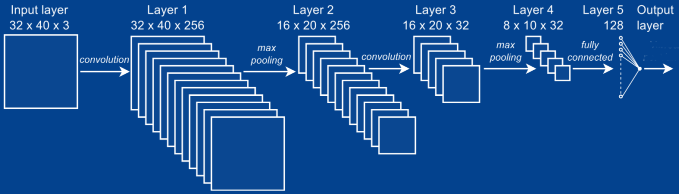
AARHUS UNIVERSITY

LESSON 07: Convolutional Neural Networks

CARSTEN EIE FRIGAARD

FALL 2023



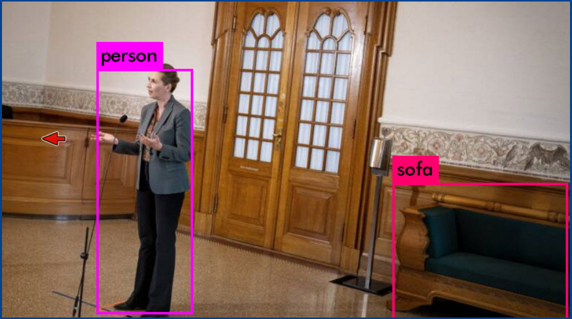
"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." — Mitchell (1997).

L09: CNN's, Agenda

- Convolutional Neural Networks (CNN's),
  - and Deep-learning (DL).
- CNN's In Practice,
  - YOLOV demo
  - The LeNET-5 Architecture.
- GPU-cluster demo.

- Opgave: `L07/CNN.ipynb`

CONVOLUTIONAL NEURAL NETWORKS



Deep-learning (DL)

Artificial Intelligence:

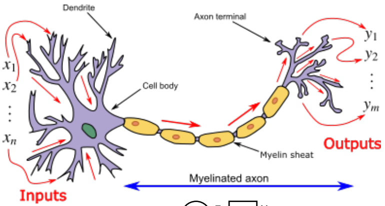
Mimicking the intelligence or behavioural pattern of humans or any other living entity.

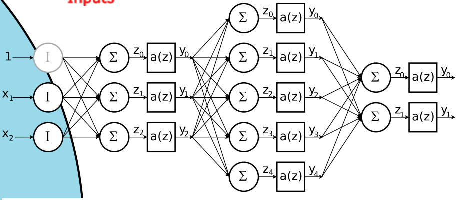
Machine Learning:

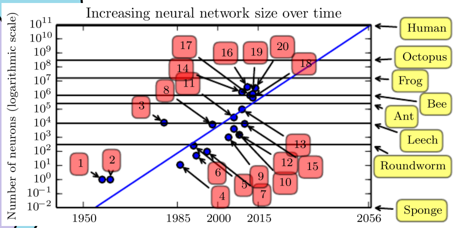
A technique by which a computer can "learn" from data, without using a complex set of different rules. This approach is mainly based on training a model from datasets.

Deep Learning:

A technique to perform machine learning inspired by our brain's own network of neurons.





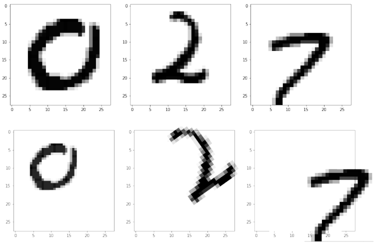


Definition of DL=??

[Deep Learning, Ian Goodfellow et. al., The MIT Press, 2016],
[Prof. Loc Vu-Quoc, commons.wikimedia.org/w/index.php?curid=72816083],
[Tukijaaliwa, commons.wikimedia.org/w/index.php?curid=9013135]

Preprocessing/Feature-extraction + Machine Learning

Can your ML model handle simple image translation (rotation and scaling)?



..no problem for Your visual cortex, right?

