Deep Learning

An Introduction

Compiled by

Dr. Sandeep Palakkal

Samsung R&D Institute India, Bangalore

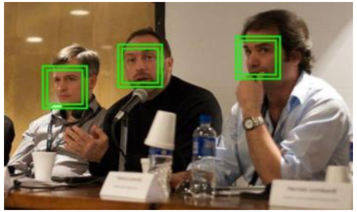
Dec 14, 2018

Machine Learning

“Machine learning gives computers the ability to learn without being explicitly programmed.” -- Arthur Samuel, 1959.

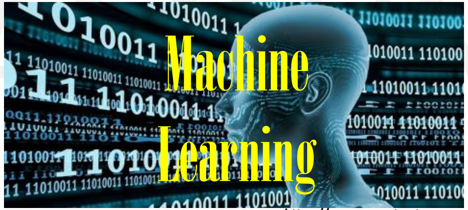
https://en.wikipedia.org

http://www.image-net.org



https://en.wikipedia.org

https://www.trademed.com



https://www.expertsystem.com

https://www.techleer.com

https://services.harman.com/

https://www.amazon.in

https://towardsdatascience.com

https://www.samsung.com

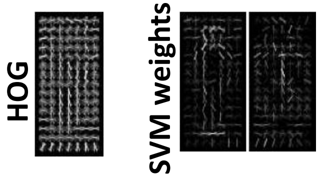
Machine Learning : The Conventional Approach

Conventional ML

Data → Preprocessing → Feature Extraction (Hand-engineered Features, e.g., SIFT, SURF, HOG, Wavelets, etc.) → Feature Selection → ML Algorithm: Prediction, Classification, Recognition, etc.

Example Human Detection using HOG [1]

Input image → **Preprocess:** Normalize gamma & colour → **Feature Extraction:** Compute gradients → Weighted vote into spatial & orientation cells → Contrast normalize over overlapping spatial blocks → Collect HOG's over detection window → **Classifier:** Linear SVM → Person / non–person classification

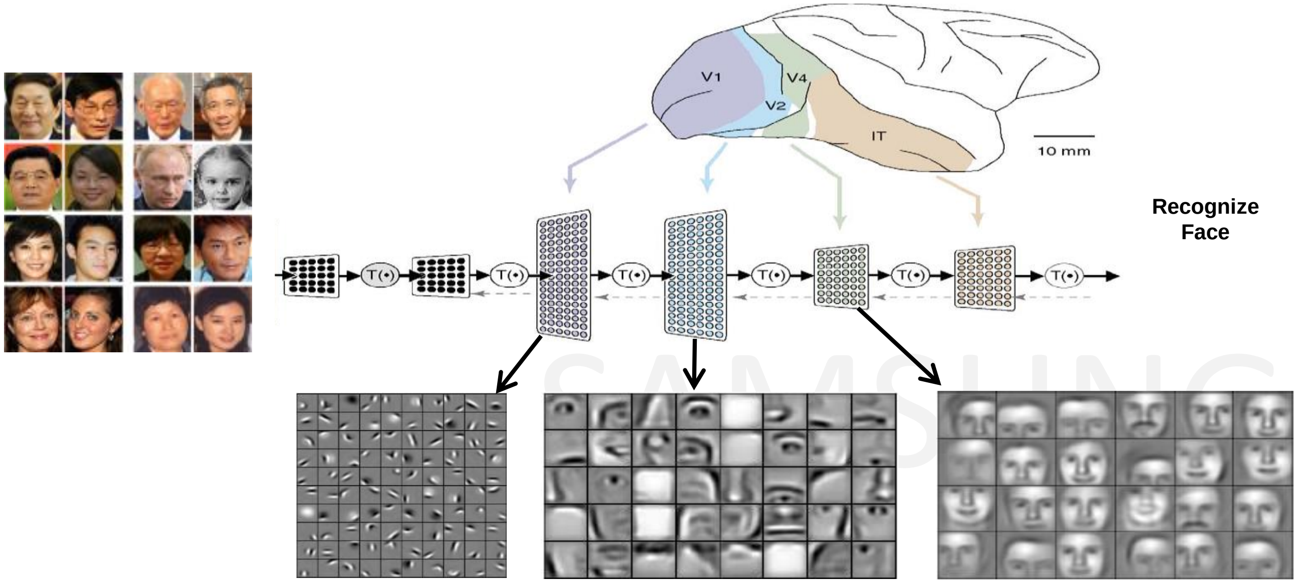


- ❖ Designing SIFT took 10 years of effort by Prof. David G. Lowe [2]
- ❖ Can we do better?

