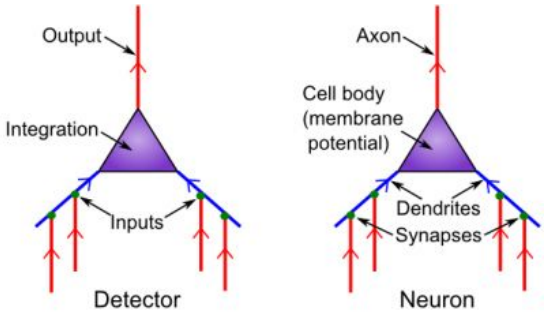
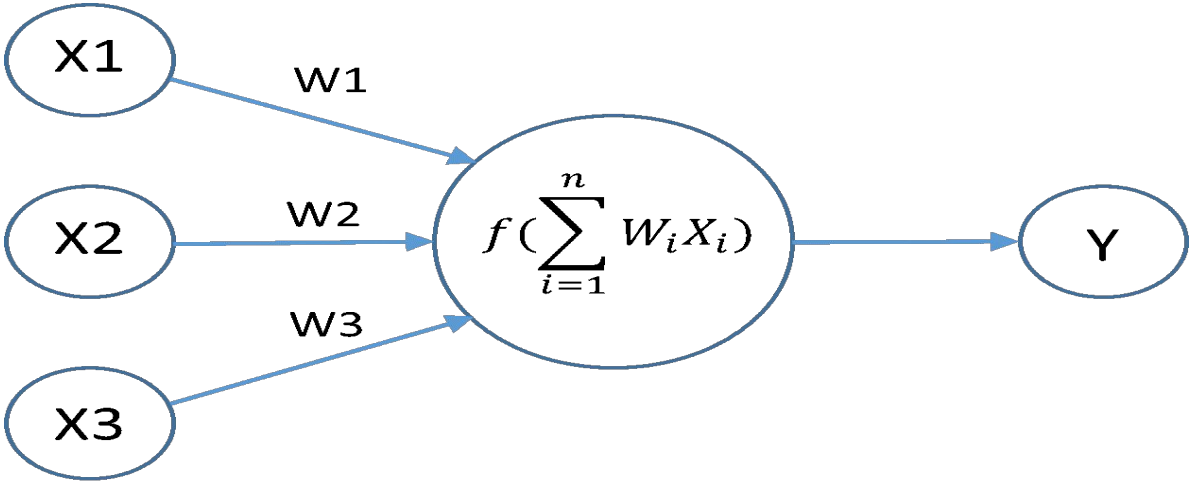
Konwolucyjne sieci neuronowe

