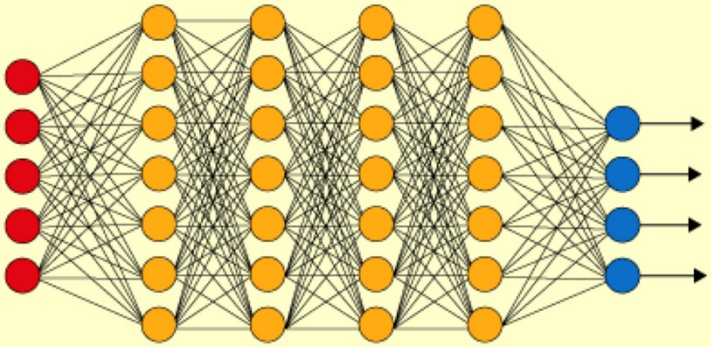


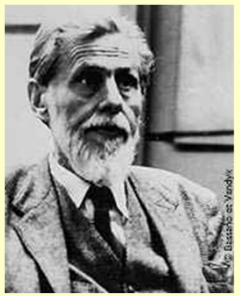
Convolutional Neural Network & Deep Learning

Ali Ridho Barakbah

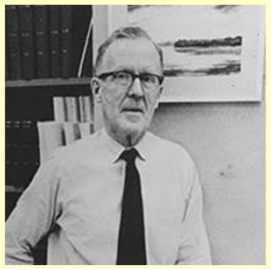
Knowledge Engineering Laboratory
Department of Information and Computer Engineering
Politeknik Elektronika Negeri Surabaya

History of Neural Networks

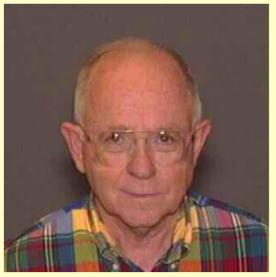
- 1943, Warren S. McCulloch and Walter Pitts → Input aggregation for computational power, threshold for firing characteristic
- 1949, Donald Hebb → Learning rule
- 1958, Frank Rosenblatt → Perceptrons
- 1960, Bernard Widrow and Ted Hoff → Adaptive Linear Neuron (Adaline), an early single-layer artificial neural network and the name of the physical device that implemented this network
- 1969, Marvin Minsky and Seymour Papert → mathematical proofs regarding perceptrons

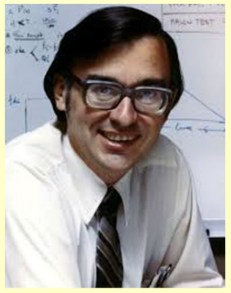

Warren S. McCulloch

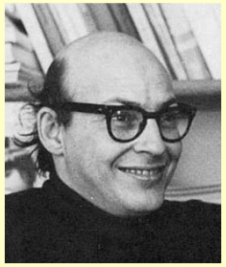

Walter Pitts

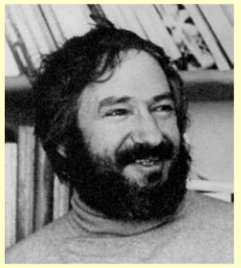

Donald Hebb


Frank Rosenblatt


Bernard Widrow

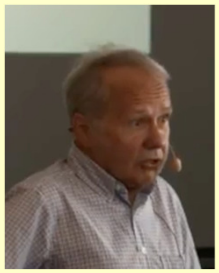

Ted Hoff


Marvin Minsky

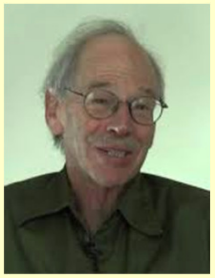

Seymour Papert

History of Neural Networks

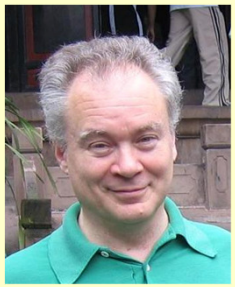
- 1970, Seppo Linnainmaa → automatic differentiation (AD) of discrete connected networks of nested differentiable functions, corresponding to Backpropagation
- 1973, Stuart Dreyfus → used backpropagation to adapt parameters of controllers in proportion to error gradients
- 1974, Paul Werbos → applying backpropagation parameters to artificial neural networks
- 1980, Kunihiko Fukushima → Neocognition, a first model of convolutional neural network
- 1982, Paul Werbos → applied Linnainmaa's AD method to neural networks in the way that is used today
- 1986, David Rumelhart, Geoffrey Hinton and Ronald Williams → showed experimentally that Linnainmaa's AD method can generate useful internal representations of incoming data in hidden layers of neural networks



Seppo Linnainmaa



Stuart Dreyfus



Paul Werbos

David Rumelhart

Kunihiko Fukushima

Geoffrey Hinton

Ronald William

History of Neural Networks

- 1997, Sepp Hochreiter and Jurgen Schmidhuber → Long-Short Term Memory (LSTM)
- 1998, Yann LeCun, Leon Bottou, Yoshua Bengio, Patrick Haffner → deep Convolutional Neural Network for digits recognition
- 2006, Geoffrey Hinton, Simon Osindero and Yee-Whye Teh → fast learning algorithm for deep belief nets
- 2013, Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton → Alexnet, re-implemented a convolutional neural network and trained on Imagenet using deep convolutional neural networks with 2 GPUs
- 2015, Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun → Deep Residual Learning for Image Recognition
- Today → GO, Deepmind


