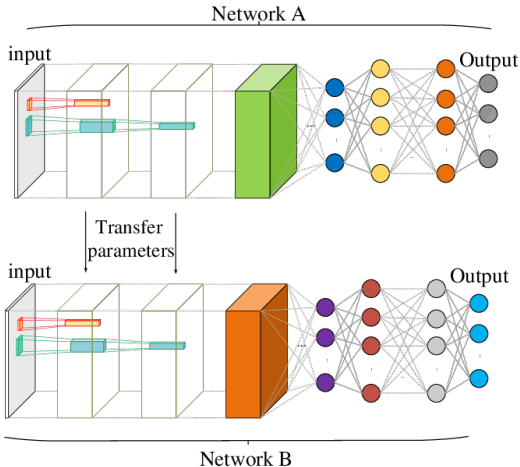
Advanced CNN architectures

Overview

Year	CNN	Developed by	Place	Top-5 error rate	No. of parameters
1998	LeNet(8)	Yann LeCun et al			60 thousand
2012	AlexNet(7)	Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever	1st	15.3%	60 million
2013	ZFNet()	Matthew Zeiler and Rob Fergus	1st	14.8%	
2014	GoogLeNet(19)	Google	1st	6.67%	4 million
2014	VGG Net(16)	Simonyan, Zisserman	2nd	7.3%	138 million
2015	ResNet(152)	Kaiming He	1st	3.6%	

Advanced CNN architectures

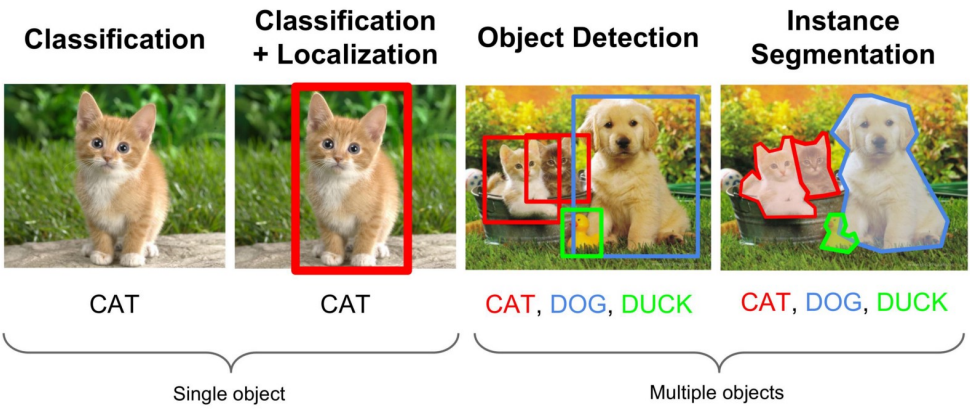
Transfer learning



Object detection

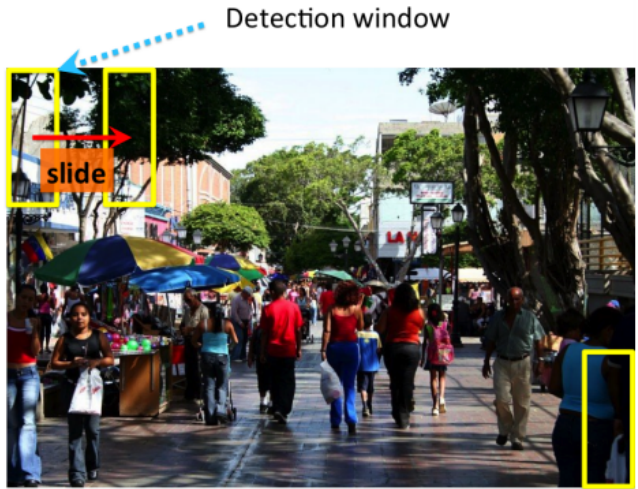
Object detection

Object detectie vs object classification vs object segmentation



Object detection

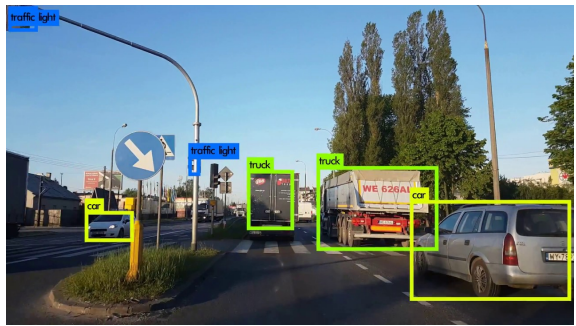
Detectie via sliding window



Object detection

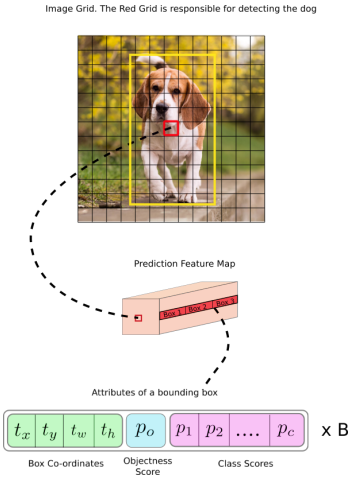
YOLO - You Only Look Once

- State-of-the-art detection.
- real-time.
- No sliding window but a single pass through the neural network.



Object detection

YOLO - You Only Look Once



More info:

- https://pjreddie.com/darknet/yolo/
- https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b