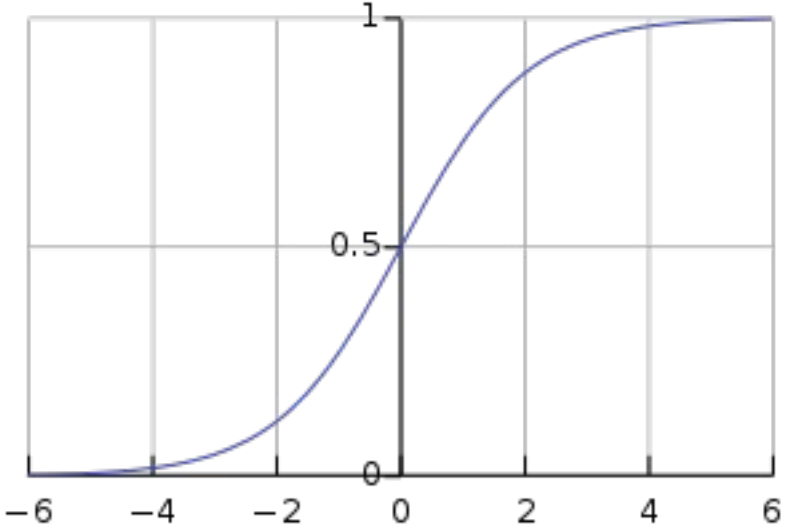
By Mateusz Macias

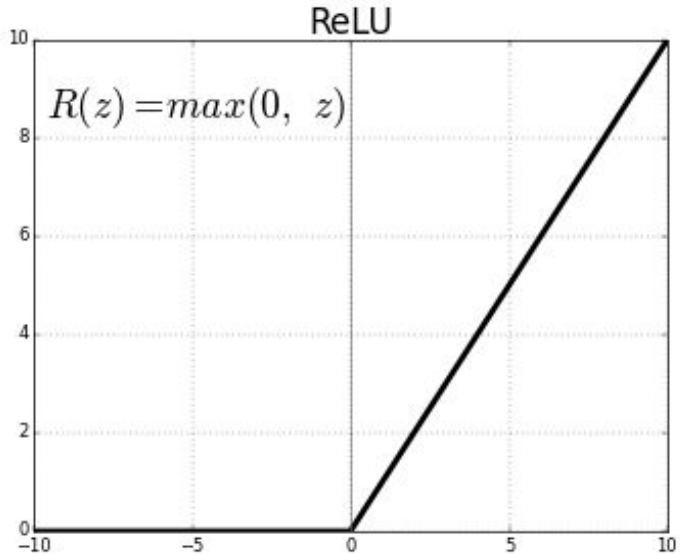
Neuron


Output
Integration
Inputs
Detector
Axon
Cell body
(membrane
potential)
Dendrites
Synapses
Neuron