Sepp Hochreiter

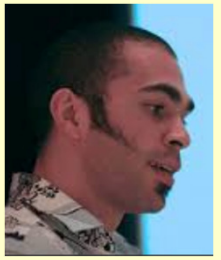

Jurgen Schmidhuber


Yann LeCun


Leon Bottou


Yoshua Bengio


Patrick Haffner

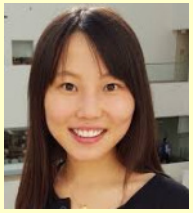

Simon Osindero


Yee-Whye Teh


Alex Krizhevsky


Ilya Sutskever

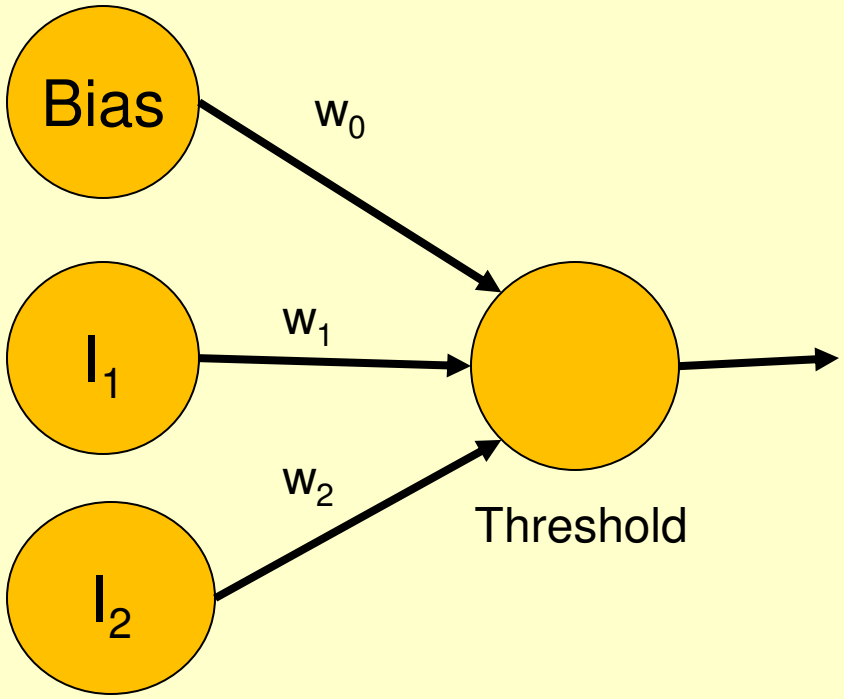

Kaiming He


Xiangyu Zhang

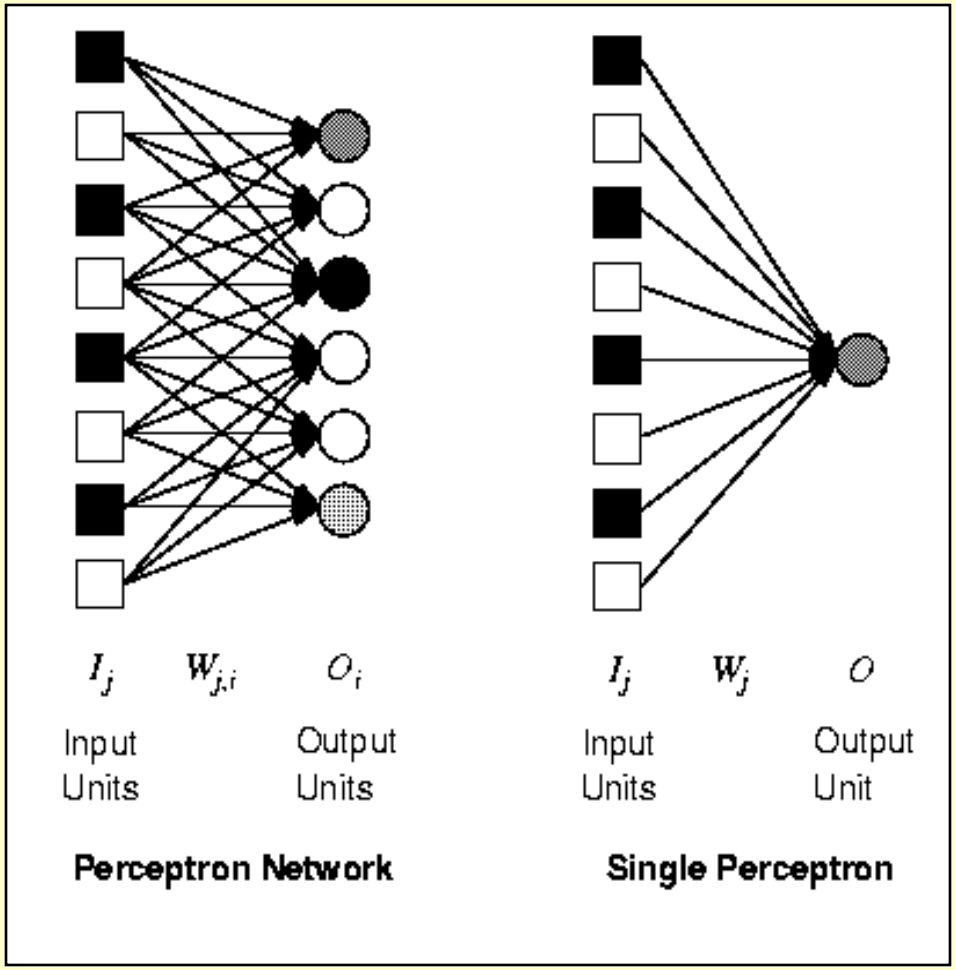

Shaoqing Ren


Jian Sun

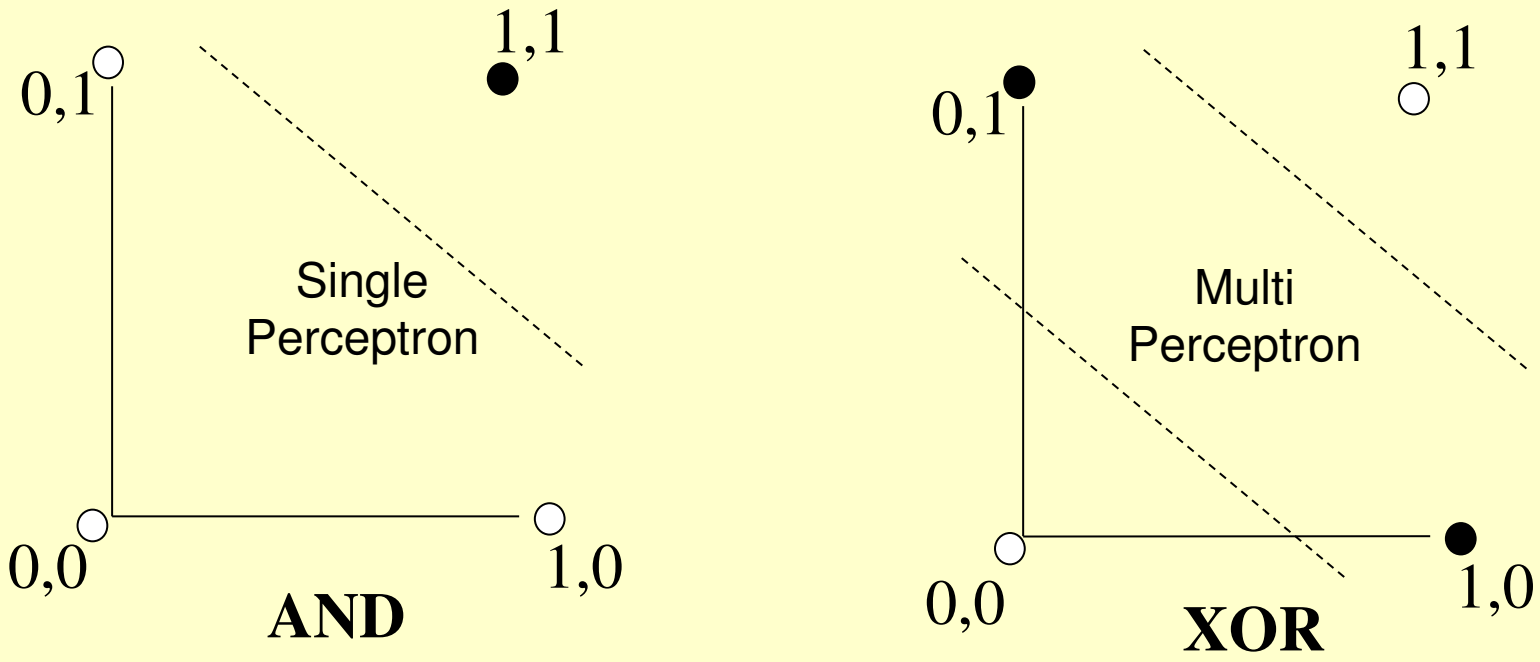
Simple Neural Network



Perceptron

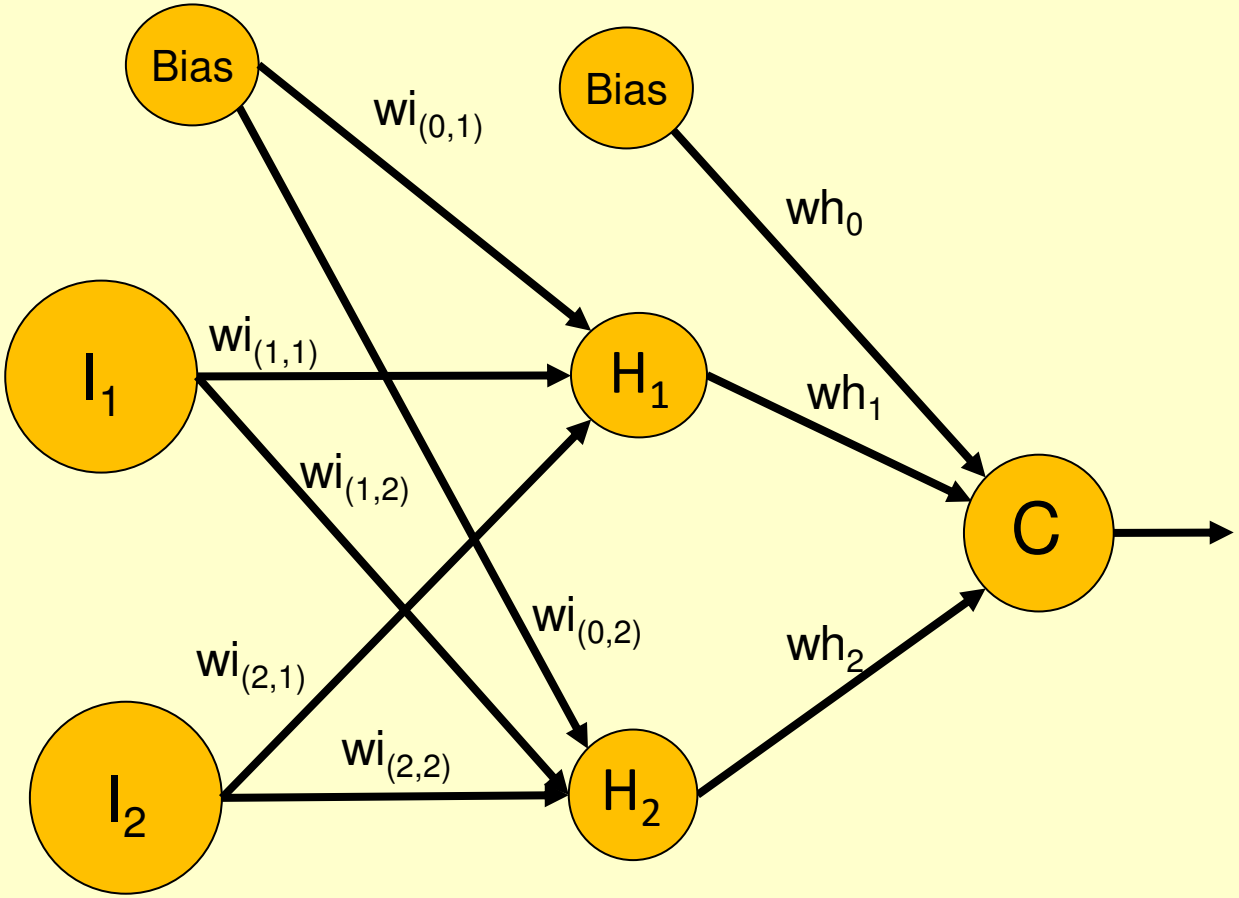

I_j
$W_{j,i}$
O_i
Input Units
Output Units
Perceptron Network
I_j
W_j
O
Input Units
Output Unit
Single Perceptron

What can perceptrons represent?

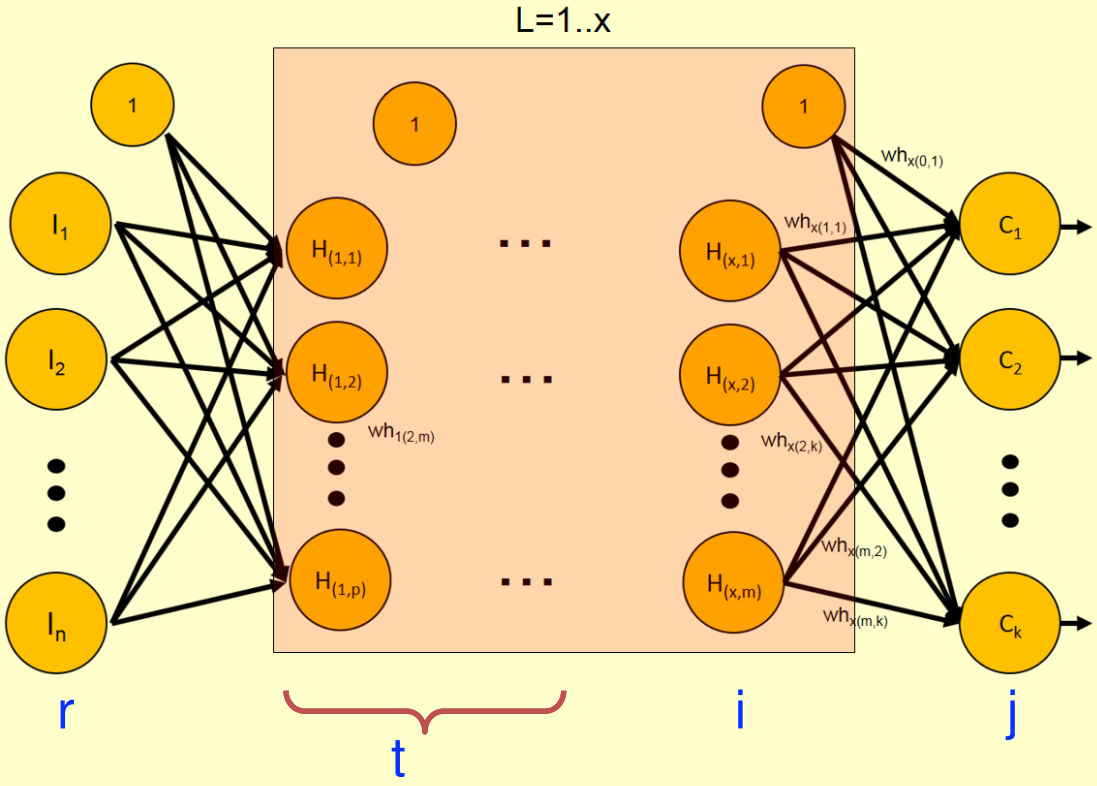


Linearly Separable

Multi Perceptron



Weight Update



i=1..m j=1..k t=1..p

$dC_j = C_j * (1-C_j) * (Target-C_j)$

$dH_{(2,i)} = H_{(2,i)} * (1-H_{(2,i)}) * \sum (wh_{2(i,j)} * dC_j)$

$dH_{(1,t)} = H_{(1,t)} * (1-H_{(1,t)}) * \sum (wh_{1(t,i)} * dH_{(t,i)})$

i=0..m j=1..k

$dwh_{x(i,j)} = \mu * H_{(x,i)} * dC_j \rightarrow H_{(x,0)}=1$

$wh_{x(i,j)} = wh_{x(i,j)} + dwh_{x(i,j)}$

x=1..x-1 t=1..p i=1..m

$dwh_{x(t,i)} = \mu * H_{(x,t)} * dH_{(x+1,i)} \rightarrow H_{(x,0)}=1$

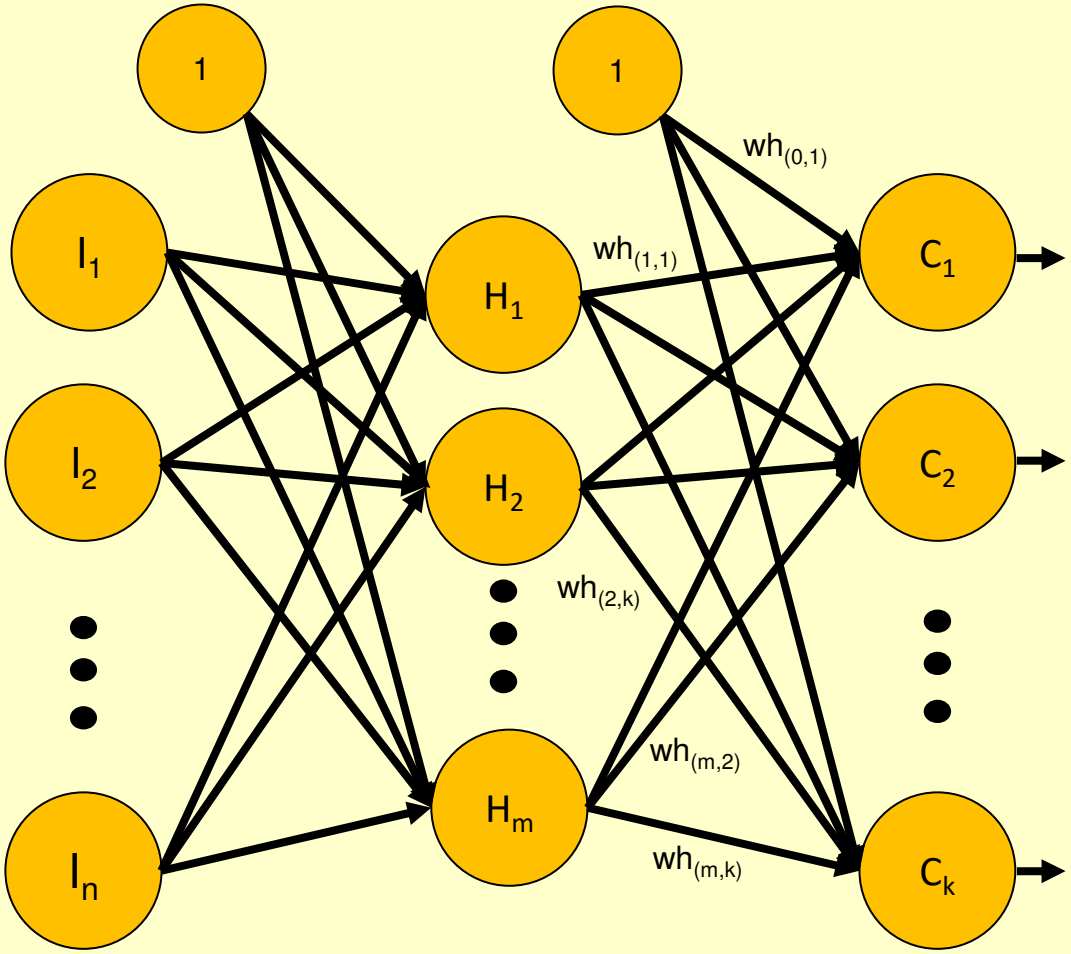
$wh_{x(t,i)} = wh_{x(t,i)} + dwh_{x(t,i)}$

r=0..n t=1..p

$dwi_{(r,t)} = \mu * I_r * dH_{(1,t)} \rightarrow I_0=1$

$wi_{(r,t)} = wi_{(r,t)} + dwi_{(r,t)}$

Learning Performance Analysis



$$(Target_1-C_1)^2 + (Target_2-C_2)^2 + \dots + (Target_k-C_k)^2$$

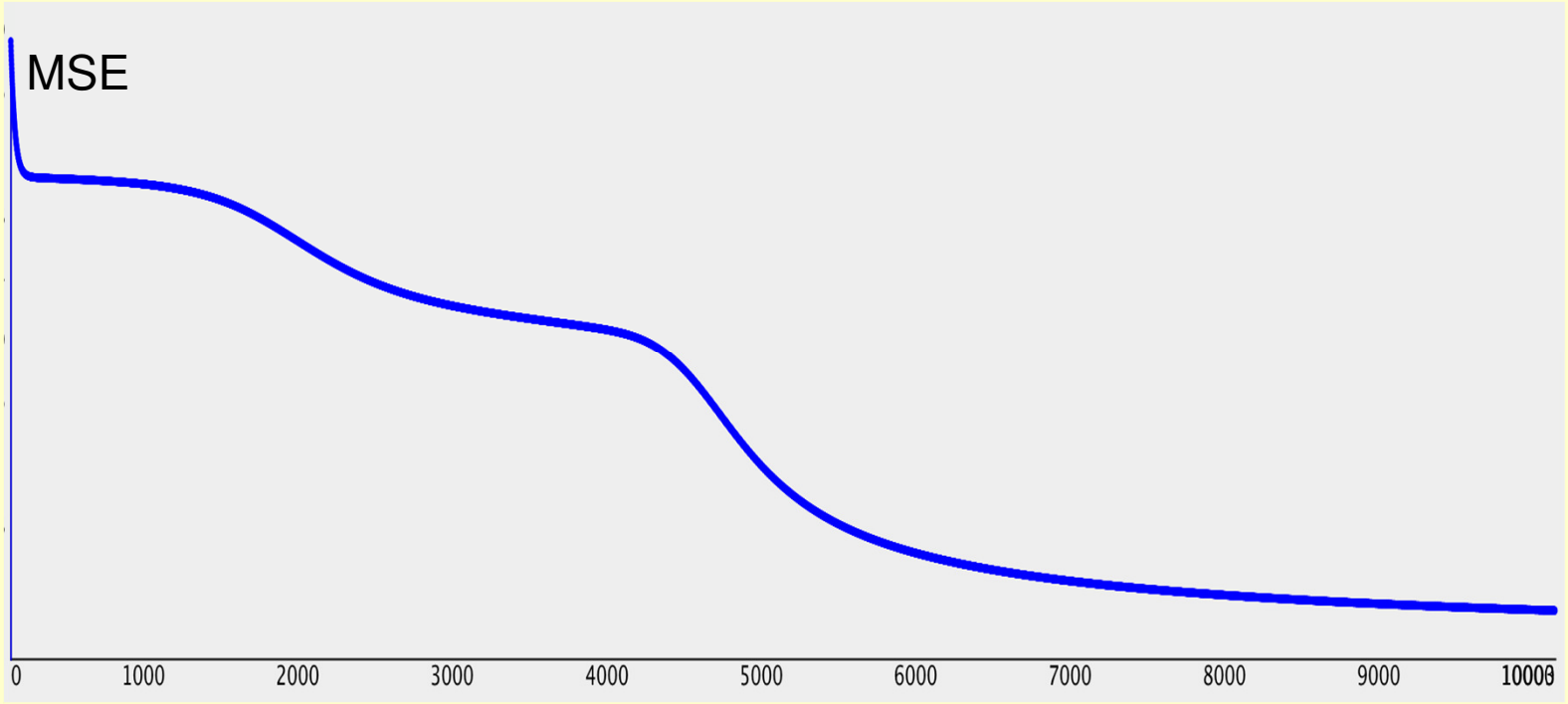
$$SSE=\sum_{d=1}^{f} SE_d$$

f ← total input sequence

$$MSE=\frac{SSE}{f}$$

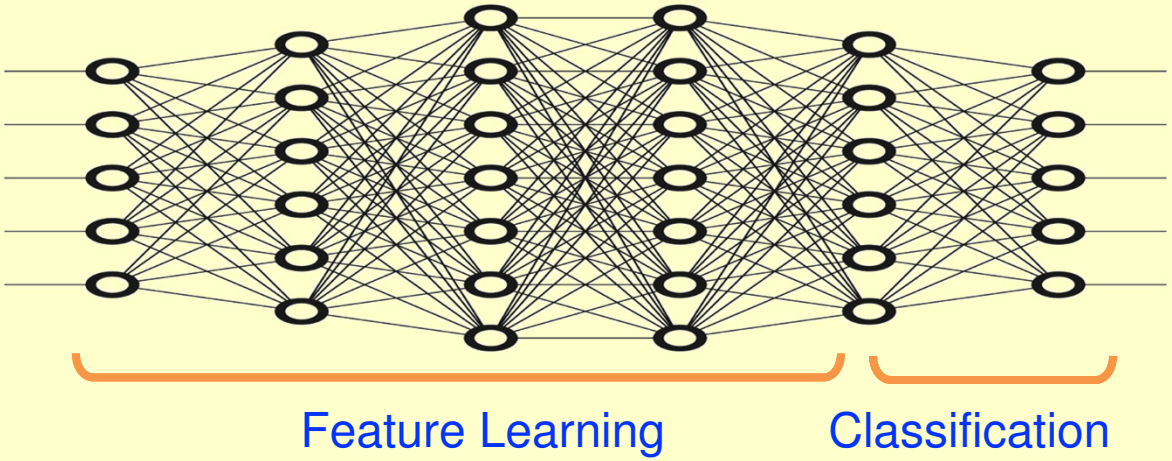
SE_d → Squared Error pada input sequence d