[1] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *CVPR* 2005.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV* 2004

Deep Learning : What?



Several Areas of Influence

- ✓ Affective Computing
- ✓ Artificial Intelligence
- ✓ Autonomous Cars
- ✓ Biology
- ✓ Business
- ✓ Computer Vision
- ✓ Finance
- ✓ Gaming
- ✓ Genomics
- ✓ Healthcare/Health Informatics
- ✓ Information Retrieval
- ✓ Natural Language Processing
- ✓ Sparse Coding
- ✓ Speech Processing
- ✓ Trading
- ✓ Weather Forecast

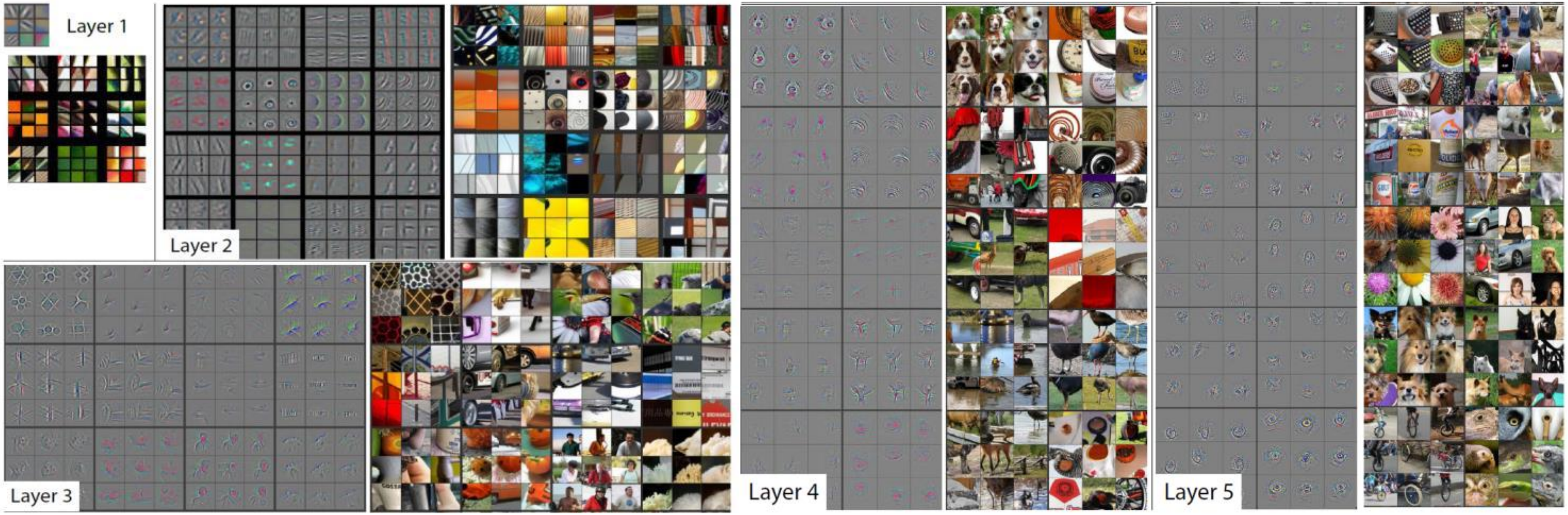
- Deep learning is a **machine learning** approach that makes use of **deep neural networks (DNN)**
- A DNN consists of a hierarchy of computational layers (usually, more than 2 layers)
- Each layer in DNN transforms the input data into slightly higher and more abstract representation
- The idea of hierarchical representation and DNN architectures are inspired by Visual Cortex of brain

[1] Y. LeCun, Y. Bengio and G. Hinton, Deep Learning, *Nature* 2015

Features Learned by AlexNet for Image Classification

SAMSUNG

- DNN learns features in multiple layers of abstraction
- Level of abstraction increases as we go deeper
- Deeper layers capture more complex structures pertinent to data