Introducing the Convolutional Neural Networks: trade of

- preprocessing/feature-extraction + classification

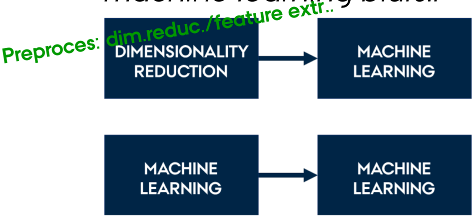
with:

- feature learning (CNN kernels) + classification (fully connected NN)

Convolutional Neural Networks

Feature Extraction vs. Feature Learning

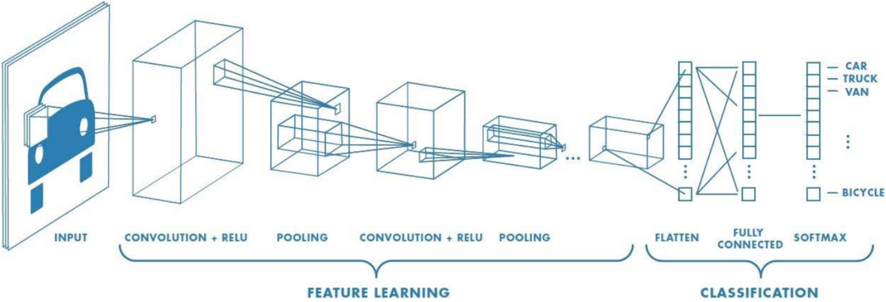
- Smart filtering:
  the distriction bewteen dimensionality reduction and machine learning blurs..

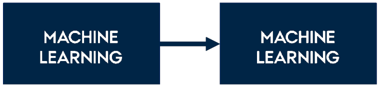


- Fundamental problem of filtering:
  What is noise, what is signal?

Convolutional Neural Networks

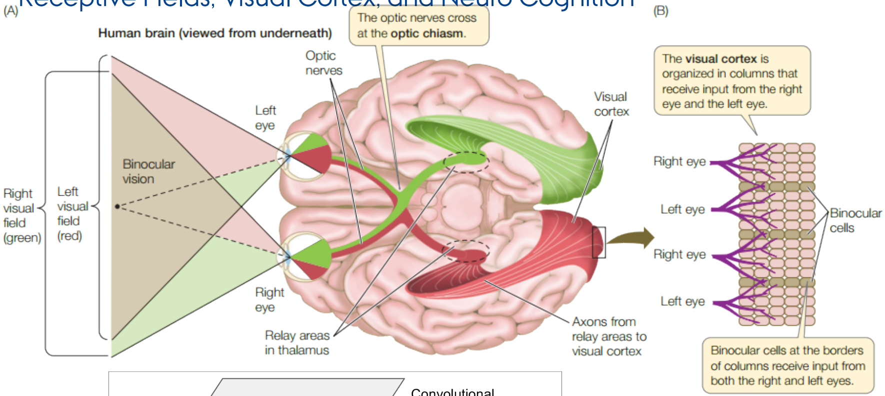
Feature Extraction vs. Feature Learning





Convolutional Neural Networks

Human Eye and Brain Image Processing: Receptive Fields, Visual Cortex, and Neuro Cognition



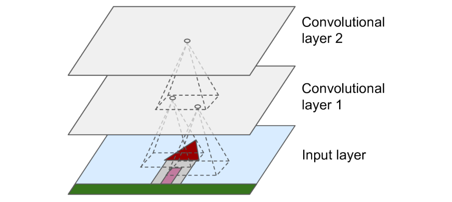


Figure 14-2. CNN layers with rectangular local receptive fields

Feature Learning + Machine Learning

CNN principle

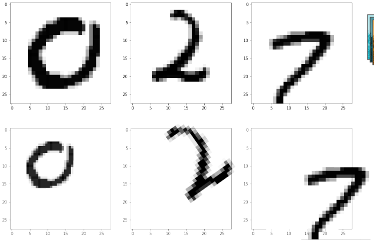
Translation, rotation, and scaling invariant

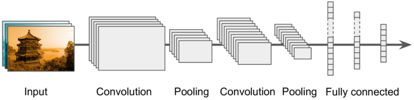
- automatic image feature extraction

that is feature learning via

- 'convolutional', 'pooling' CNN layers,

then a final fully connected NN





..no problem for Your CNN ML model, right?