Gradient Descent

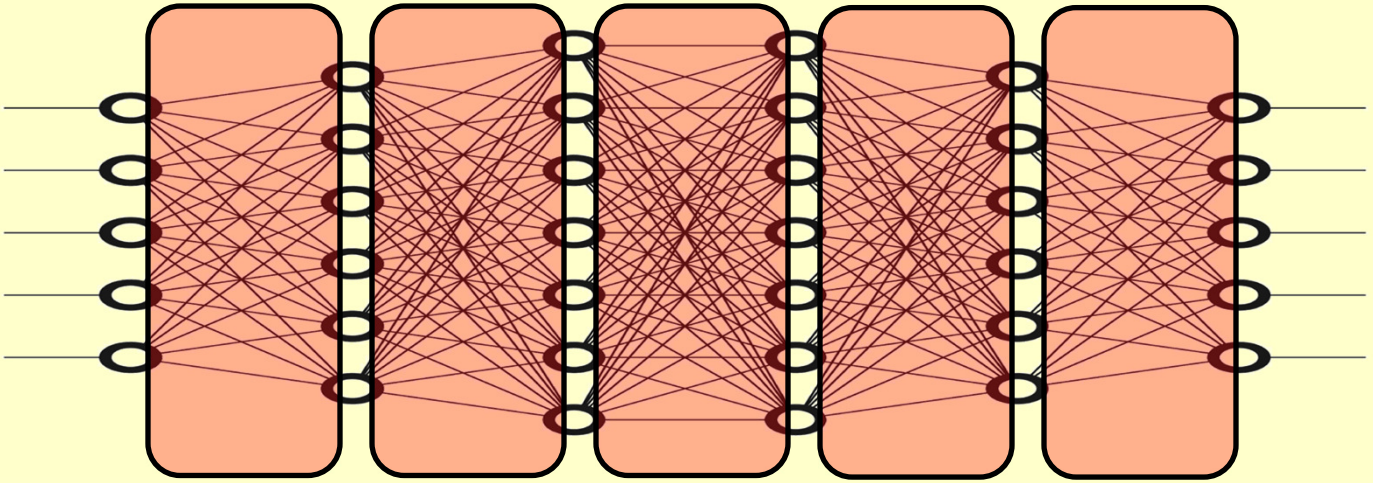


Deep Learning?

- A neural network with several layers
- Consists of two processes:
  - Feature Learning
  - Classification
- Learning for each layer of feature identification with auto-encoders



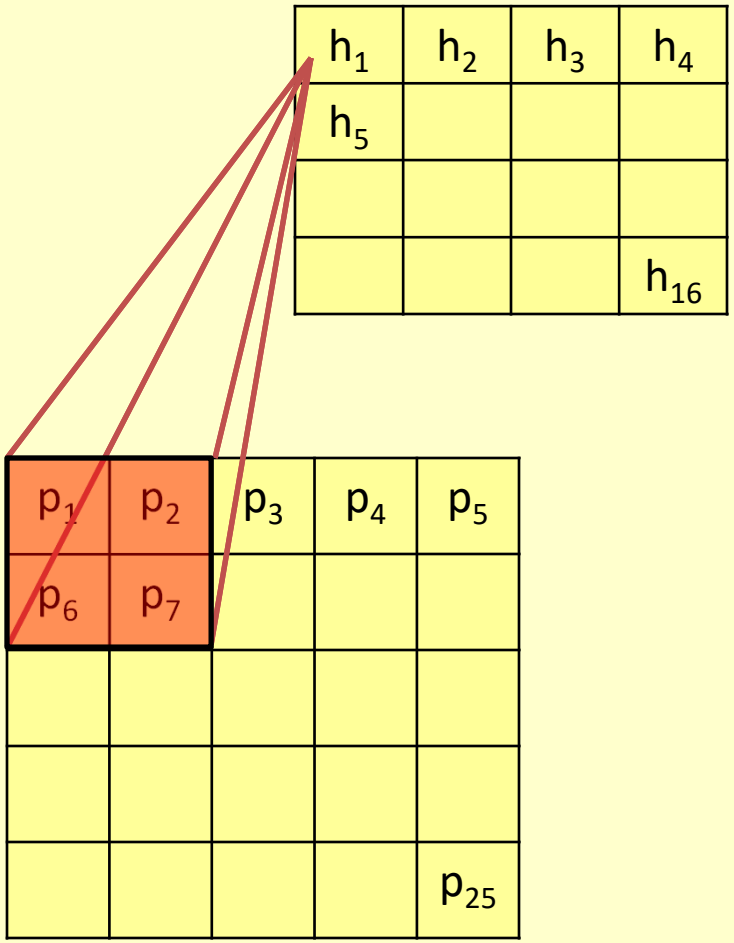
Learning Process?

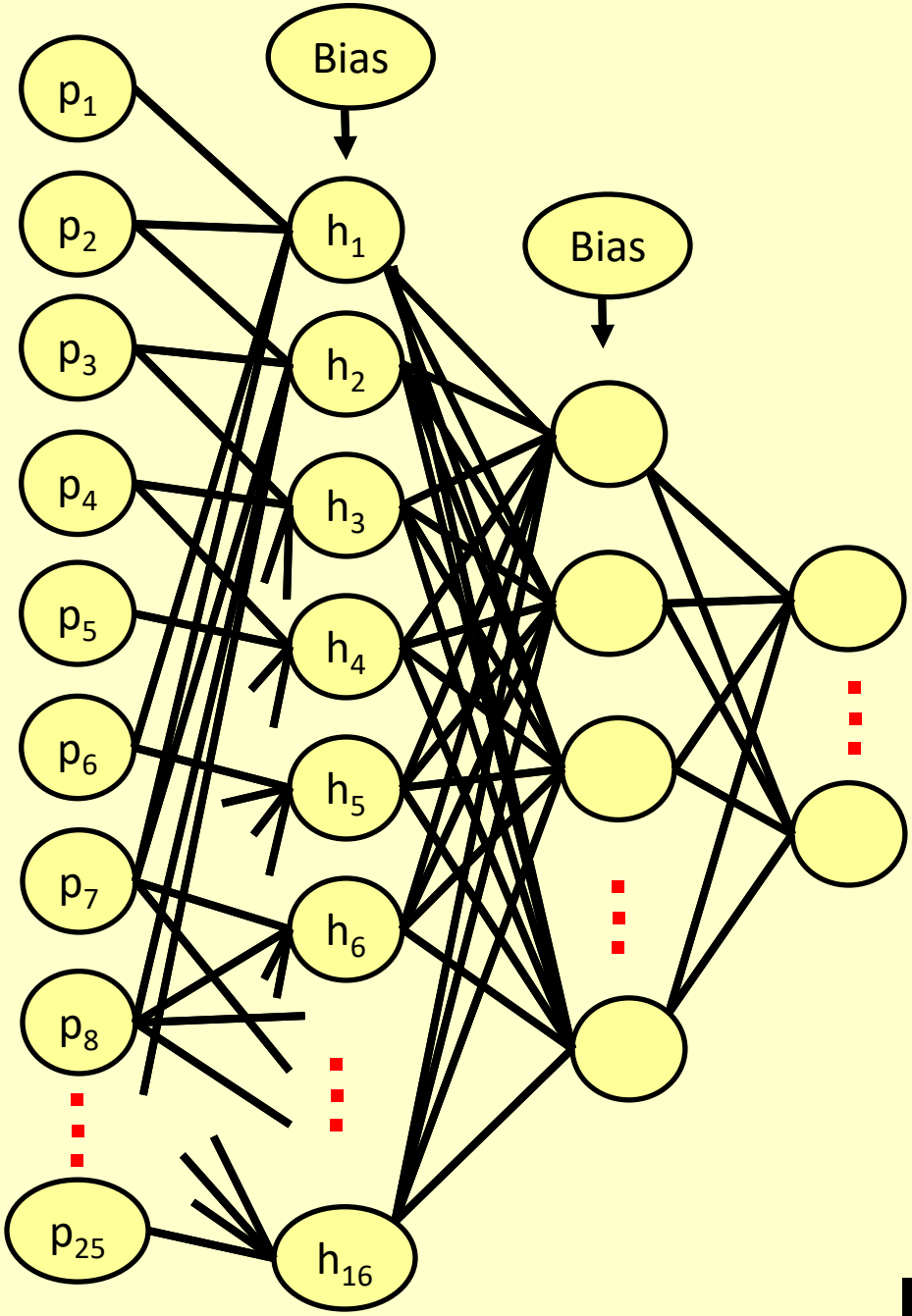


Convolutional Neural Network

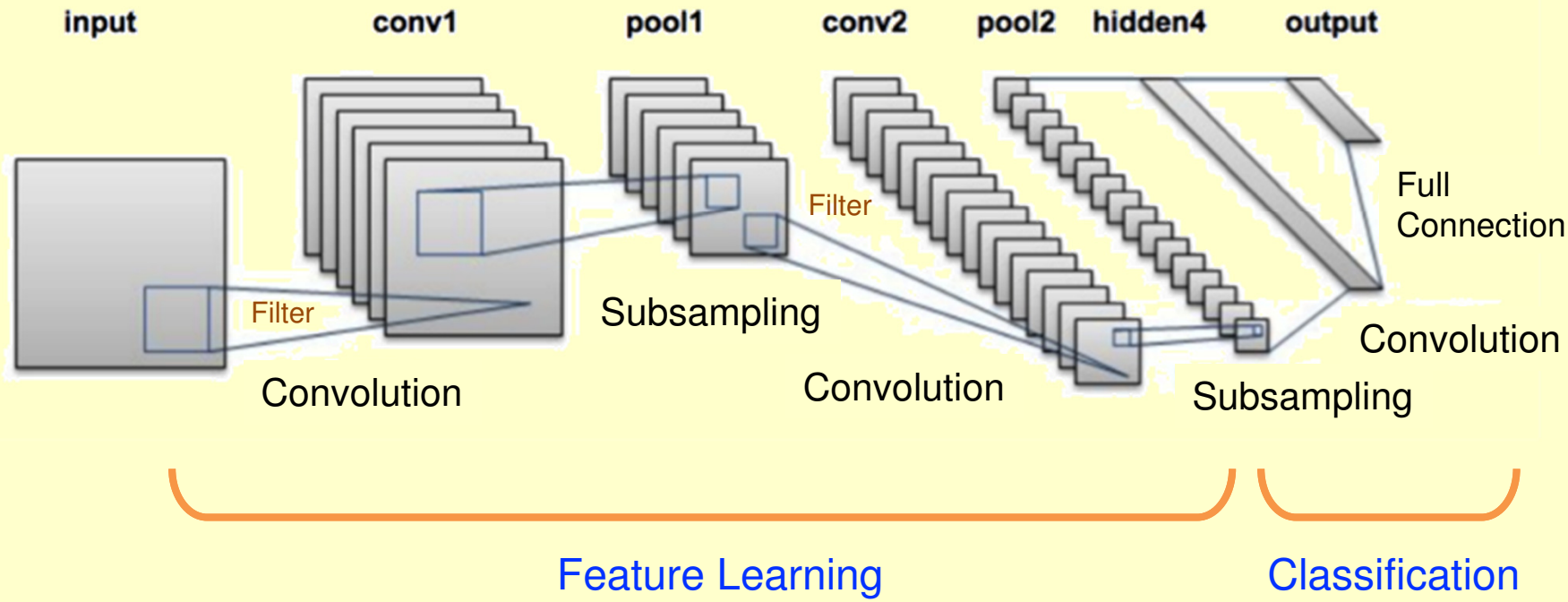
- Neural Network with convolution operator
- Specifically for problems with low dimensional grid-like structure (e.g. 2D)
- Sparse representation