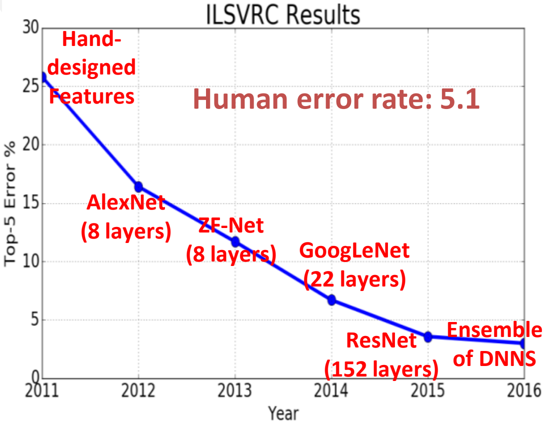


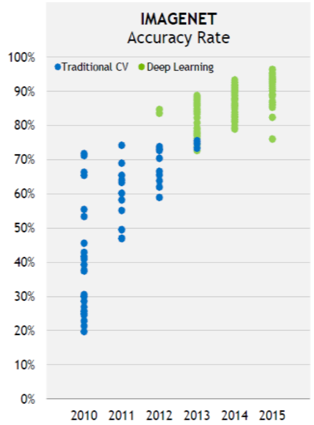
[1] M. D. Zeiler and R. Fergus. "Visualizing and understanding convolutional networks," *ECCV*, 2014.

Deep V/S Shallow Learning

Courtesy: http://cs231n.github.io/classification/

- ❑ Features: **Selectivity vs invariance**
  - *Selective* to relevant variations (e.g., between class)
  - *Invariant* to irrelevant variations (e.g., within-class)
- ❑ Deep networks: complex, hierarchical learning
  - Stack of non-linear input-output mapping
  - Each stack transforms input to increase selectivity & invariance
- ❑ Shallow networks would be heavier to get same performance



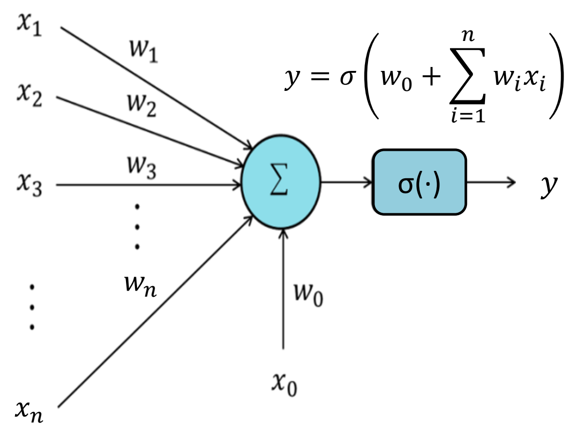


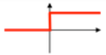
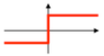
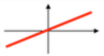
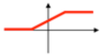
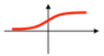
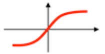
Credits: NVIDIA CES 2016 Press Conference

Artificial Neural Networks (ANN)

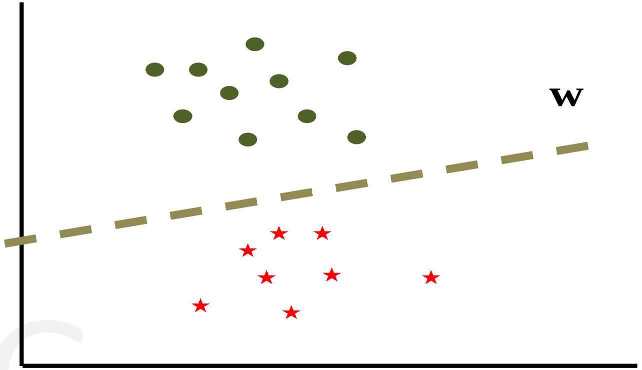
Single Perceptron (Rosenblatt-1958)

$$y = \sigma\left(w_0 + \sum_{i=1}^{n} w_i x_i\right)$$

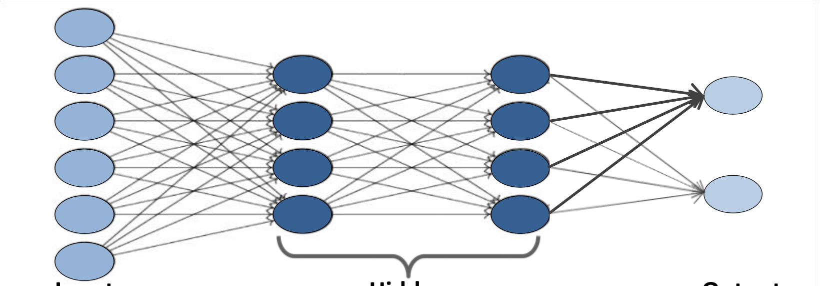


Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$	Multi-layer NN	

Logistic Regression/Classification by Perceptron



The Neural Network: Multilayer Perceptrons (MLP)



A 2-Layer MLP (Single Hidden Layer) is a universal approximator!!!