Feature Learning + Machine Learning

CNN principle: Convolutional Layer (via CNN kernels)

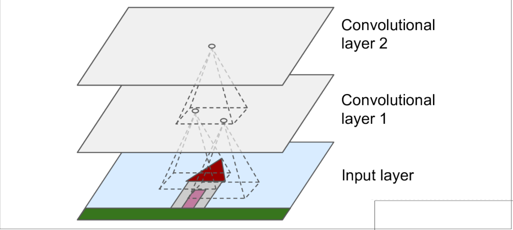


Figure 14-2. CNN layers with rectangular local receptive fields

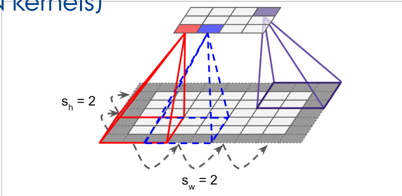


Figure 14-4. Reducing dimensionality using a stride of 2

Feature map 1 => with kernel 1

Feature map 2 => with kernel 2

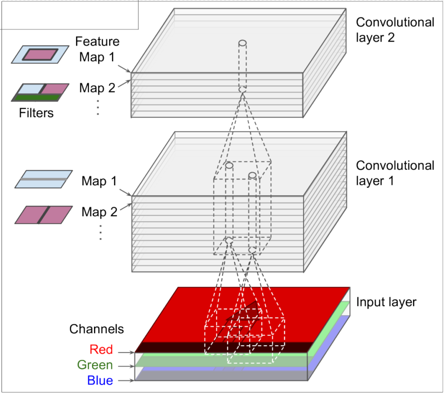


Figure 14-6. Convolution layers with multiple feature maps, and images with three color