Sparse Representation

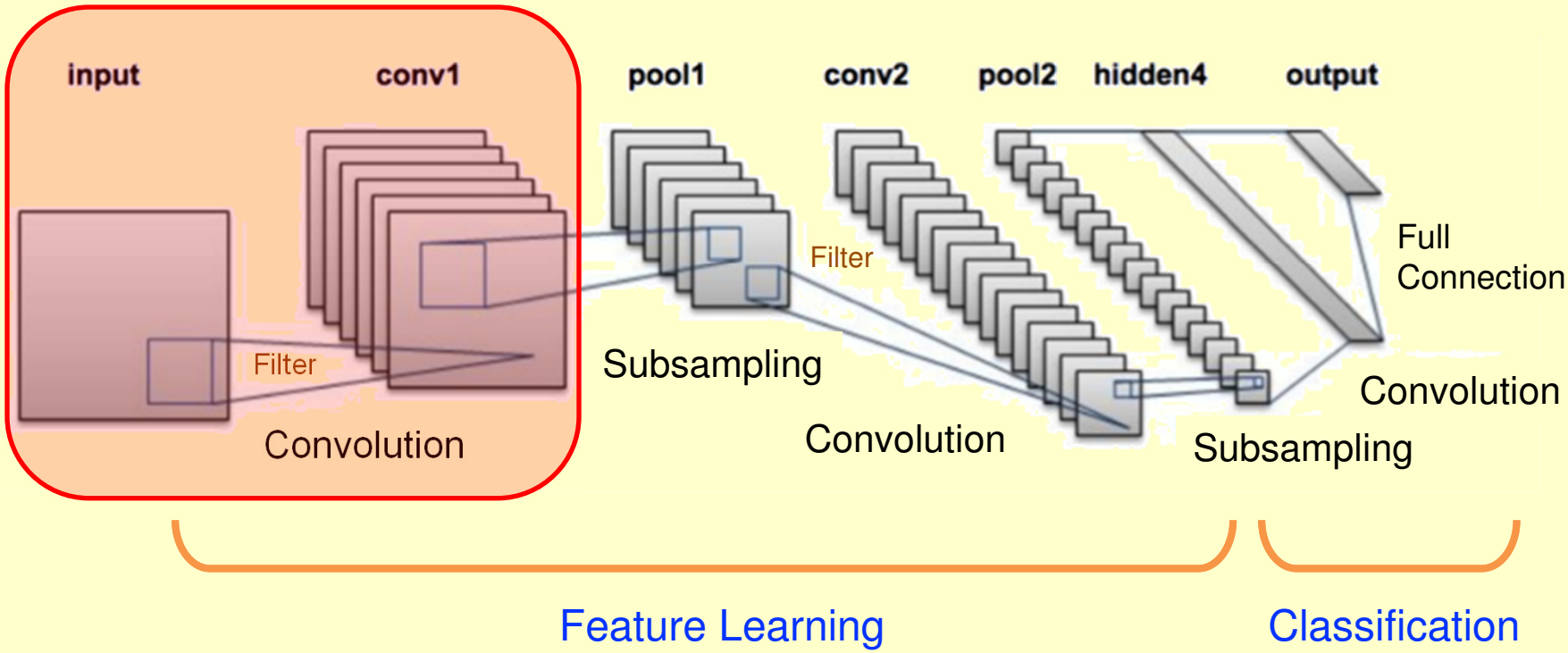




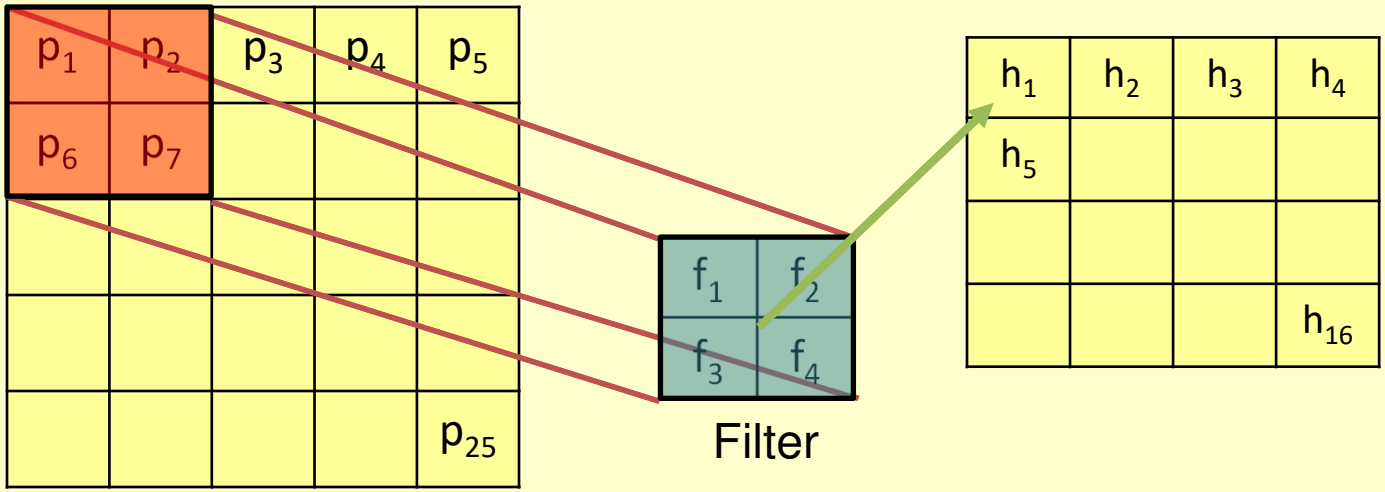
Deep Learning for CNN



Deep Learning for CNN



Learning for Convolution Layer



7	2	5	1	4
2	8	9	0	6
3	5	2	5	6
7	9	1	2	3
5	4	8	6	4

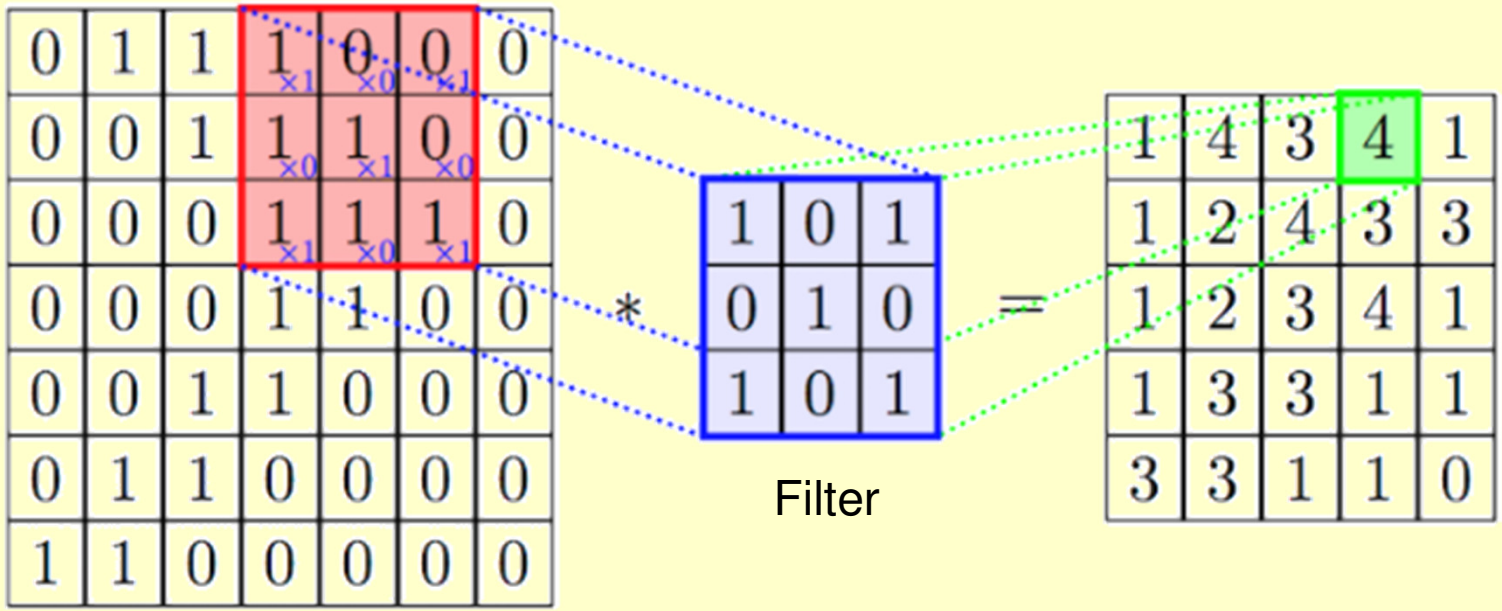
X

1	0
0	1

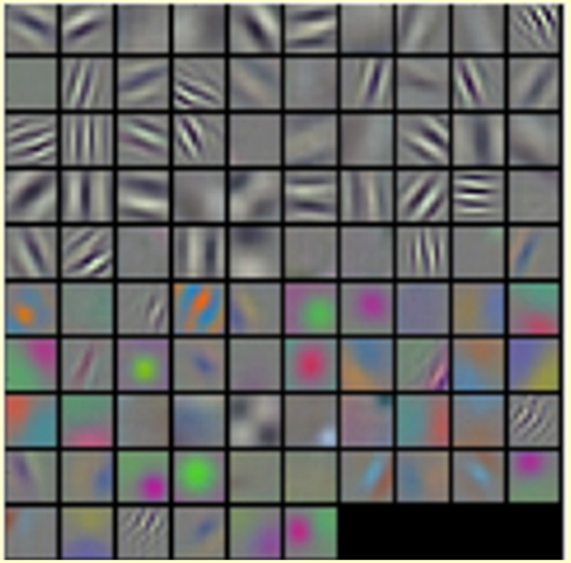
=

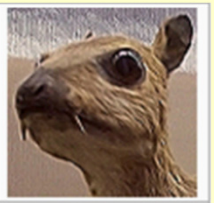
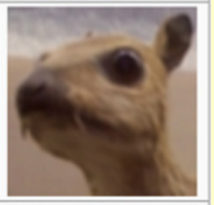
15	11	5	7
7	10	14	6
12	6	4	8
11	17	7	6

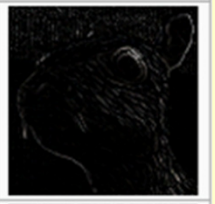
Filter


0 1 1 1 0 0 0
0 0 1 1 1 0 0
0 0 0 1 1 1 0
0 0 0 1 1 0 0
0 0 1 1 0 0 0
0 1 1 0 0 0 0
1 1 0 0 0 0 0
×1 ×0 ×1
×0 ×1 ×0
×1 ×0 ×1
*
1 0 1
0 1 0
1 0 1
Filter
=
1 4 3 4 1
1 2 4 3 3
1 2 3 4 1
1 3 3 1 1
3 3 1 1 0

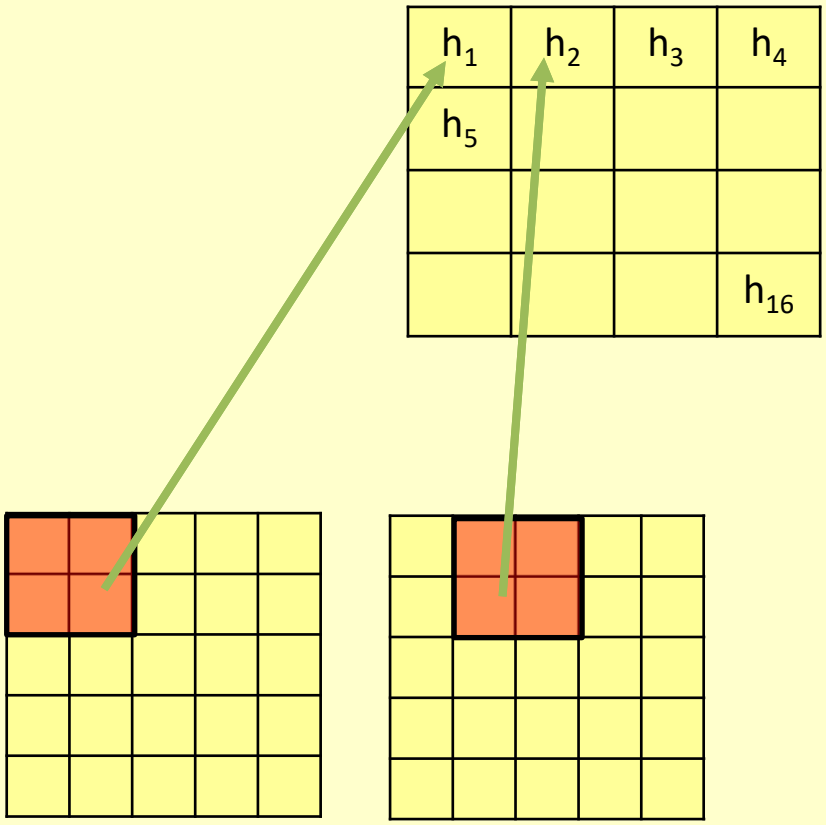
Filter Visualization

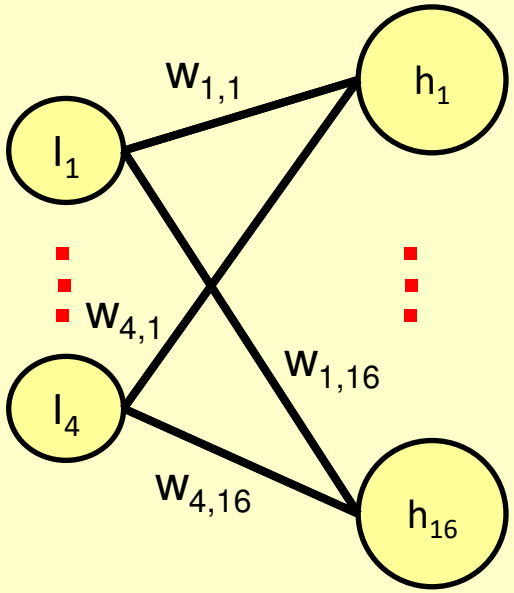


Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

Shared Parameters in Convolution with Filter





$w_{1,1}=w_{1,2}= \ldots = w_{1,16}$

7	2	5	1	4
2	8	9	0	6
3	5	2	5	6
7	9	1	2	3
5	4	8	6	4

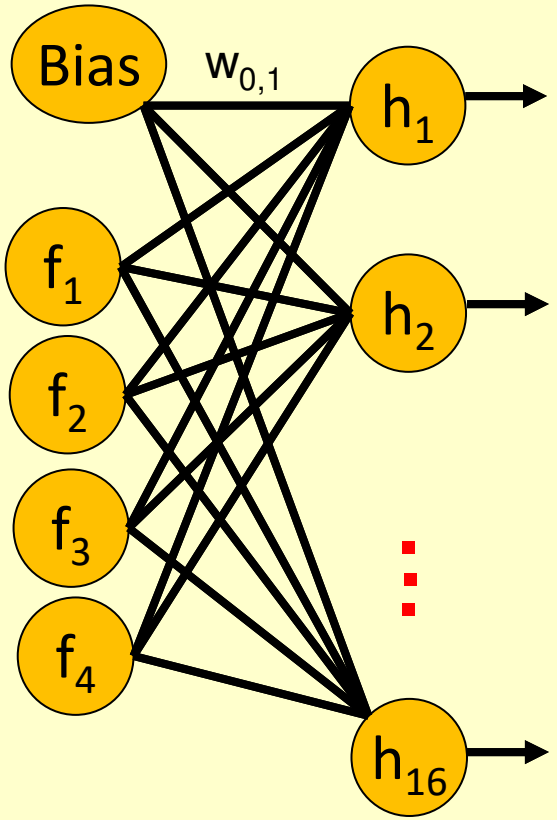
X

1	0
0	1

=

15	11	5	7
7	10	14	6
12	6	4	8
11	17	7	6

	f_1	f_2	f_3	f_4	Target
p_1	7	2	2	8	15
p_2	2	5	8	9	11
p_3	5	1	9	0	5
p_4	1	4	0	6	7
p_5	2	8	3	5	7
...					...
p_{25}	2	3	6	4	6

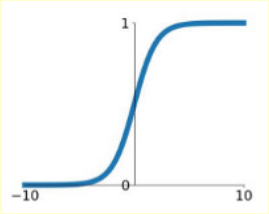


$w_{1,1}=w_{1,2}= \ldots = w_{1,16}$

Activation Function

- Sigmoid

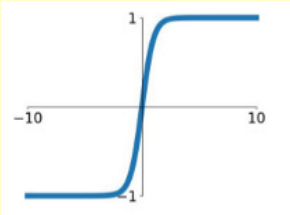
$$y = \frac{1}{(1+e^{-x})}$$



$$y' = y * (1-y)$$

- Hyperbolic Tangent (Tanh)

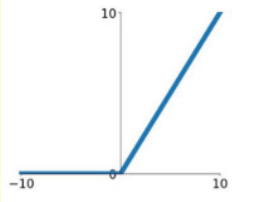
$$y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



$$y' = 1-y^2$$

- Rectified Linear Units (ReLU)