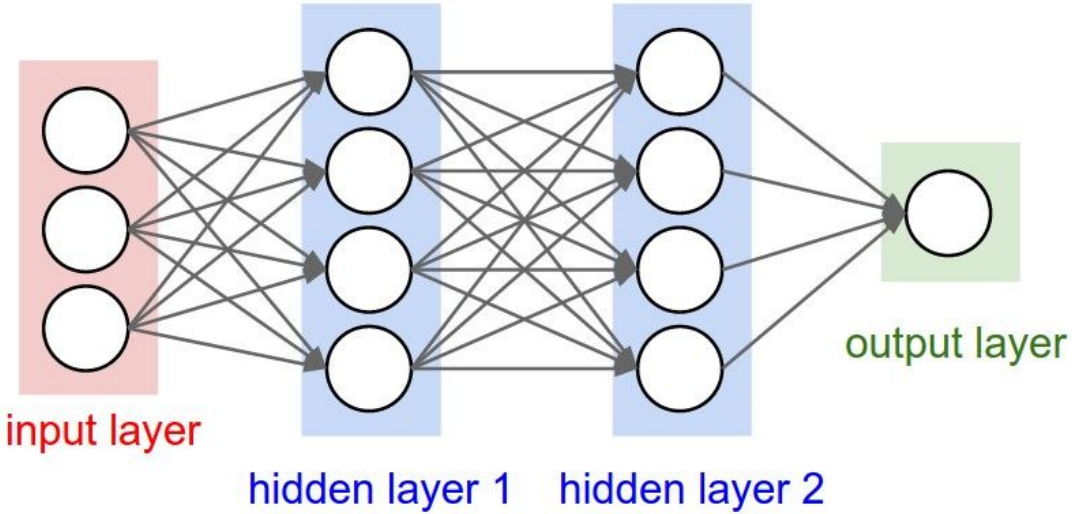

X1
X2
X3
W1
W2
W3
$f(\sum_{i=1}^{n} W_i X_i)$
Y

Sigmoid

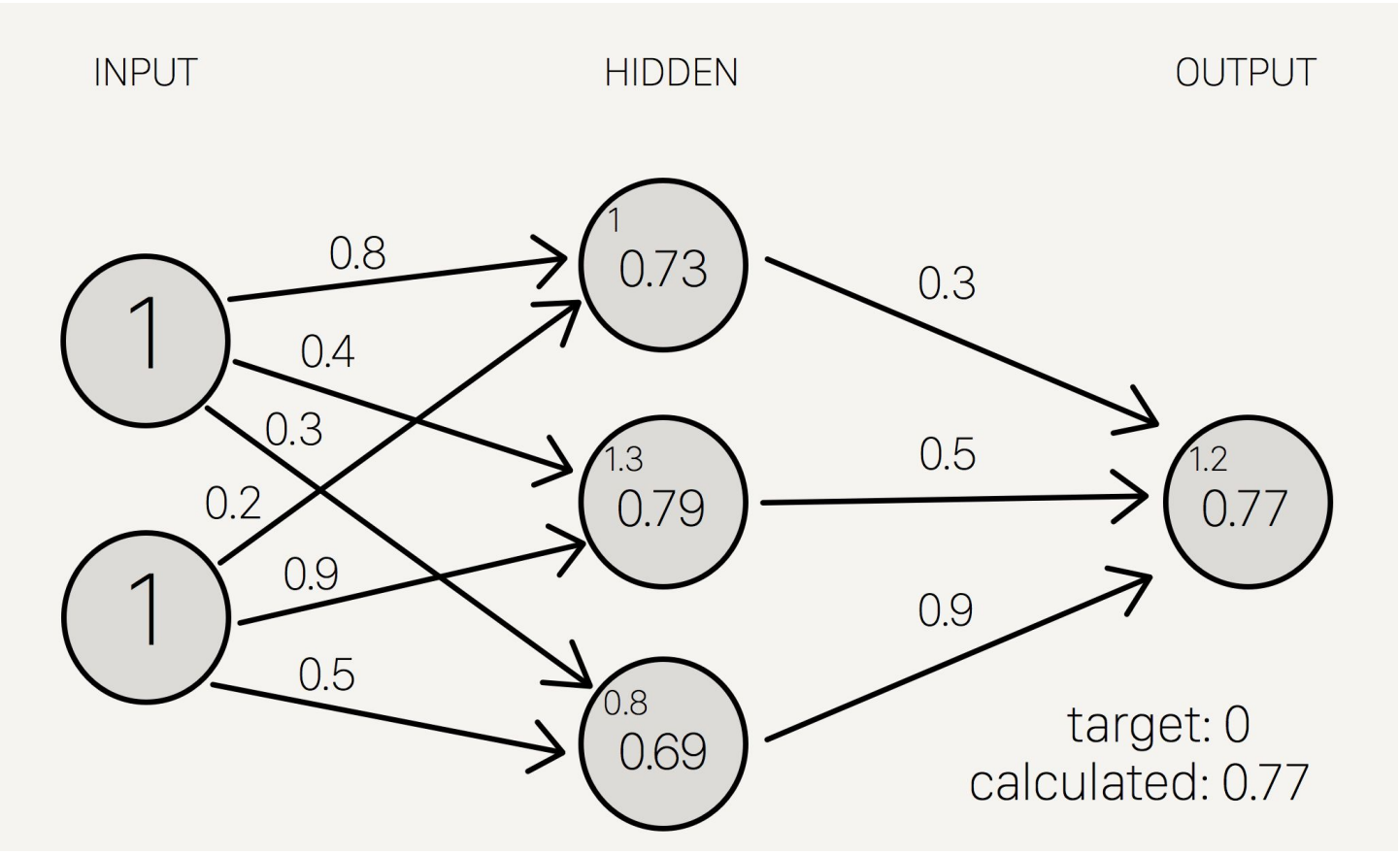




Sieć neuronowa



$$Y = f(W1*f(W2*f(W3*X)))$$


INPUT
HIDDEN
OUTPUT
1
1
0.8
0.4
0.3
0.2
0.9
0.5
1
0.73
1.3
0.79
0.8
0.69
0.3
0.5
0.9
1.2
0.77
target: 0
calculated: 0.77