CNN Kernels

2D Convolution with a Kernel: Principle

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

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

114				

5*60 + (-1)*113 + (-1)*73 = 114

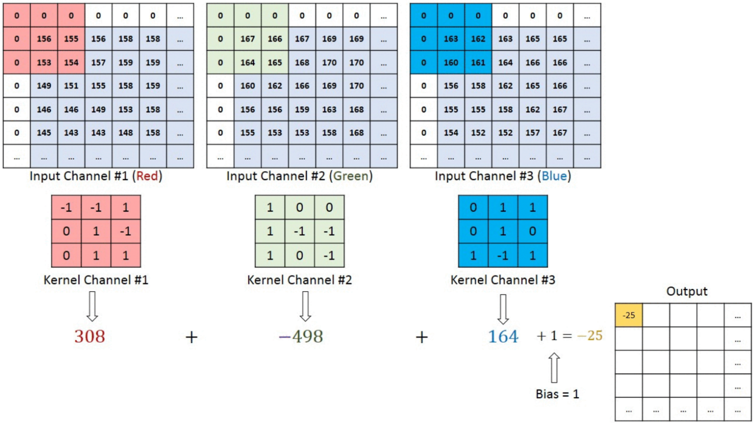
CNN Kernels

2D Convolution with a Kernel: Different Kernels

NOTE: GIMP demo on grey-scale Lenna: *Filters | Generic | Convolution Matrix..*

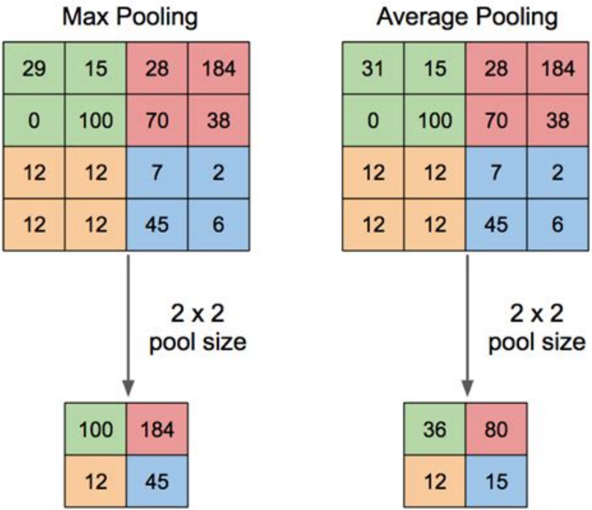
CNN Kernels in 3D

3D Convolution with a Kernel: Principle



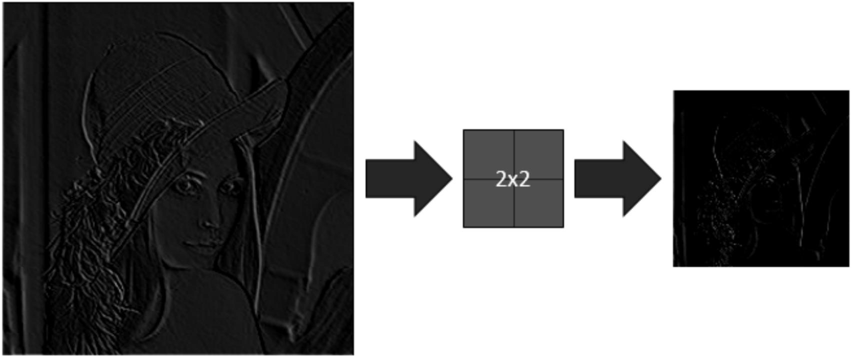
CNN Pooling

The Principle



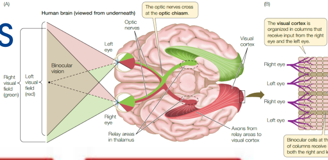
CNN Pooling

On real data (effectivly subsampling)

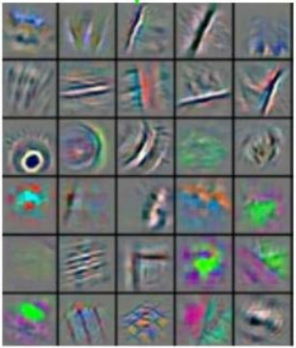


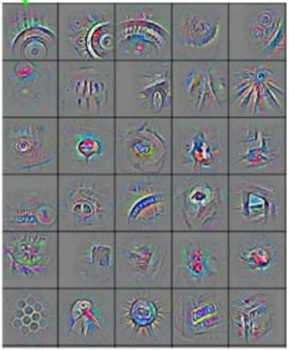
Convolutional Neural Networks

Low-, Mid-, and High-Level Feature Extraction



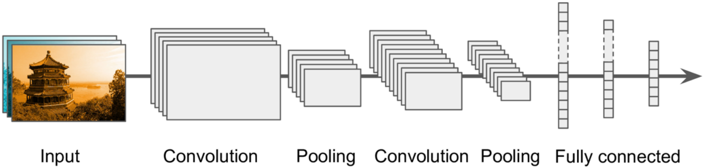
Low-Level Feature → Mid-Level Feature → High-Level Feature → Trainable Classifier →

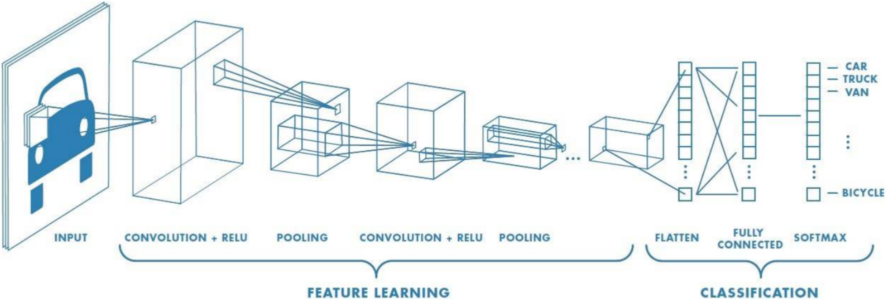




Convolutional Neural Networks

Stacking It All Up..