Classification Boundary of a 2-Layer MLP

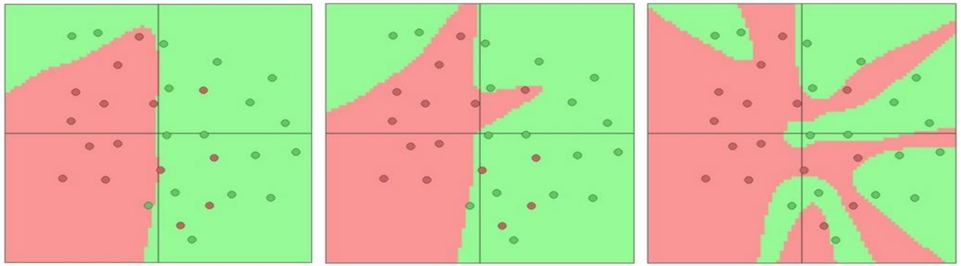
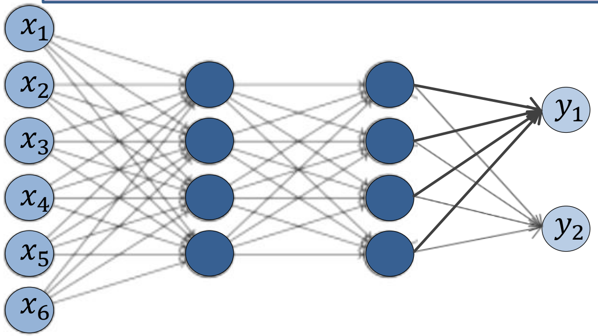
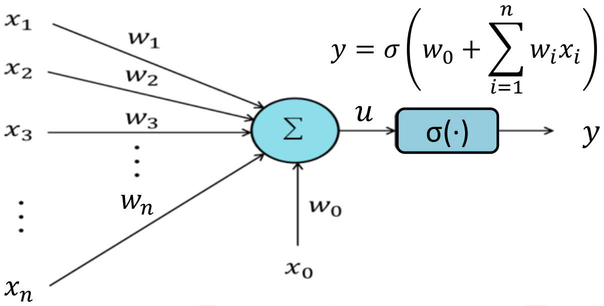


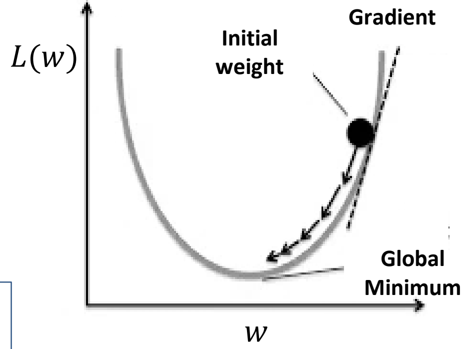
Image Credit: http://cs231n.github.io/neural-networks-1/

Training Neural Networks: Gradient Descend & Backpropagation

Objective of training: Find weights **w** that reduces an error/loss between output and desired output







Loss Function

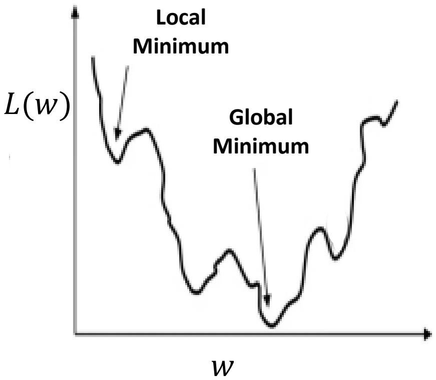
- ❑ Desired output: $\mathbf{y} = \{y_1, y_2, \ldots, y_m\}$
- ❑ Output of network: $\hat{\mathbf{y}} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m\}$
- ❑ Loss functions:
  - Euclidean: $L = \frac{1}{2m}\sum_{j=1}^{m}(\hat{y}_j - y_j)^2$
- ❑ Other loss functions:
  - Entropy loss function, L1 Loss function, etc.