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

Softmax

$$p(C_k|x) = y_k = \frac{e^{a_k}}{\sum_{k'=1}^{K} e^{a_{k'}}}$$

Proces uczenia

Funkcja straty może wyglądać np. tak:

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \tilde{y}_i)^2$$

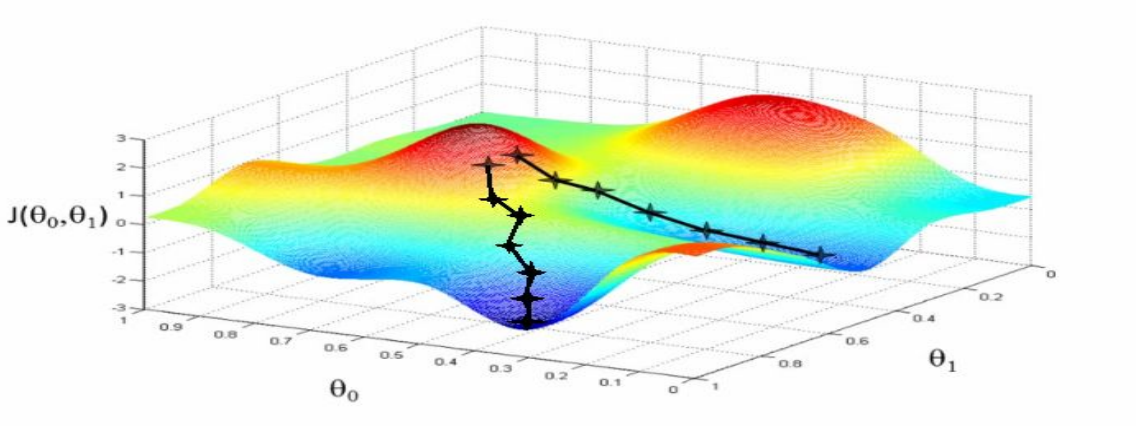
Załóżmy że F jest różniczkowalna i ciągła.