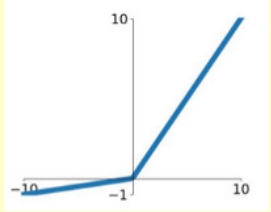
$$y=Max(0,x)$$



y'=1 → if x>0
y'=0 → otherwise

- Leaky ReLU

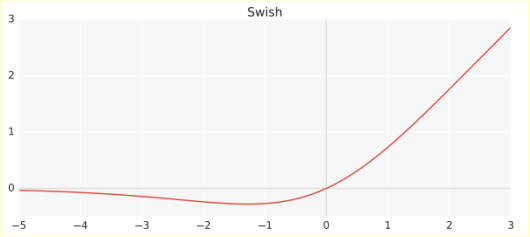
if x>0 → y = x
else → y = ax where $0 \leq a \leq 1$



y'=1 → if x>0
y'=a → otherwise

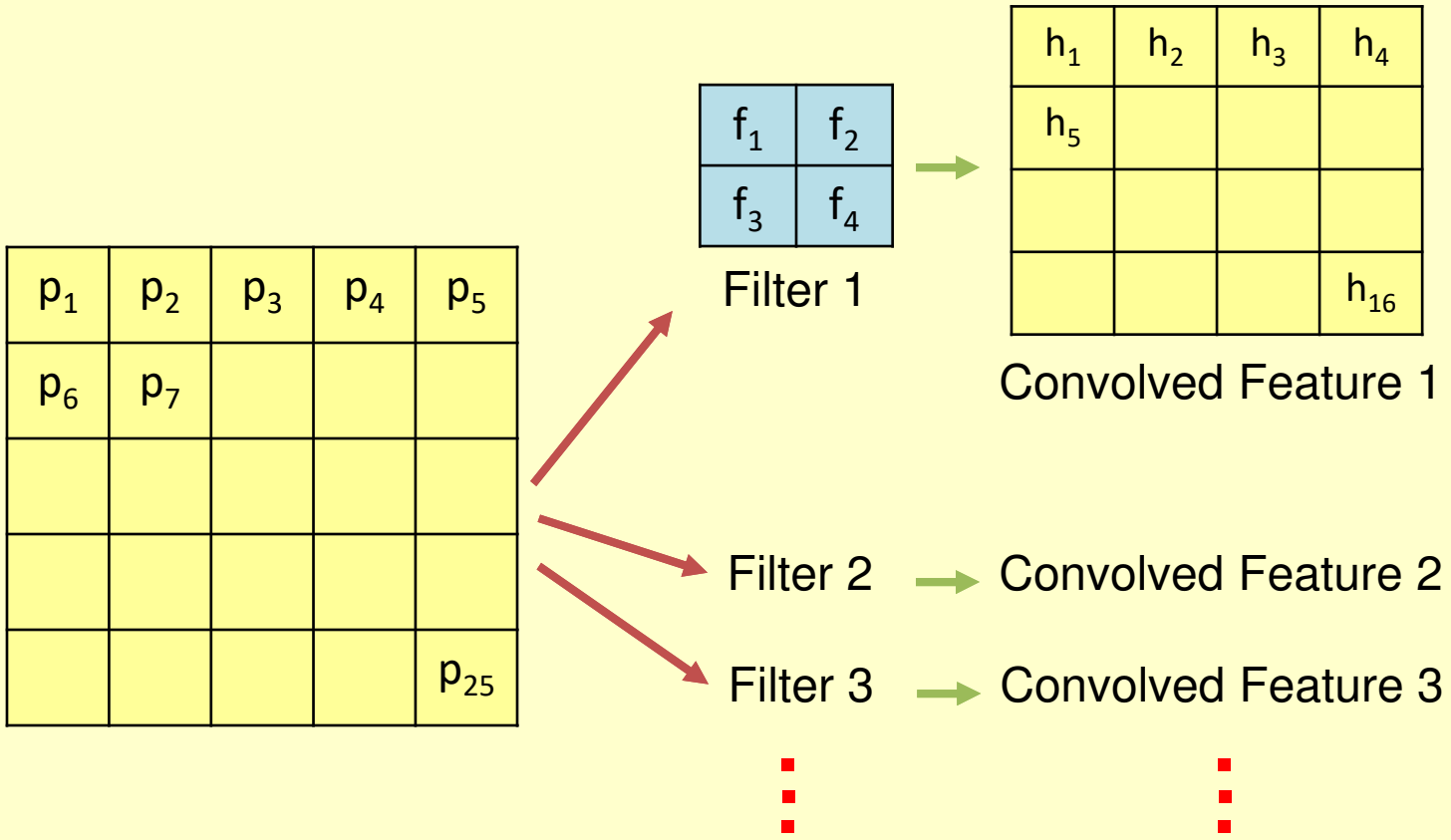
- Swish

$$y = \frac{x}{(1+e^{-x})}$$

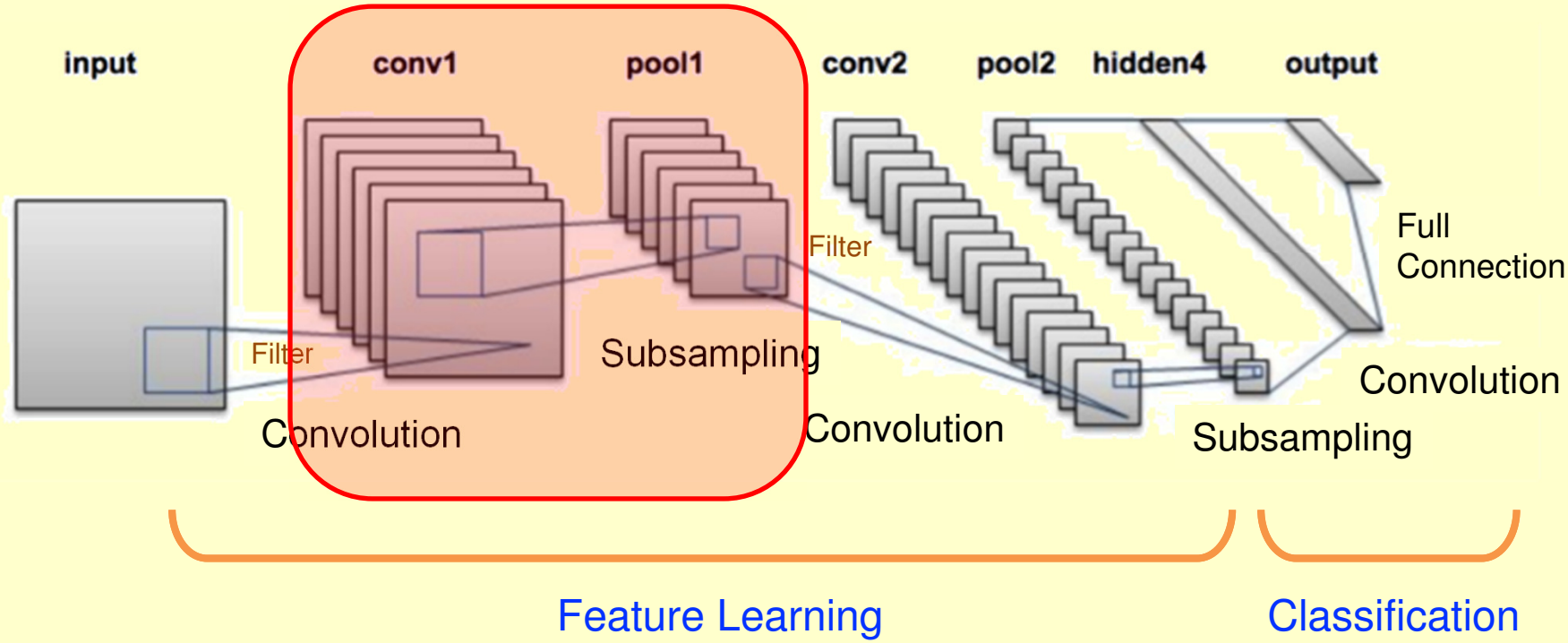


$$y' = y + x \,.\, (1-y)$$

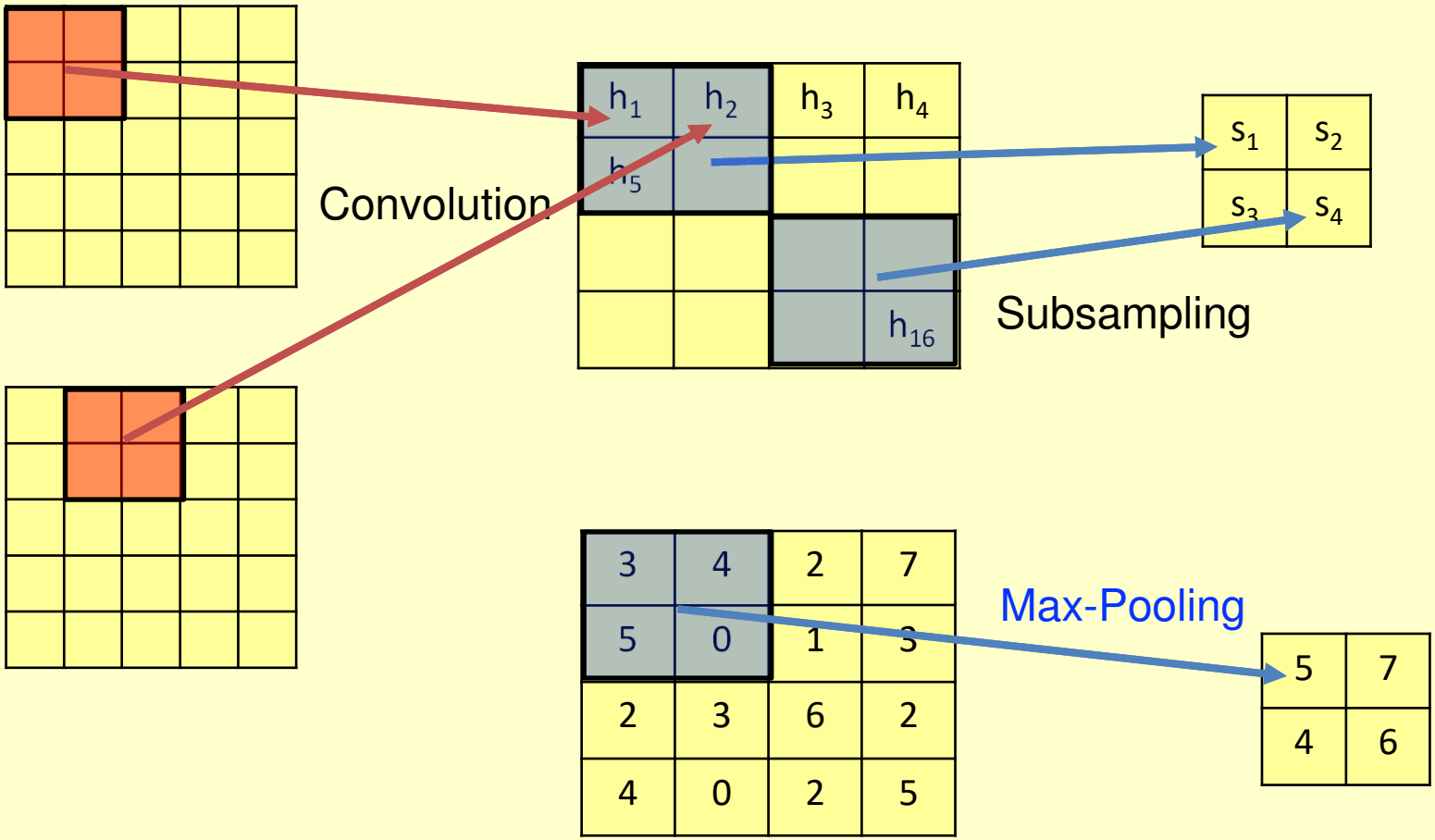
Learning for Convolution Layer



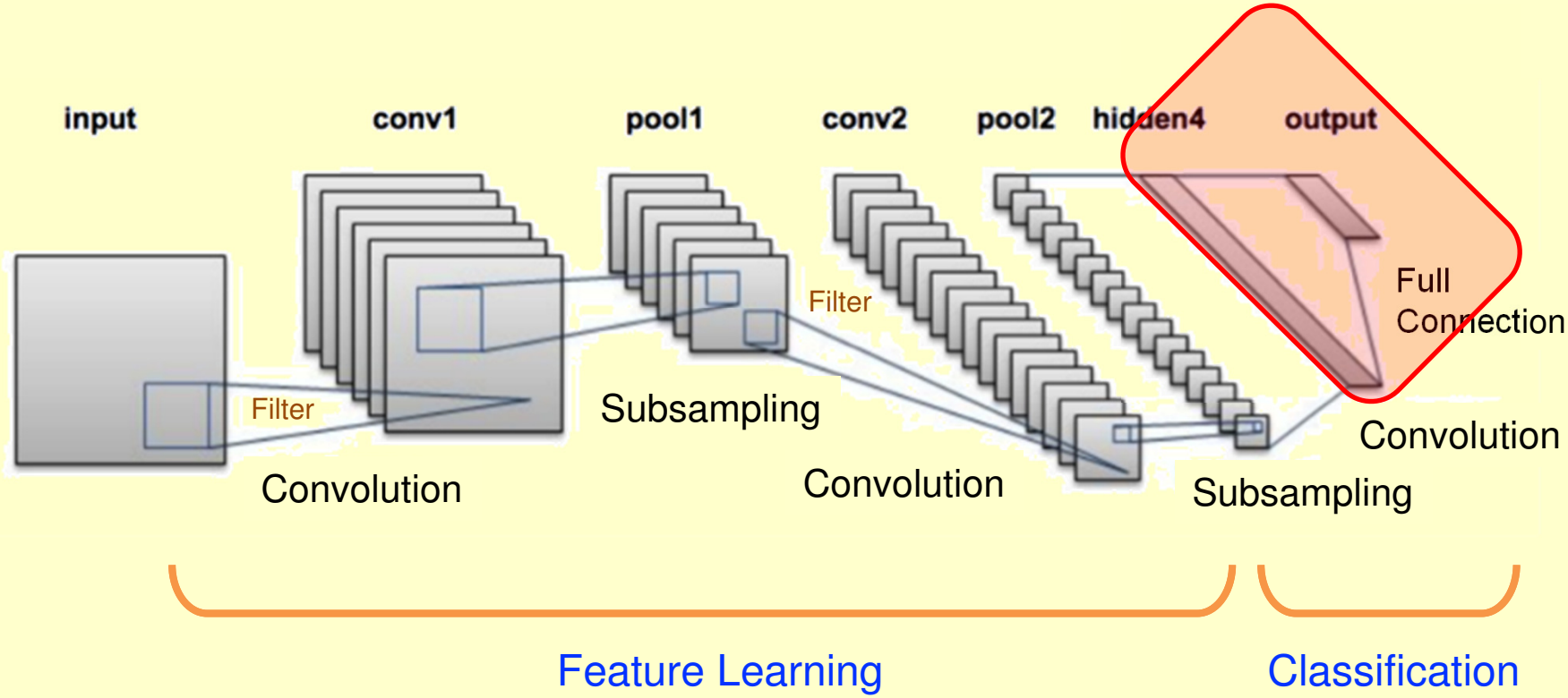
Deep Learning for CNN



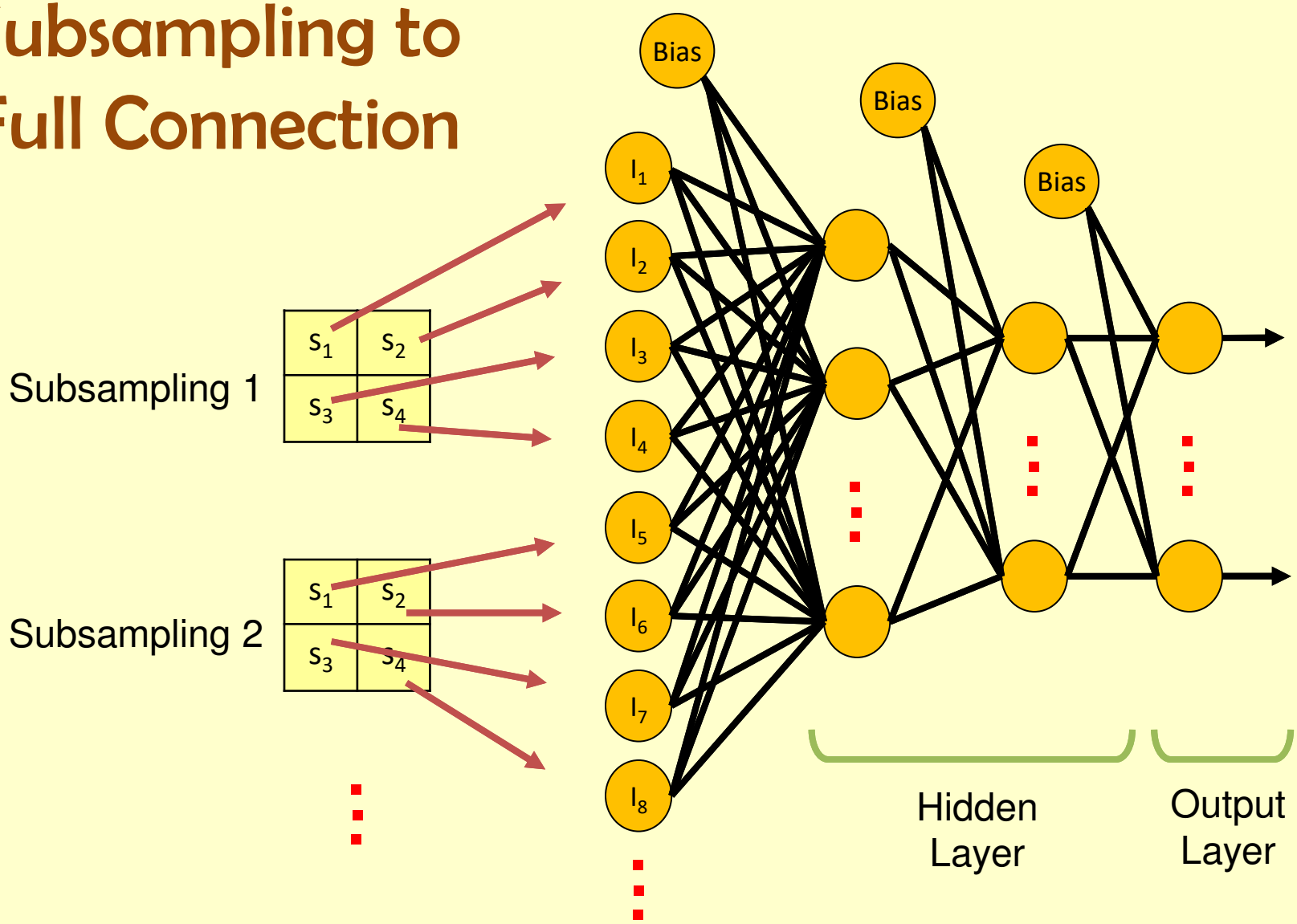
Subsampling (Pooling)