CNN'S IN PRACTICE

CNN's in Practice

Varroa mite detector—with a CNN somewhere in the pipeline

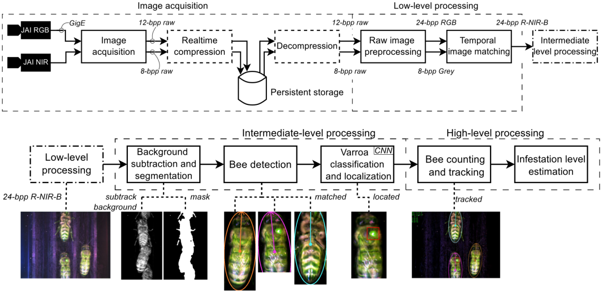

A computer vision system to monitor the infestation level of *Varroa destructor* in a honeybee colony

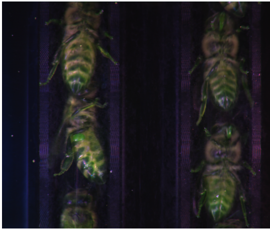
Kim Bjerge[a,*], Carsten Eie Frigaard[a], Peter Høgh Mikkelsen[a], Thomas Holm Nielsen[a], Michael Mishih[a], Per Kryger[b]

[a] School of Engineering, Aarhus University, Finlandsgade 22, 8200 Aarhus N, Denmark
[b] Department of Agroecology - Entomology and Plant Pathology, Aarhus University, Flakkebjerg, Forsøgsvej 1, 4200 Slagelse, Denmark

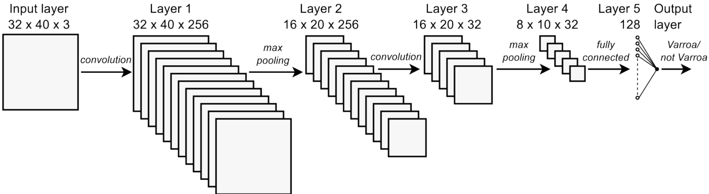
Abstract

Image processing pipeline





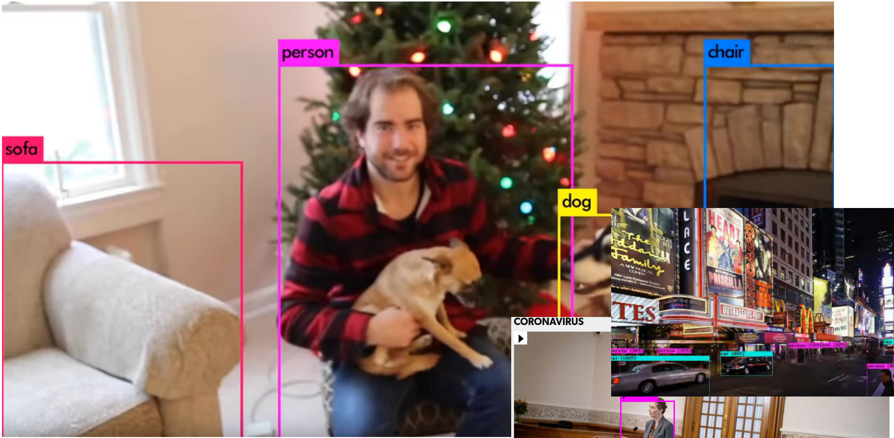
CNN architecture



CNN’s in Practice

YOLOv2+3+4 (You Only Look Once)

Real-time object detection, demo..



[https://homl.info/yolodemo]

[https://github.com/AlexeyAB/darknet]

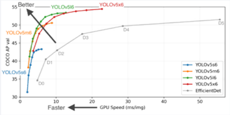
Intro til YOLOV5..