Proces uczenia cz2

Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

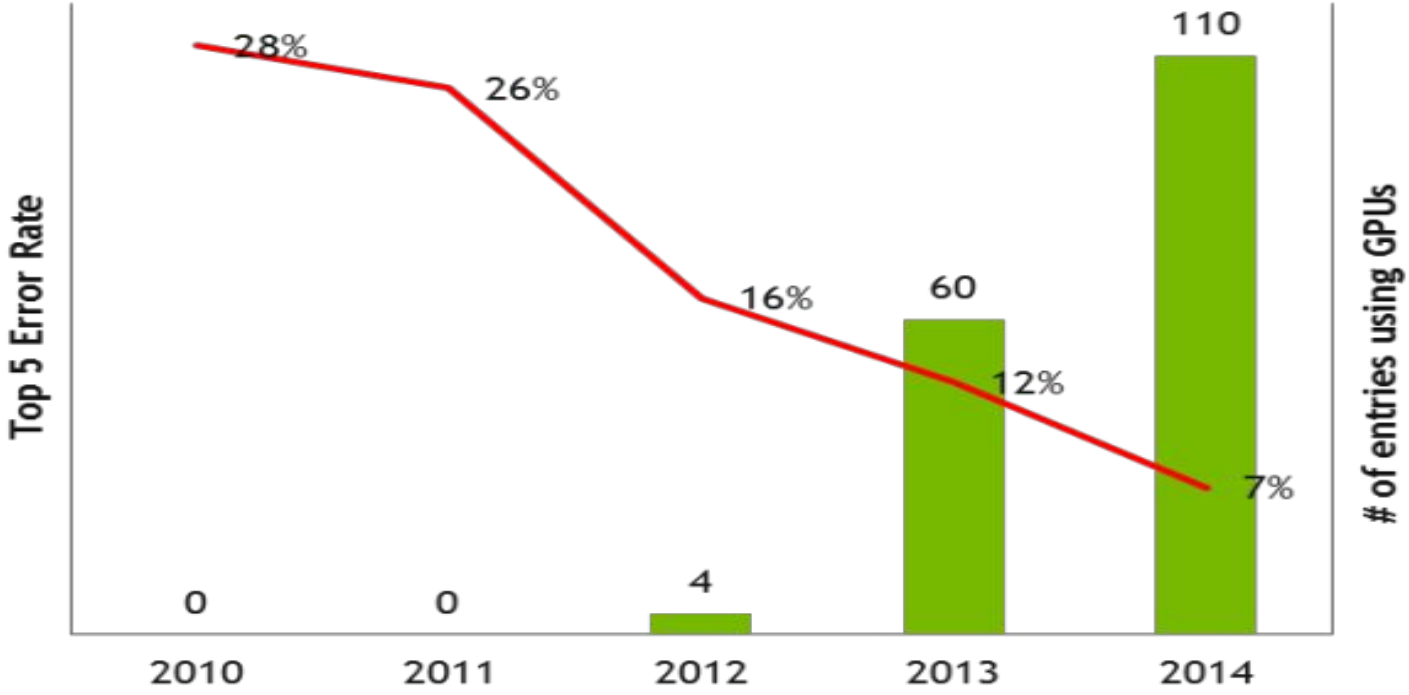
}



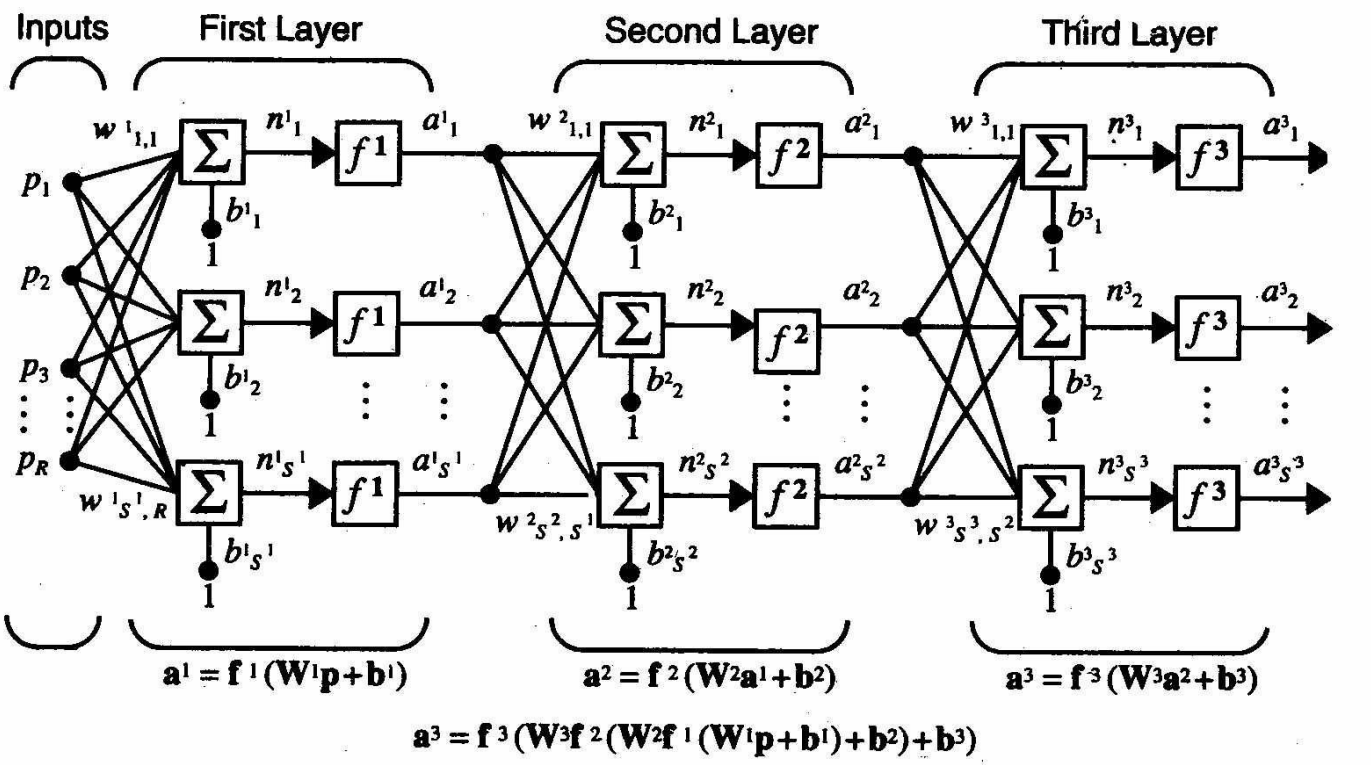
Zadanie na dziś