Gradient Descent & backpropagation

- Iterative algorithm
  - k-th iteration: $w^k = w^{k-1} - \eta \frac{\partial L}{\partial w}$
- Apply chain rule to compute gradient:

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial y_j}\frac{\partial y_j}{\partial u_j}\frac{\partial u_j}{\partial w_i}$$

- For hidden layers, apply chain rule backwards
- Gradient of later layers propagates backward [1]



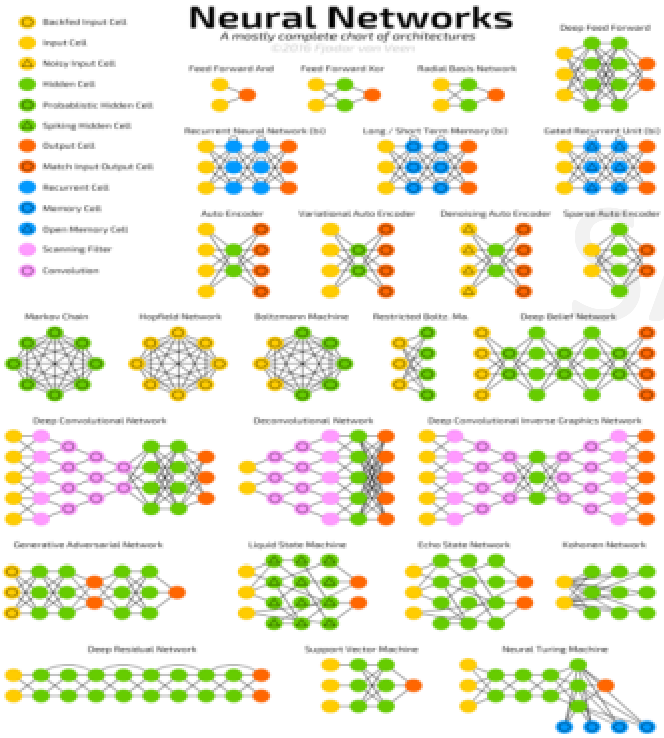
[1] Tom M. Mitchell, Machine Learning, McGraw Hill, 1997

[2] A. Karpathy, Stanford University CS231N Course Notes, http://cs231n.github.io/

Types of DNNs

Several types of (deep) neural networks

- Convolutional neural networks (CNN)
- Auto-encoders
- Recurrent neural networks (RNN)
- Long short-term memory (LSTM)
- Deep belief networks (DBN)
- (Restricted) Deep Boltzmann networks (DBM & RBM)



http://www.asimovinstitute.org/neural-network-zoo/

Convolutional Neural Networks (CNN)

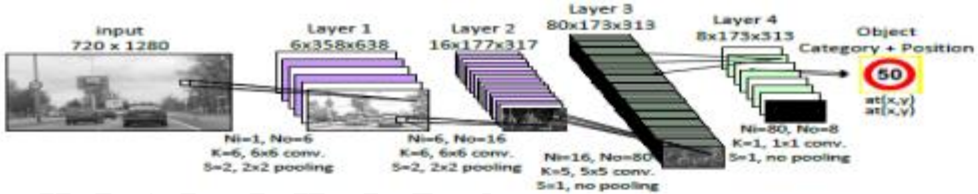
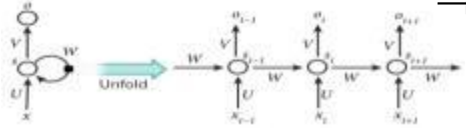


Image credit: M. Peemen et al., VLIW Code Generation for a Convolutional Network Accelerator, SCOPES, 2015

RNN & LSTM



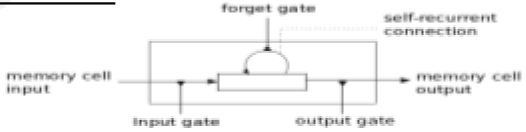


Image credits: Y. LeCun, Y. Bengio and G. Hinton, Deep Learning, *Nature* 2015, http://deeplearning.net/tutorial/lstm.html

Autoencoder

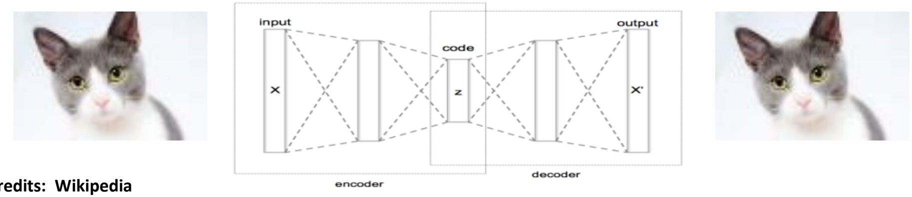
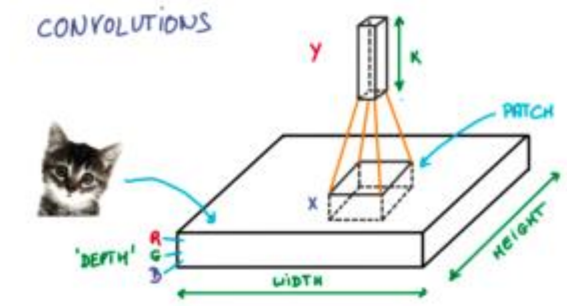