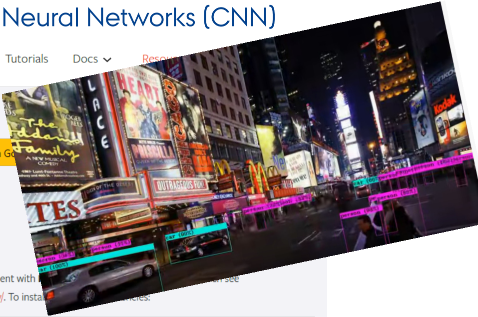
YOLOV3/4/5 Family and Convolutional Neural Networks (CNN)

PyTorch Get Started Ecosystem Mobile Blog Tutorials Docs Reso

YOLOV5

View on Github Open on Go





BEFORE YOU START

Start from a **Python>=3.8** environment with ... see https://pytorch.org/get-started/locally/. To instal ... cies:

```
pip install -qr https://raw.githubusercontent.com/ultralytics/yolov5/master/req
```

MODEL DESCRIPTION

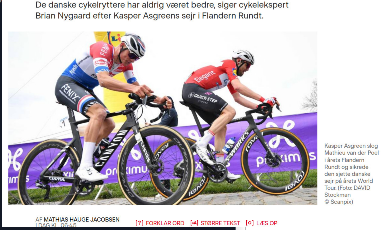
Nano	Small	Medium	Large	XLarge
YOLOv5n	YOLOv5s	YOLOv5m	YOLOv5l	YOLOv5x
4 MB$_{FP16}$	14 MB$_{FP16}$	41 MB$_{FP16}$	89 MB$_{FP16}$	166 MB$_{FP16}$
6.3 ms$_{V100}$	6.4 ms$_{V100}$	8.2 ms$_{V100}$	10.1 ms$_{V100}$	12.1 ms$_{V100}$
28.4 mAP$_{COCO}$	37.2 mAP$_{COCO}$	45.2 mAP$_{COCO}$	48.8 mAP$_{COCO}$	50.7 mAP$_{COCO}$

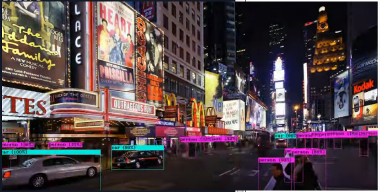
YOLOV5: [https://pytorch.org/hub/ultralytics_yolov5/]
Demo video: [https://www.youtube.com/watch?v=1_SiUOYUoOI]