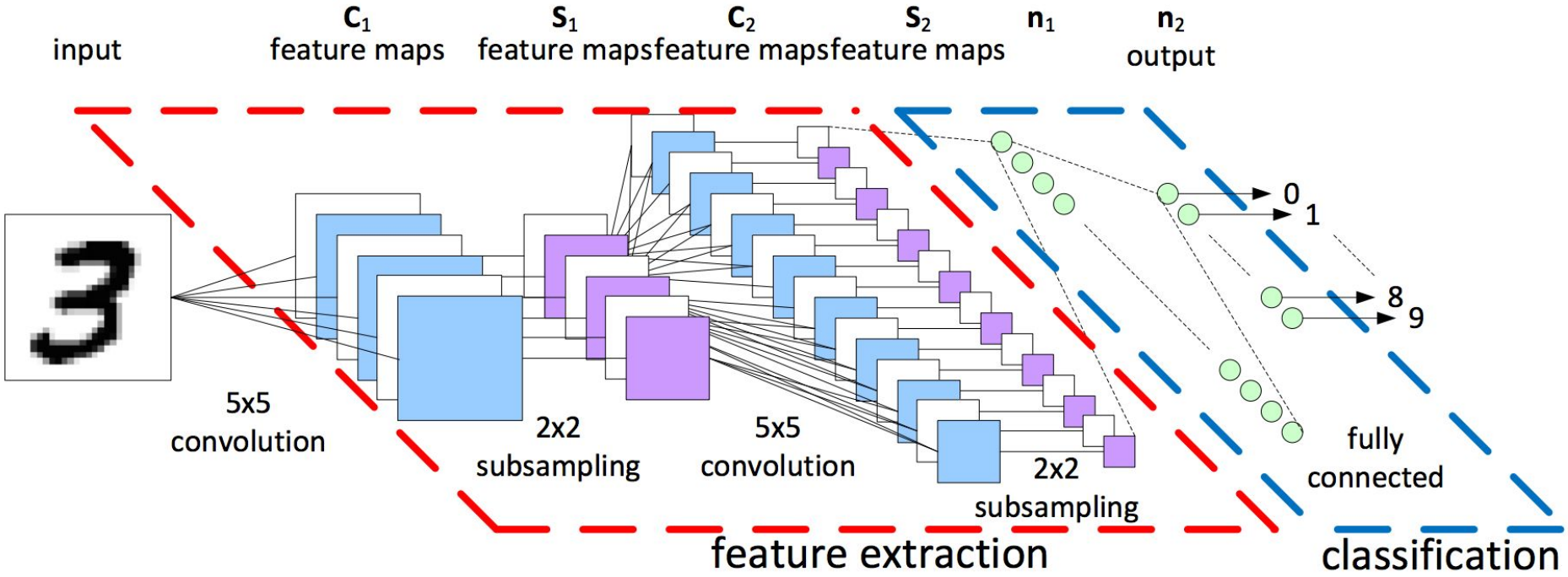
- Klasyfikacja obrazków N-klas
- Inputy: obrazki czyli macierze trójwymiarowe (width x height x 3 – bo kanały RGB)
- Outputy: wektory prawdopodobieństwa długości N