Deep Learning for CNN

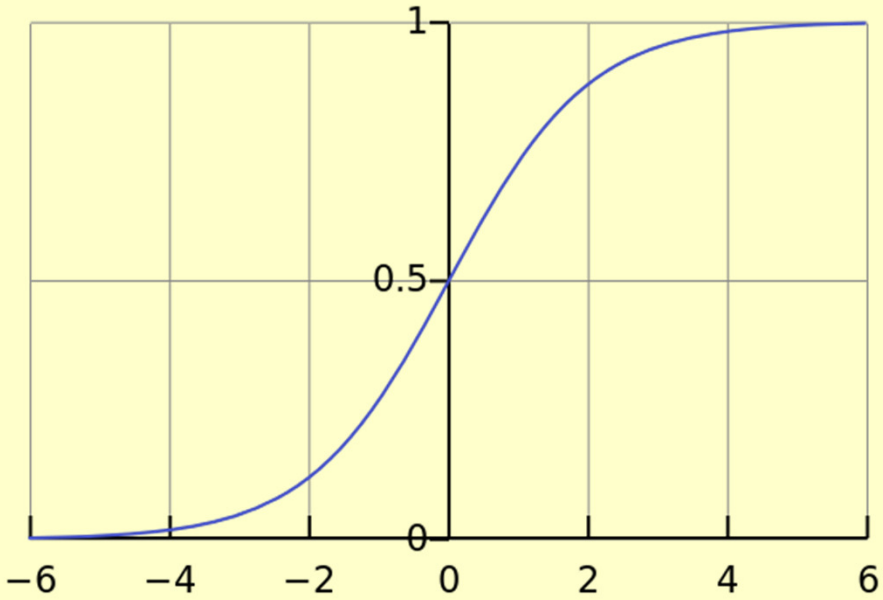


Subsampling to Full Connection



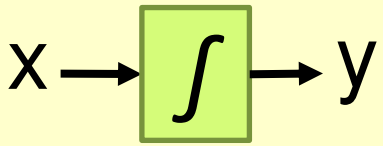
Activation Function for Hidden Layer in Full Connection

Sigmoid

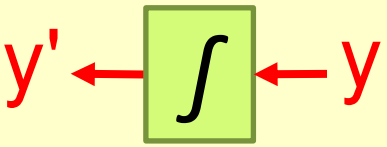


$$f(x) = \frac{1}{(1+e^{-x})}$$

Derivative Sigmoid Function

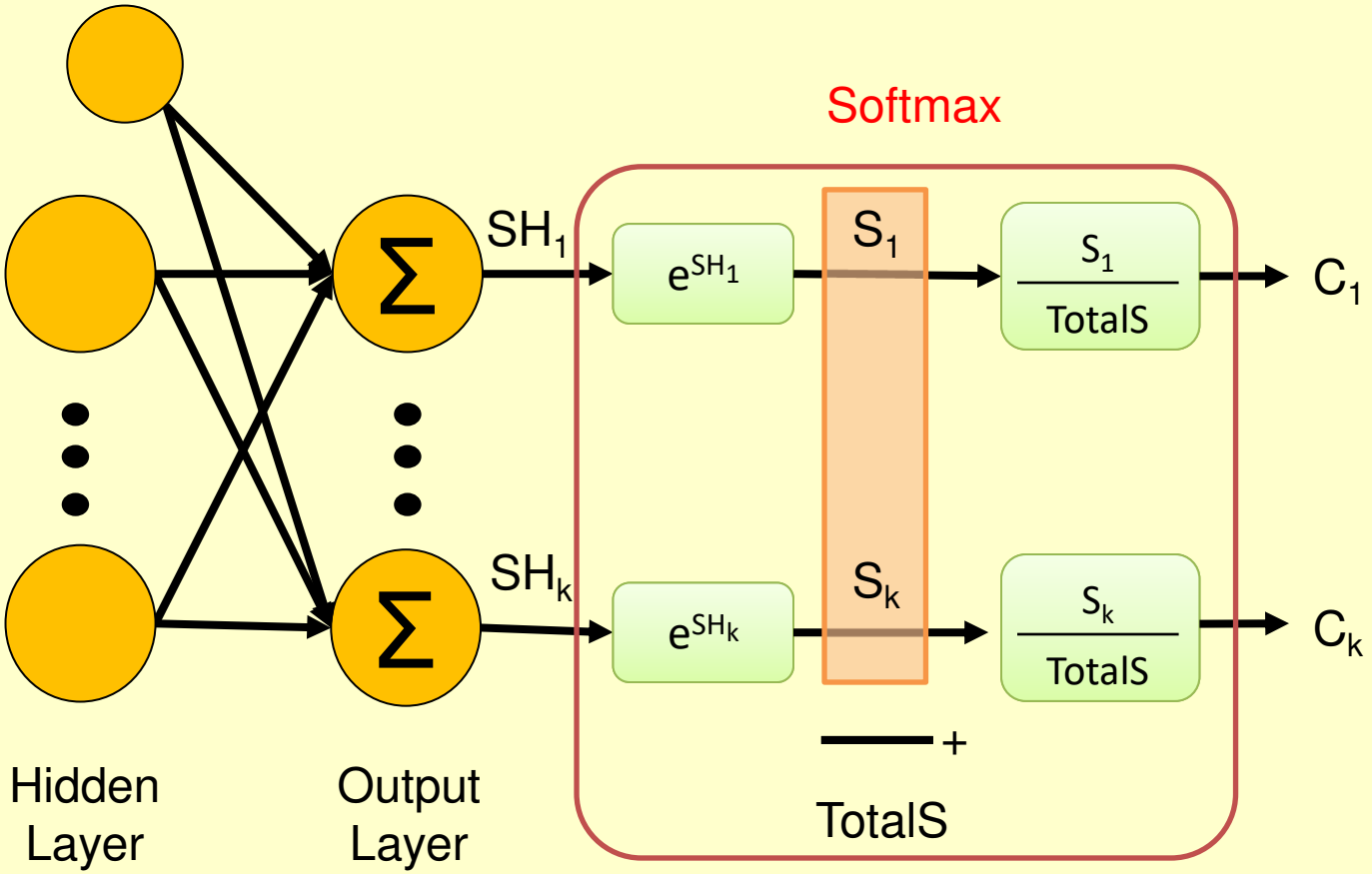


$$y = \frac{1}{(1+e^{-x})}$$

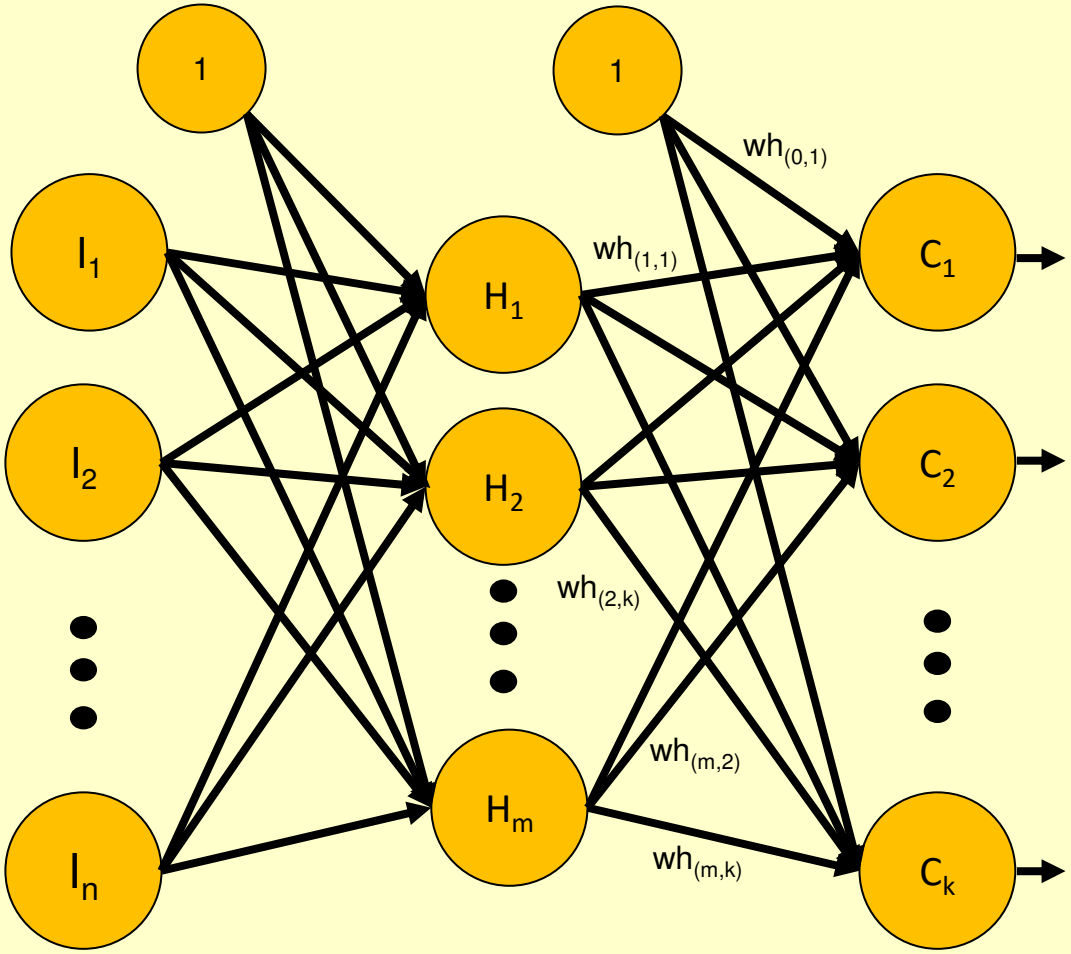


$$y' = y * (1-y)$$

Activation Function for Output Layer in Full Connection



Learning Performance Analysis



$$(Target_1-C_1)^2 + (Target_2-C_2)^2 + \dots + (Target_k-C_k)^2$$

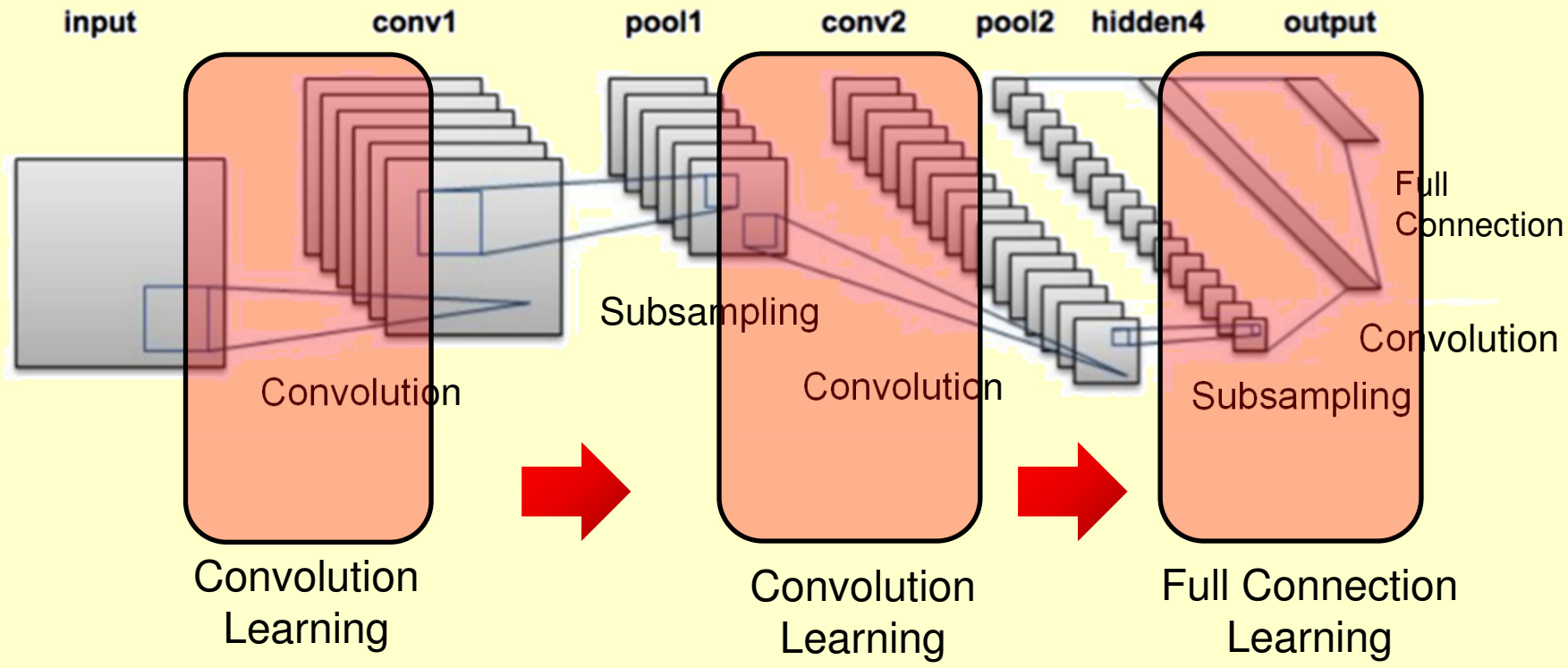
$$SSE=\sum_{d=1}^{f} SE_d$$

f ← total input sequence

$$MSE=\frac{SSE}{f}$$

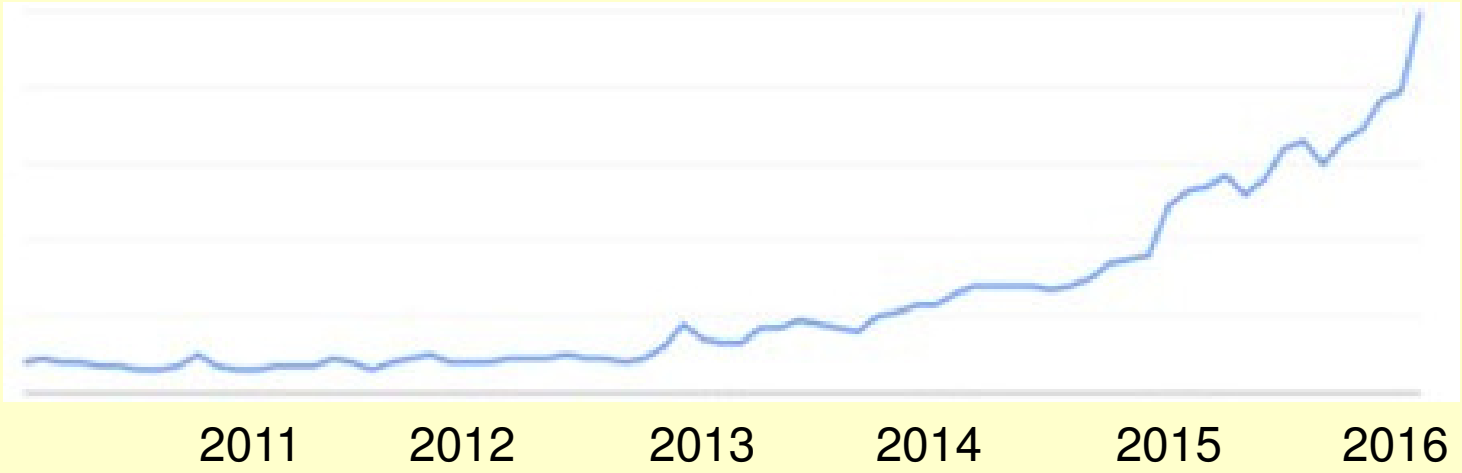
SE_d → Squared Error pada input sequence d