CNN's in Practice

YOLOv2+3+4 (You Only Look Once)

```
Terminal
o obj/art.o obj/tag.o obj/cifar.o obj/go.o obj/rnn.o obj/segmenter.o obj/regressor.o obj/classifier.o obj/coco.o
 obj/yolo.o obj/detector.o obj/nightmare.o obj/instance-segmenter.o obj/darknet.o libdarknet.a -o darknet -lm -p
thread  -L/opt/opencv/opencv4/lib -lopencv_core -lopencv_imgproc -lopencv_imgcodecs -lopencv_videoio -lopencv_hi
ghgui  -ltiff -lstdc++ libdarknet.a
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   256 x 256 x   3   ->   256 x 256 x  32  0.113
    1 max          2 x 2 / 2   256 x 256 x  32   ->   128 x 128 x  32
    2 conv     64  3 x 3 / 1   128 x 128 x  32   ->   128 x 128 x  64  0.604
    3 max          2 x 2 / 2   128 x 128 x  64   ->    64 x  64 x  64
    4 conv    128  3 x 3 / 1    64 x  64 x  64   ->    64 x  64 x 128  0.604
    5 conv     64  1 x 1 / 1    64 x  64 x 128   ->    64 x  64 x  64  0.067
    6 conv    128  3 x 3 / 1    64 x  64 x  64   ->    64 x  64 x 128  0.604
    7 max          2 x 2 / 2    64 x  64 x 128   ->    32 x  32 x 128
    8 conv    256  3 x 3 / 1    32 x  32 x 128   ->    32 x  32 x 256  0.604
    9 conv    128  1 x 1 / 1    32 x  32 x 256   ->    32 x  32 x 128  0.067
   10 conv    256  3 x 3 / 1    32 x  32 x 128   ->    32 x  32 x 256  0.604
   11 max          2 x 2 / 2    32 x  32 x 256   ->    16 x  16 x 256
   12 conv    512  3 x 3 / 1    16 x  16 x 256   ->    16 x  16 x 512  0.604
   13 conv    256  1 x 1 / 1    16 x  16 x 512   ->    16 x  16 x 256  0.067
   14 conv    512  3 x 3 / 1    16 x  16 x 256   ->    16 x  16 x 512  0.604
   15 conv    256  1 x 1 / 1    16 x  16 x 512   ->    16 x  16 x 256  0.067 BFLOPs
   16 conv    512  3 x 3 / 1    16 x  16 x 256   ->    16 x  16 x 512  0.604 BFLOPs
   17 max          2 x 2 / 2    16 x  16 x 512   ->     8 x   8 x 512
   18 conv   1024  3 x 3 / 1     8 x   8 x 512   ->     8 x   8 x1024  0.604
   19 conv    512  1 x 1 / 1     8 x   8 x1024   ->     8 x   8 x 512  0.067
   20 conv   1024  3 x 3 / 1     8 x   8 x 512   ->     8 x   8 x1024  0.604
   21 conv    512  1 x 1 / 1     8 x   8 x1024   ->     8 x   8 x 512  0.067
   22 conv   1024  3 x 3 / 1     8 x   8 x 512   ->     8 x   8 x1024  0.604
   23 conv   1000  1 x 1 / 1     8 x   8 x1024   ->     8 x   8 x1000  0.131
   24 avg                        8 x   8 x1000   ->  1000
   25 softmax                                        1000
Loading weights from darknet19.weights...Done!
test.jpg: Predicted in 1.965434 seconds.
64.68%: mountain bike
16.00%: web site
12.87%: bicycle-built-for-two
 1.05%: crash helmet
 0.86%: alp
cef@leno:~/texitmal/darknet$
```





CNN's in Practice

The LeNET-5 Architecture

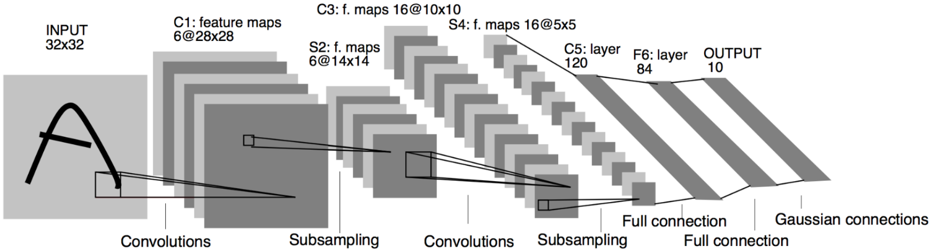


Table 14-1. LeNet-5 architecture

Layer	Type	Maps	Size	Kernel size	Stride	Activation
Out	Fully Connected	–	10	–	–	RBF
F6	Fully Connected	–	84	–	–	tanh
C5	Convolution	120	1×1	5×5	1	tanh
S4	Avg Pooling	16	5×5	2×2	2	tanh
C3	Convolution	16	10×10	5×5	1	tanh
S2	Avg Pooling	6	14×14	2×2	2	tanh
C1	Convolution	6	28×28	5×5	1	tanh
In	Input	1	32×32	–	–	–

Other famours CNN-architectures:

- AlexNet,
- GoogLeNet (inception),
- ResNet (152 layers, skip-connections),
- VGGNet,
- Inception-v4 (GoogLeNet + ResNet),
- ...