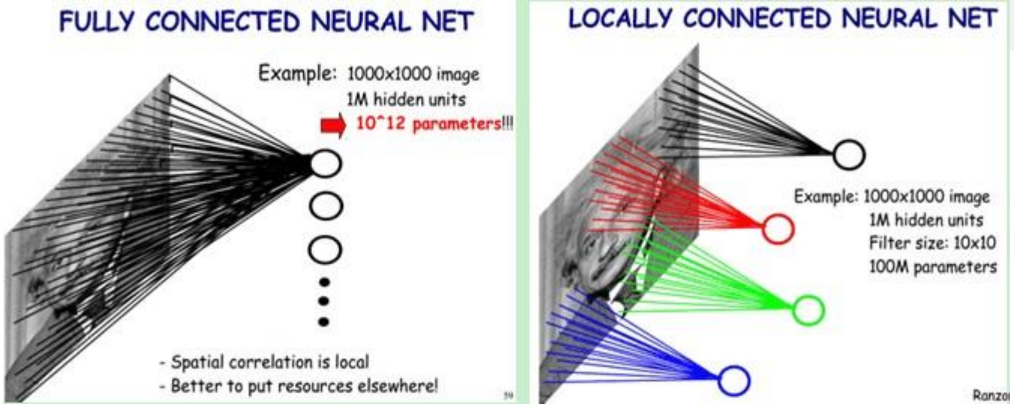
Image credits: Wikipedia

Convolutional Neural Networks (CNN)

- 3 Key ideas :
  - Local receptive field
  - Shared weights
  - Spatial/temporal subsampling (pooling)



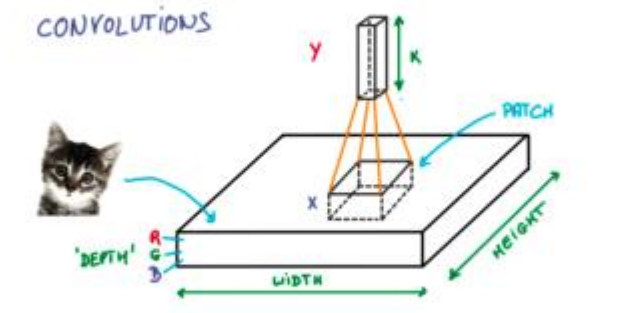
Convolution in 3D (Image courtesy: https://ireneli.eu/)



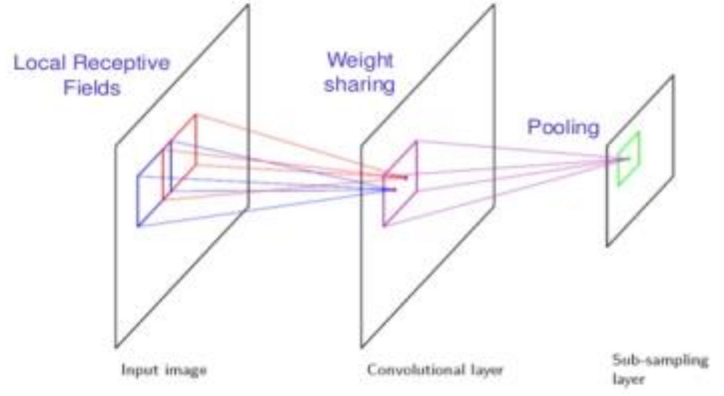
Source: https://ireneli.eu/

Convolutional Neural Networks (CNN)

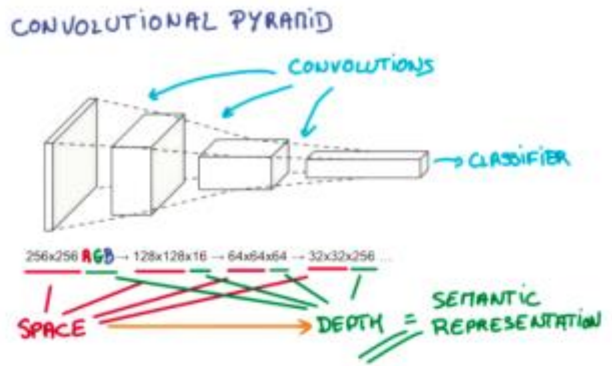
- 3 Key ideas :
  - Local receptive field
  - Shared weights
  - Spatial/temporal subsampling (pooling)