Learning Processes



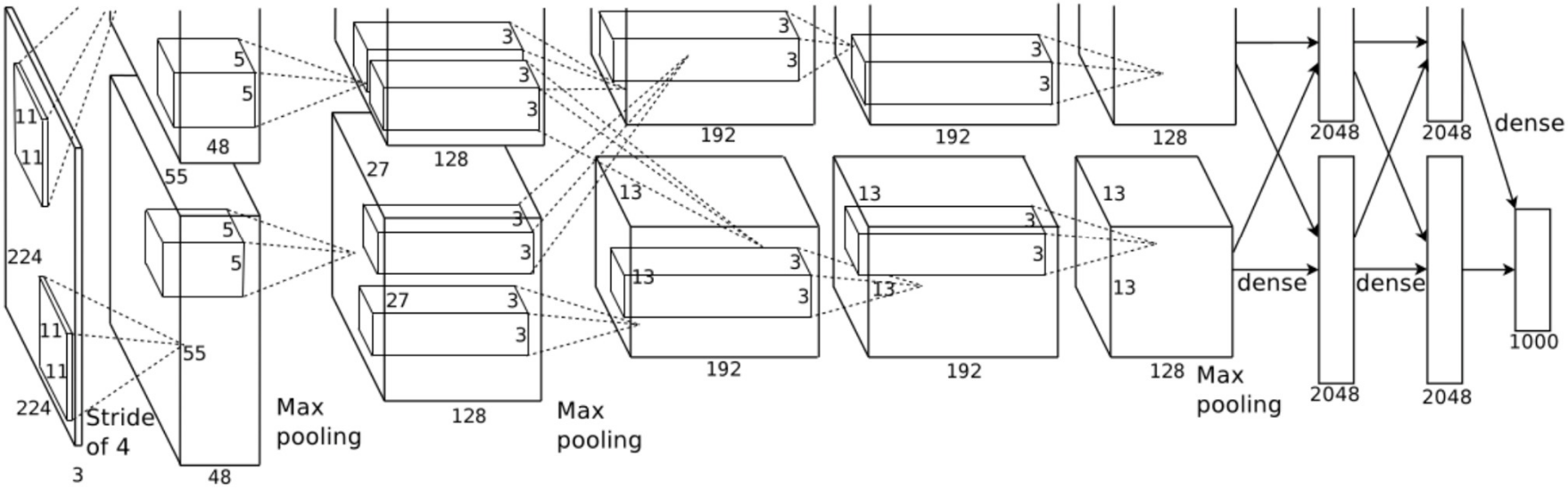
Deep Learning Trend

Source: Google Trends



Alexnet Architecture

Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton (2013)



- implemented CNN [LeCun et.al. 1998]
- trained on Imagenet with 2 GPUs
- used ReLU instead of Sigmoid or Tanh
- Dropout mechanism

Imagenet

http://www.image-net.org/

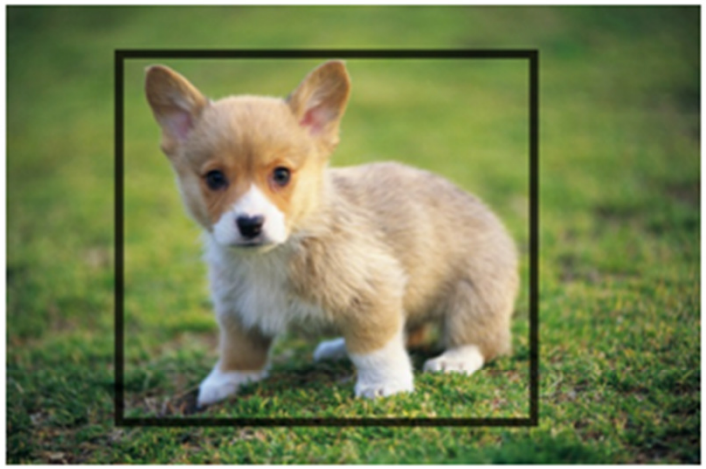
- Imagenet was published [Deng et.al. 2009], consists of 16 million images annotated by human, with term of Wordnet (100.000 classes)
- Subset of Imagenet → ILSVRC (Imagenet Large Scale Visual Recognition Challenge), consists of 1 million images with 1000 classes and challenges Top-5 and Top-1 error measured

- 2010 → Large-scale Support Vector Machine performed 28% error rate
- 2011 → Compressed Fisher Vector approach with 26% error
- 2012 → Alexnet performed impressive 16% Top-5 error, while the second best around 26%
- 2013 → Clarifai performed 11.7% error
- 2014 → GoogLeNet performed 6.6% error with 22 layers
- 2015 → Microsoft Research Asia performed 3.56% error with 150 layers
- 2016 → Trimps-Soushen (Chinese AI teams, supported by the Chinese Ministry of Public Security) performed 2.99% error
- 2017 → Trimps-Soushen performed 2.48% error

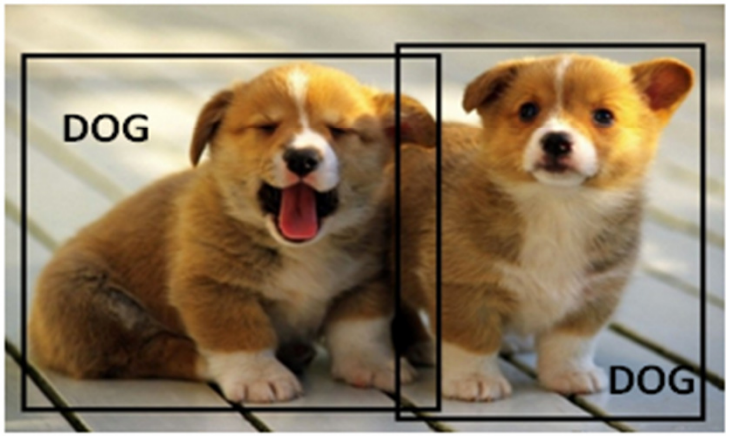
Today, Deep Learning is almost everywhere

- Object classification, detection, segmentation and localization
- Image captioning, question answering
- Machine translation
- Speech recognition
- Robotics
- Object retrieval and tracking

Object Classification is the task of identifying that picture is a dog



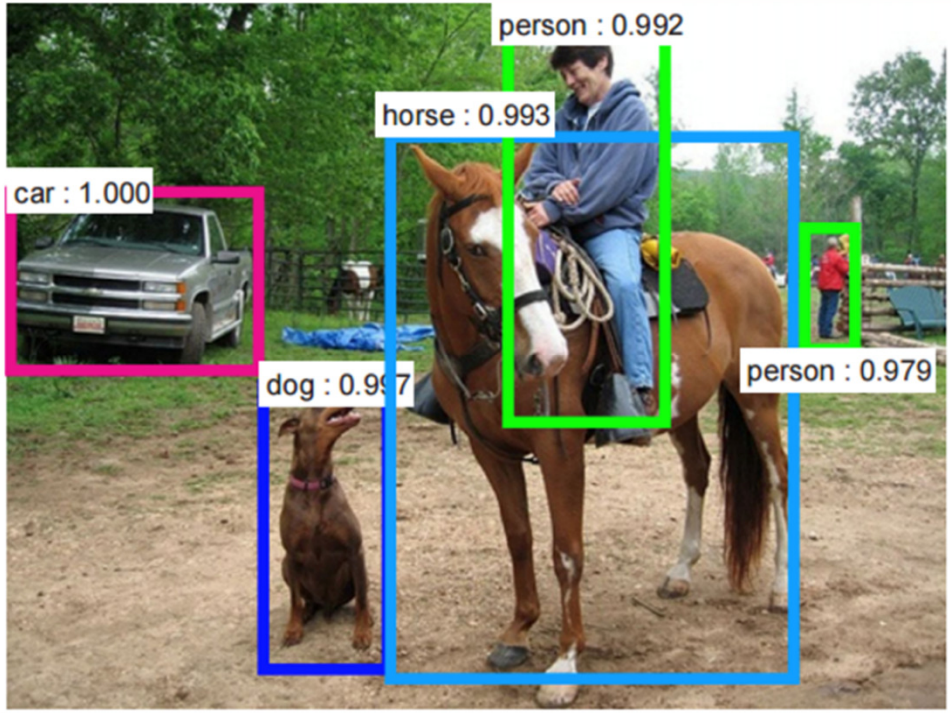
Object Localization involves the class label as well as a bounding box to show where the object is located.

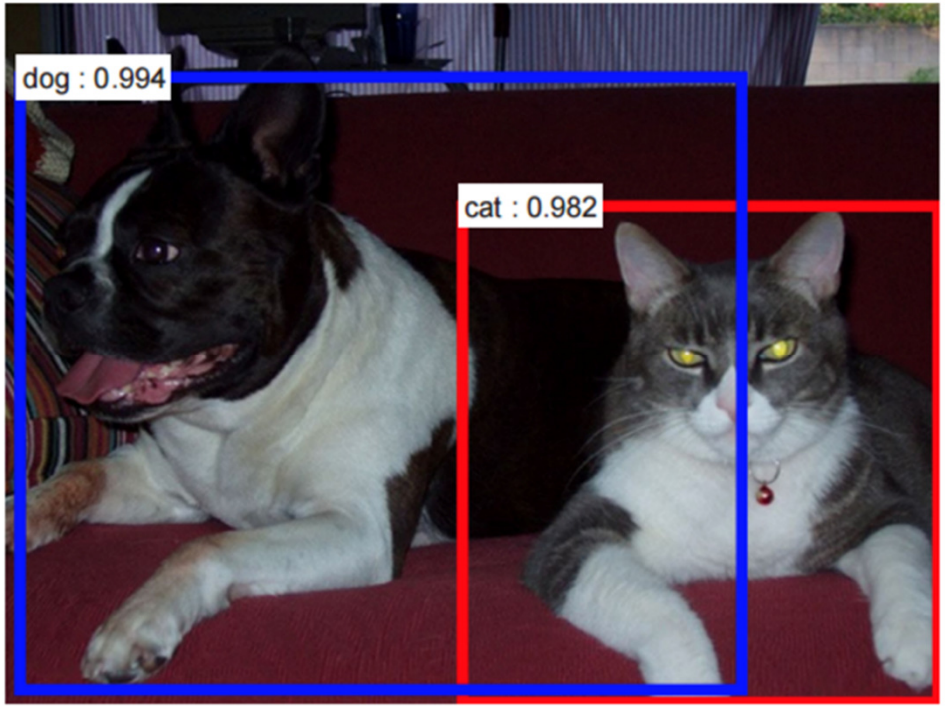


Object Detection involves localization of multiple objects (doesn't have to be the same class).

Object Segmentation involves the class label as well as an outline of the object in interest.

Source: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/

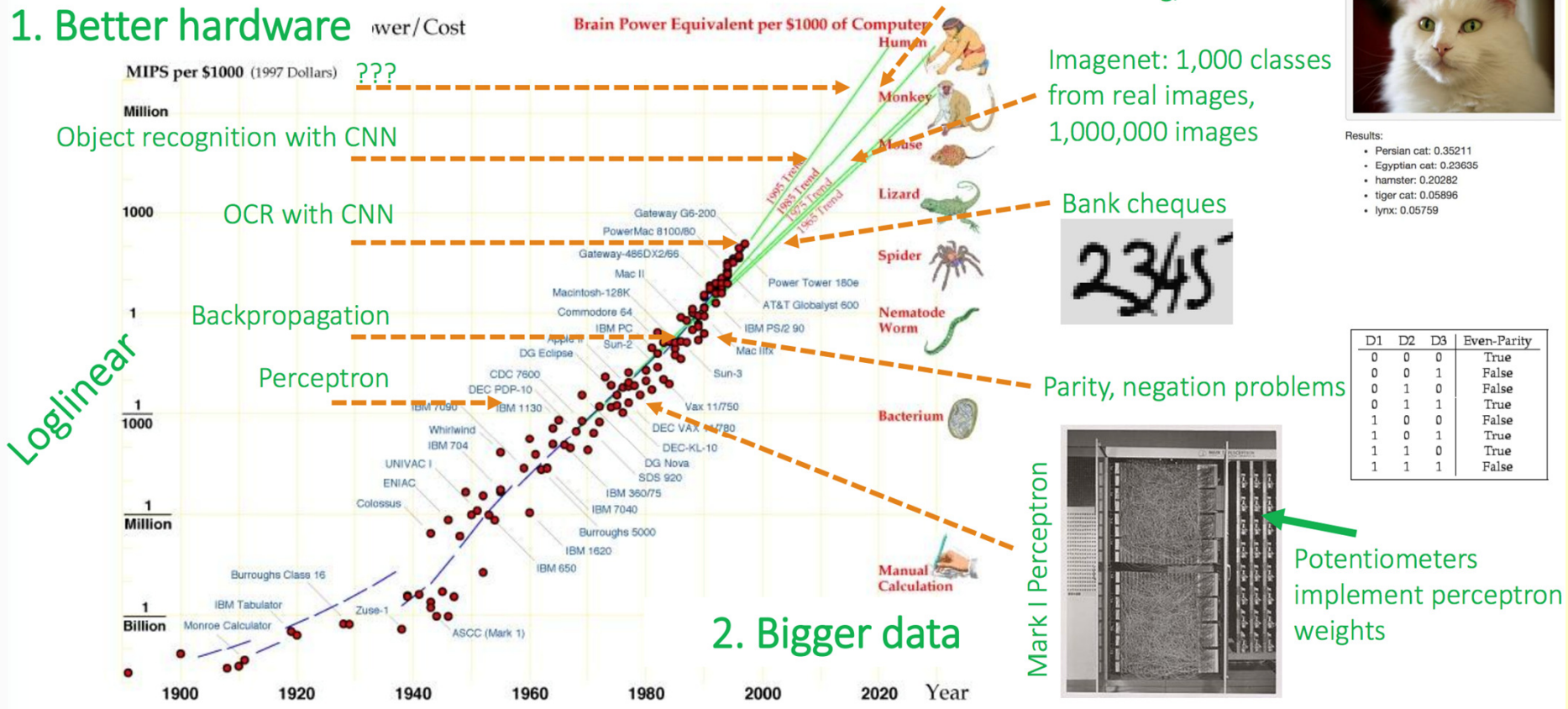




Source: Shaoqing Ren, *et al,* "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", 2015

So, why now?