CNN's in Practice

A LeNET-5 'Like' Architecture

Table 14-1. LeNet-5 architecture

Layer	Type	Maps	Size	Kernel size	Stride	Activation
Out	Fully Connected	–	10	–	–	RBF
F6	Fully Connected	–	84	–	–	tanh
C5	Convolution	120	1×1	5×5	1	tanh
S4	Avg Pooling	16	5×5	2×2	2	tanh
C3	Convolution	16	10×10	5×5	1	tanh
S2	Avg Pooling	6	14×14	2×2	2	tanh
C1	Convolution	6	28×28	5×5	1	tanh
In	Input	1	32×32	–	–	–

In [9]:

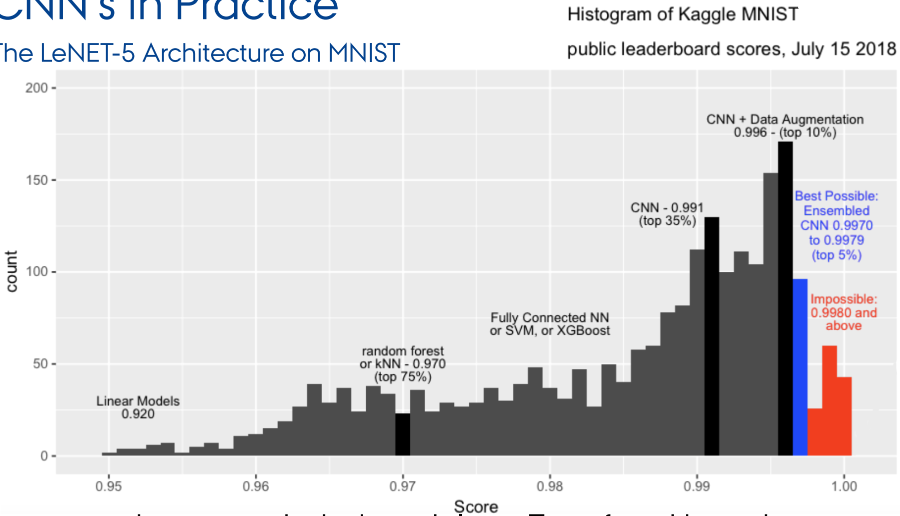
```python
import keras
from keras import layers

model = keras.Sequential()

model.add(layers.Conv2D(filters=6, kernel_size=(3, 3),
                        activation='relu', input_shape=(32,32,1)))
model.add(layers.AveragePooling2D())

model.add(layers.Conv2D(filters=16, kernel_size=(3, 3), activation='relu'))
model.add(layers.AveragePooling2D())

model.add(layers.Flatten())

model.add(layers.Dense(units=120, activation='relu'))
model.add(layers.Dense(units=84,  activation='relu'))
model.add(layers.Dense(units=10,  activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 30, 30, 6)         60
_________________________________________________________________
average_pooling2d_5 (Average (None, 15, 15, 6)         0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 13, 13, 16)        880
_________________________________________________________________
average_pooling2d_6 (Average (None, 6, 6, 16)          0
_________________________________________________________________
flatten_3 (Flatten)          (None, 576)               0
_________________________________________________________________
dense_7 (Dense)              (None, 120)               69240
_________________________________________________________________
dense_8 (Dense)              (None, 84)                10164
_________________________________________________________________
dense_9 (Dense)              (None, 10)                850
=================================================================
Total params: 81,194
Trainable params: 81,194
Non-trainable params: 0
```

[https://colab.research.google.com/drive/1CVm5OPGE4vhtB5I_a_yc4h5F-itKOVL9]

CNN's in Practice

The LeNET-5 Architecture on MNIST



- using pre-trainded models => Transfered Learning,
- object detection,
- semantic segmentation,
- time series => RNN's, ...

[https://www.kaggle.com/datasets/numbersareuseful/public-leaderboard-mnist]