Convolution in 3D **(Image courtesy: https://ireneli.eu/)**



3 Key ideas in a Convolution Layer **(Image Courtesy: Cloudy Ngyuen)**



CNN Architecture **(Image courtesy: https://ireneli.eu/)**

Source: https://ireneli.eu/

A Primer on Convolution

Input image

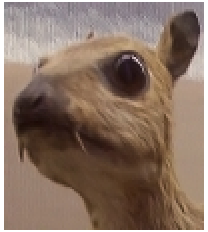
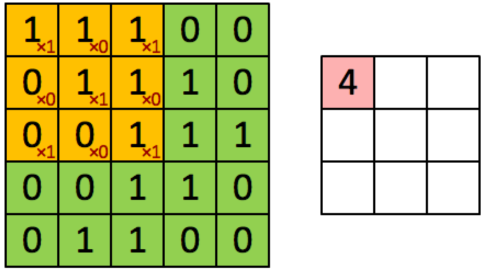


Image credit: Wikipedia

Convolution Kernel

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

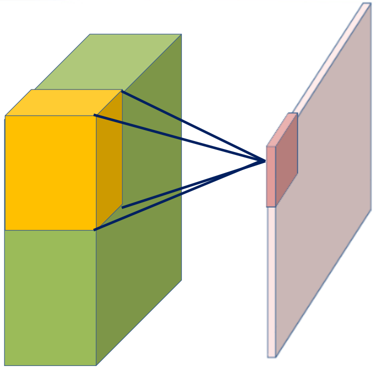
Feature map



Image

Convolved Feature

Source: http://deeplearning.stanford.edu/



Convolution in 3D

Source: B. P. Lathi, Linear Systems and Signals, 2nd Ed., 2004

- The most fundamental operation in signal/image processing
- For an MxN image

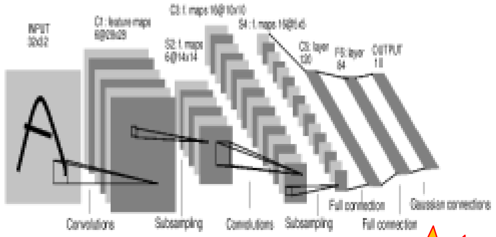
$$y(m,n)=\sum_{p=0}^{M-1}\sum_{q=0}^{N-1}x(p,q)\,h(m-p,n-q)$$

- Measures “similarity” between image and convolutional filter
- In CNN (3D):

$$y(m,n,k)=\sum_{c=0}^{C-1}\sum_{p=0}^{M-1}\sum_{q=0}^{N-1}x(p,q,c)\,h(m-p,n-q,c)$$