1. Better hardware



Datasets of everything (captions, question-answering, …), reinforcement learning, ???

Imagenet: 1,000 classes from real images, 1,000,000 images

Results:
- Persian cat: 0.35211
- Egyptian cat: 0.23635
- hamster: 0.20282
- tiger cat: 0.05896
- lynx: 0.05759

Bank cheques

Parity, negation problems

D1	D2	D3	Even-Parity
0	0	0	True
0	0	1	False
0	1	0	False
0	1	1	True
1	0	0	False
1	0	1	True
1	1	0	True
1	1	1	False

Mark I Perceptron

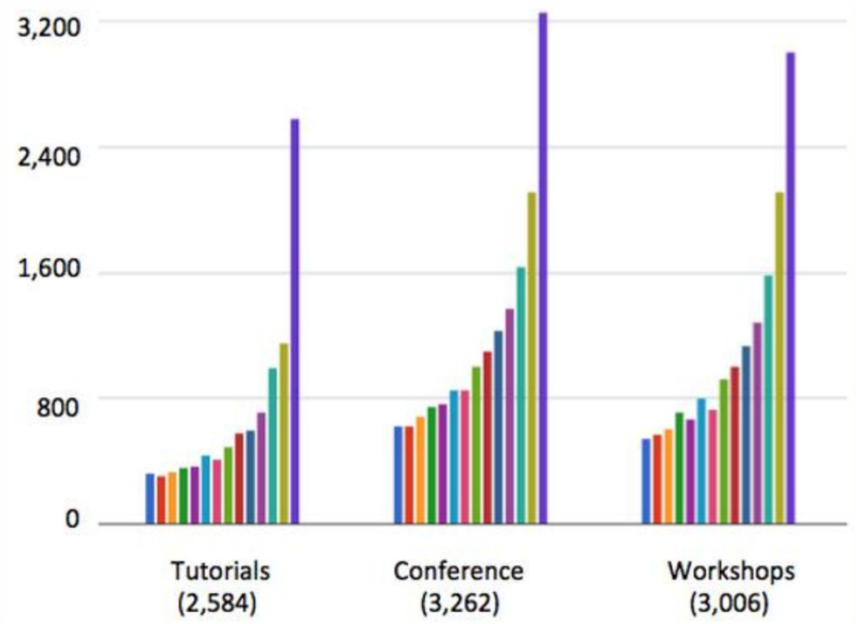
Potentiometers implement perceptron weights

2. Bigger data

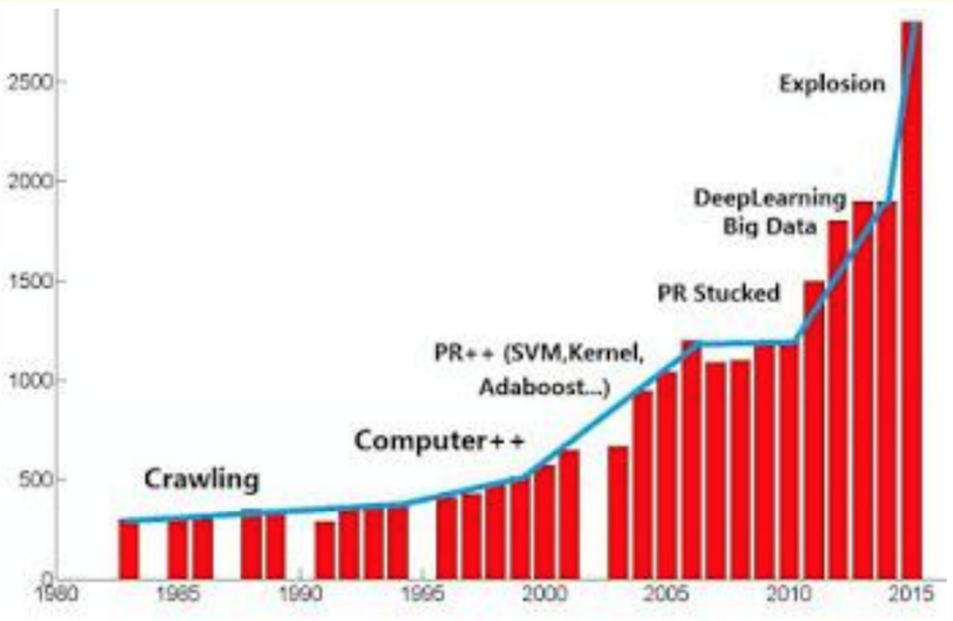
Source: UVA Deep Learning Course

→ Better hardware
→ Bigger data
→ Better regularization methods
→ Better optimization methods

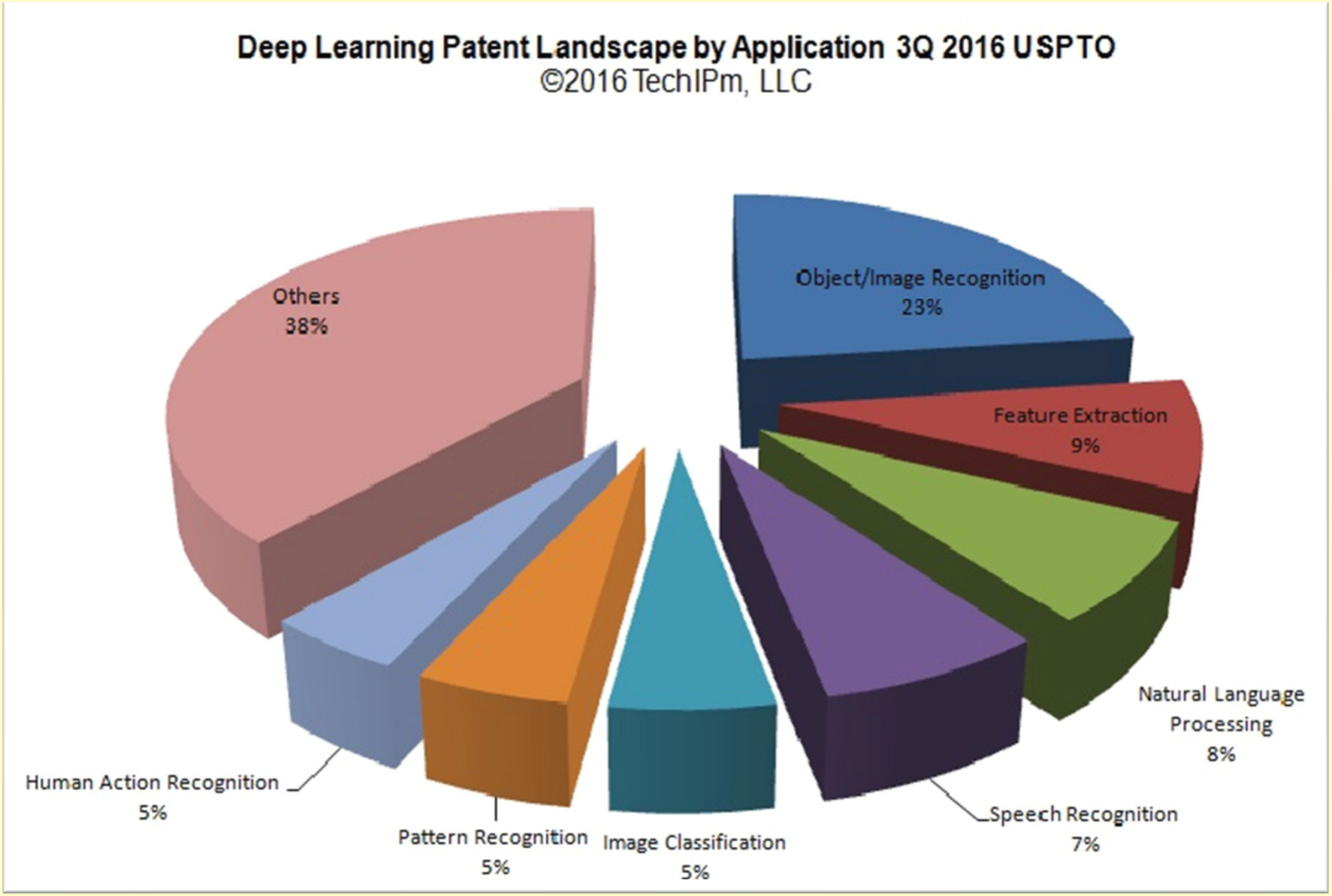
Discussion Trend for Deep Learning discussions

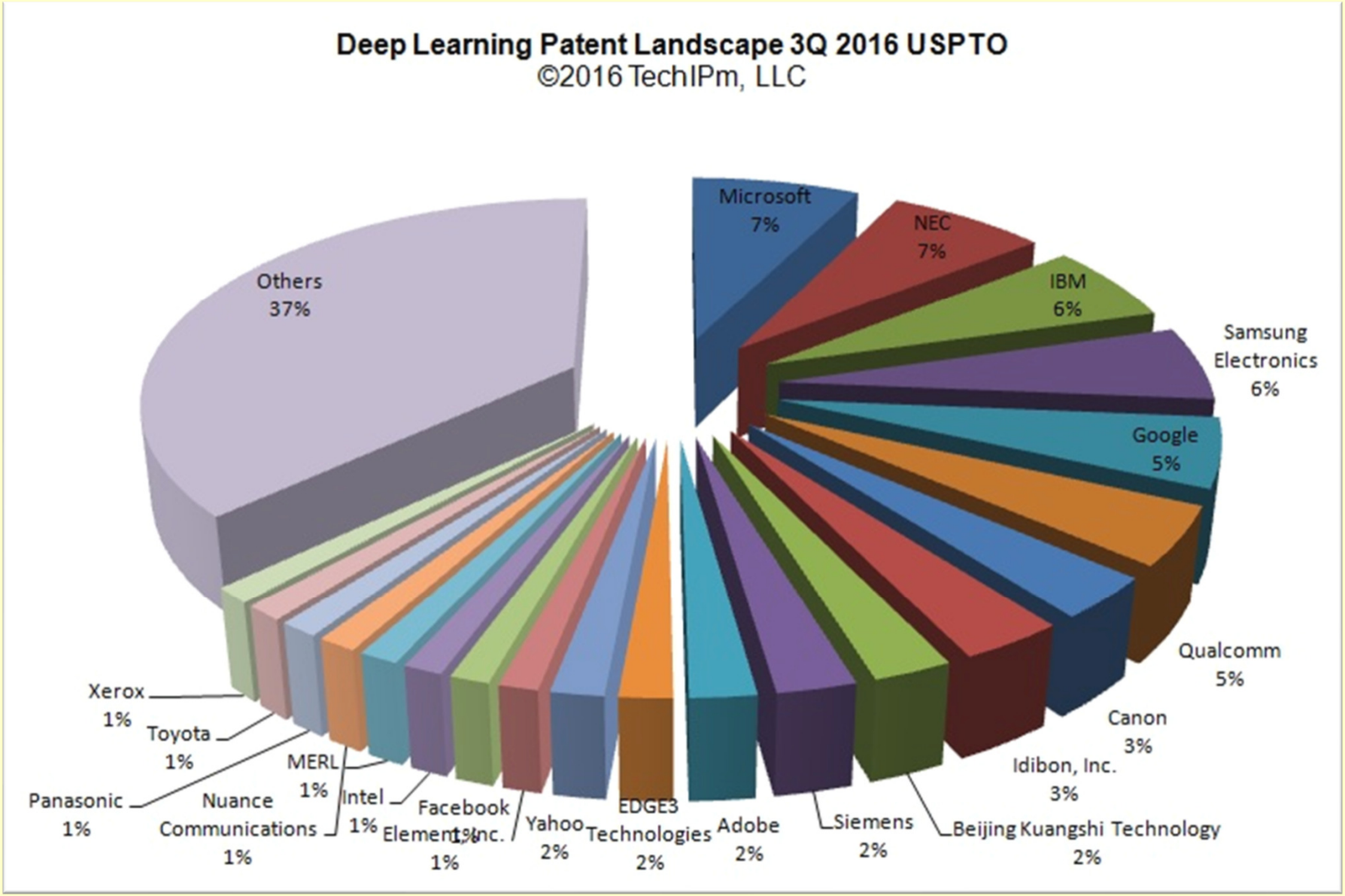


Conference on Neural Information Processing Systems



Conference on Computer Vision and Pattern Recognition


Deep Learning Patent Landscape by Application 3Q 2016 USPTO
©2016 TechIPm, LLC
Others
38%
Object/Image Recognition
23%
Feature Extraction
9%
Natural Language
Processing
8%
Speech Recognition
7%
Image Classification
5%
Pattern Recognition
5%
Human Action Recognition
5%


Deep Learning Patent Landscape 3Q 2016 USPTO
©2016 TechIPm, LLC
Microsoft 7%
NEC 7%
IBM 6%
Samsung Electronics 6%
Google 5%
Qualcomm 5%
Canon 3%
Idibon, Inc. 3%
Beijing Kuangshi Technology 2%
Siemens 2%
Adobe 2%
EDGE3 Technologies 2%
Yahoo 2%
Facebook 1%
Element Inc. 1%
Intel 1%
MERL 1%
Nuance Communications 1%
Panasonic 1%
Toyota 1%
Xerox 1%
Others 37%

References

- Weifeng Li, Victor Benjamin, Xiao Liu, Hsinchun Chen, “Deep Learning: An Overview”, University of Arizona, April 2015.
- Hung-Yi Lee, “Deep Learning Tutorial”, Speech Processing and Machine Learning Laboratory, Department of Electrical Engineering, National Taiwan University
- Tony Martinez, “Deep Learning”, Neural Network and Machine Learning Laboratory, Brigham Young University
- Barnabas Pochos, Aati Singh, “Introduction to Machine Learning (Deep Learning)”, Machine Learning Department, Carnegie Mellon
- David Corne, “An Introduction to Deep Learning”, School of Mathematical and Computer Sciences, Heriot Watt University
- Efstratios Gavves, Kirill Gavrilyuk, Berkay Kicanaoglu, Patrick Putzky, “Introduction to Neural Networks and Deep Learning”, UVA Deep Learning Course, University of Amsterdam
- Mingxuan Sun, “Introduction to Deep Learning and Its Application”, Louisiana State University
- Honglak Lee, “Tutorial on Deep Learning and Aplications”, University of Machigan
- Yangyan Li, “A Brief Introduction to Deep Learning”, School of Computer Science and Technology, Shandong University
- Angjoo Kanazawa, “Convolutional Neural Networks”, University of Maryland for Advanced Computer Studies
- Nojun Kwak, “Introduction to Convolutional Neural Networks (CNNs)”, Department of Transdisciplinary Studies, Seoul National University, Jan 2016
- Boris Ginsburg, “Introduction: Convolutional Neural Networks for Visual Recognition”, Intel Science and Technology Centers-Computational Intelligence (ICRI-CI), Intel
- Vicky Kalogeiton, “Introduction to Convolutional Neural Networks”, Deep Learning reading Group, Inria, July 2016
- Fei-Fei Li, Andrej Karpathy, Justin Johnson, “Convolutional Neural Networks”, Computer Science Department, Stanford University, Jan 2016
- Prakash Jay, “Back-Propagation is very simple. Who made it Complicated?”, from website: becominghuman.ai