T. Dettmers, Understanding Convolution in Deep Learning, http://timdettmers.com/2015/03/26/convolution-deep-learning/

Evolution of CNN : Summary



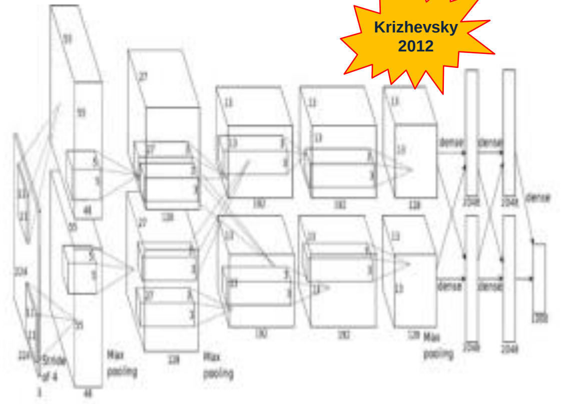
Sparse connections : S2 to C3

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

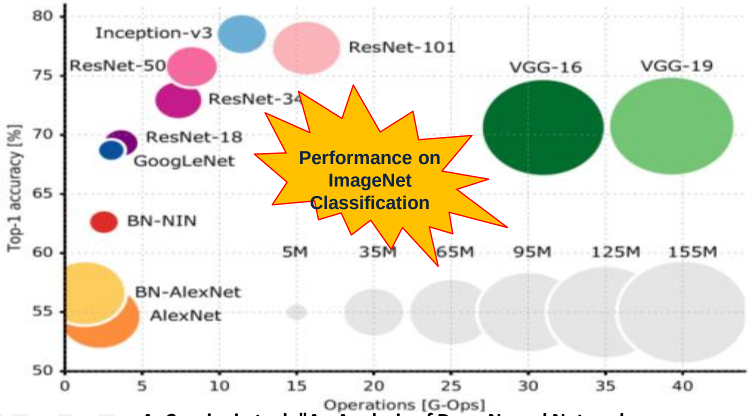
Le-Cun 1998

Le-Net

Krizhevsky 2012

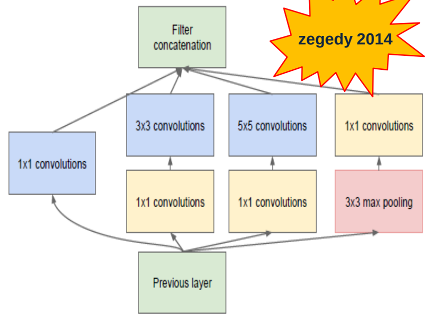


AlexNet Architecture

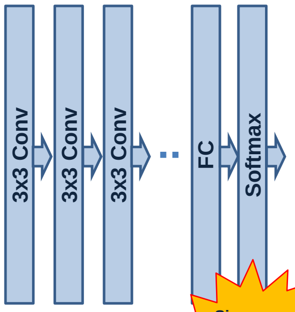


A. Canziani et. al, "An Analysis of Deep Neural Network Models for Practical Applications." *arXiv preprint* 2016

zegedy 2014



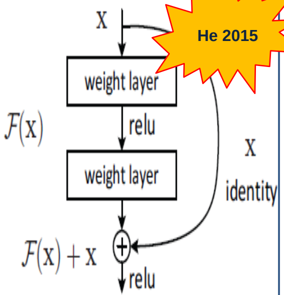
Inception Module (GoogLeNet)



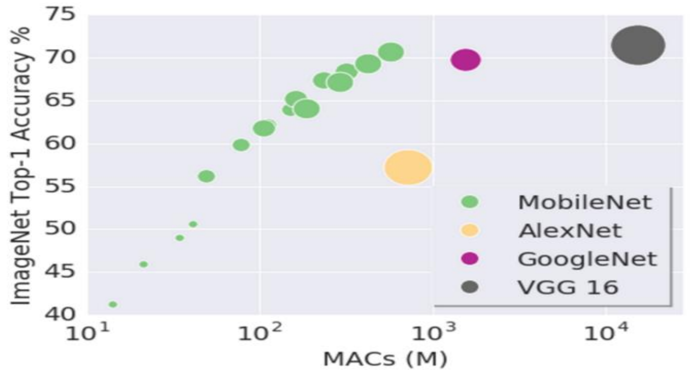
Simonyan 2014

VGG Net

He 2015



Residual learning (ResNet)

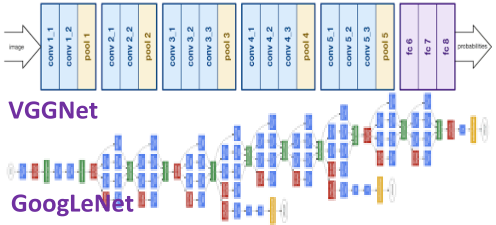


Deep Learning for Autonomous Cars

Autonomous Cars Use Case Scenario Examples

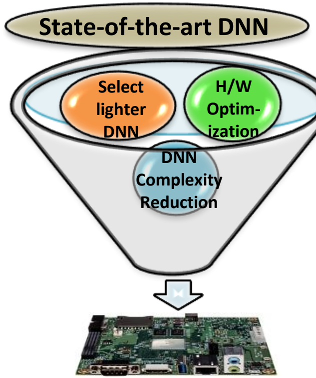
- Pedestrian/cyclist detection
- Free space and lane detection
- Vehicle detection
- Driver monitoring
- Traffic sign board recognition
- Differentiation of vehicle types
- HD mapping of road, etc.

Deep Learning Challenges in Autonomous Cars



- Highly accurate ML models
- State-of-the-art performance in ADAS

- DNNs are large with very high computational complexity requirements
- Large memory requirement on embedded systems
- Multiple optimizations required for deploying on embedded system



Algorithm Optimizations

Compatible Hardware Selection