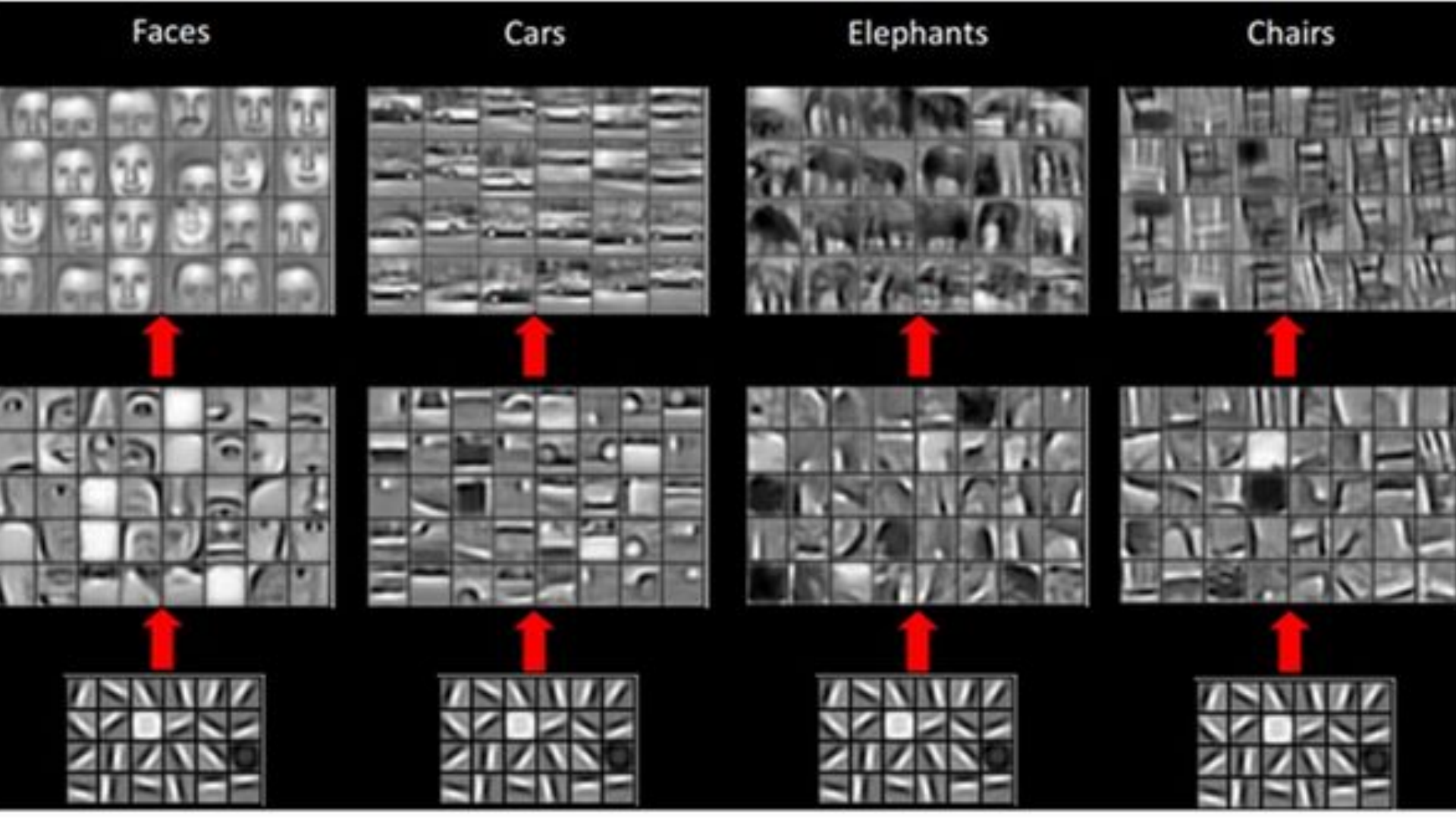
IMAGENET


Top 5 Error Rate
# of entries using GPUs
28%
26%
16%
12%
7%
0
0
4
60
110
2010
2011
2012
2013
2014

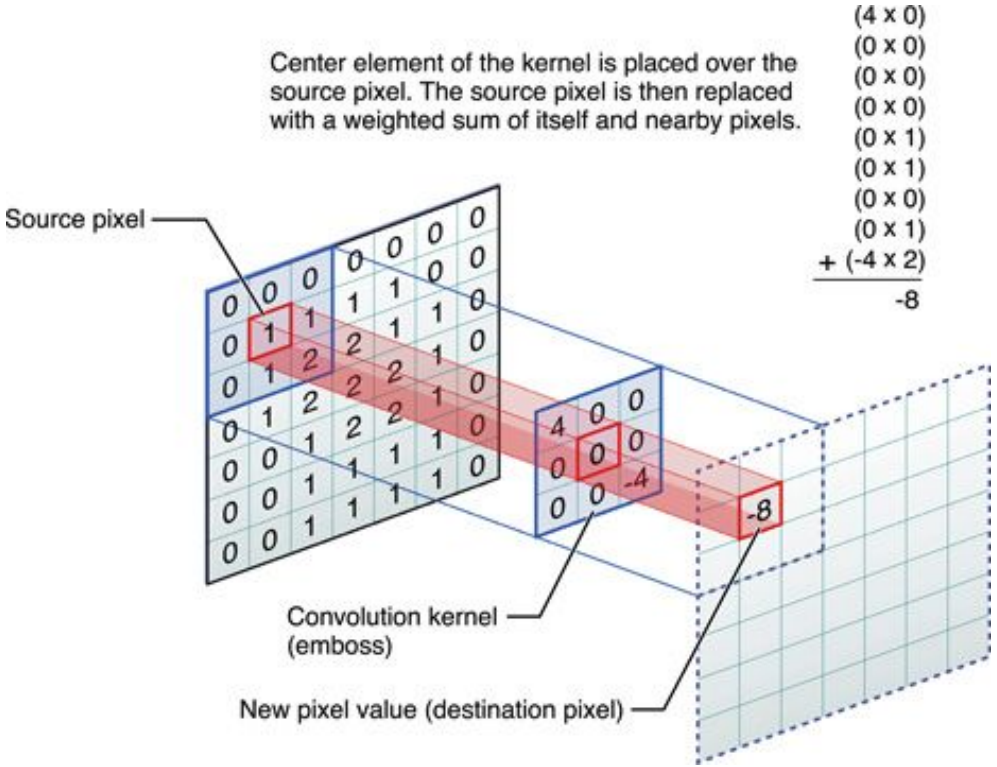
Sieć fully connected?

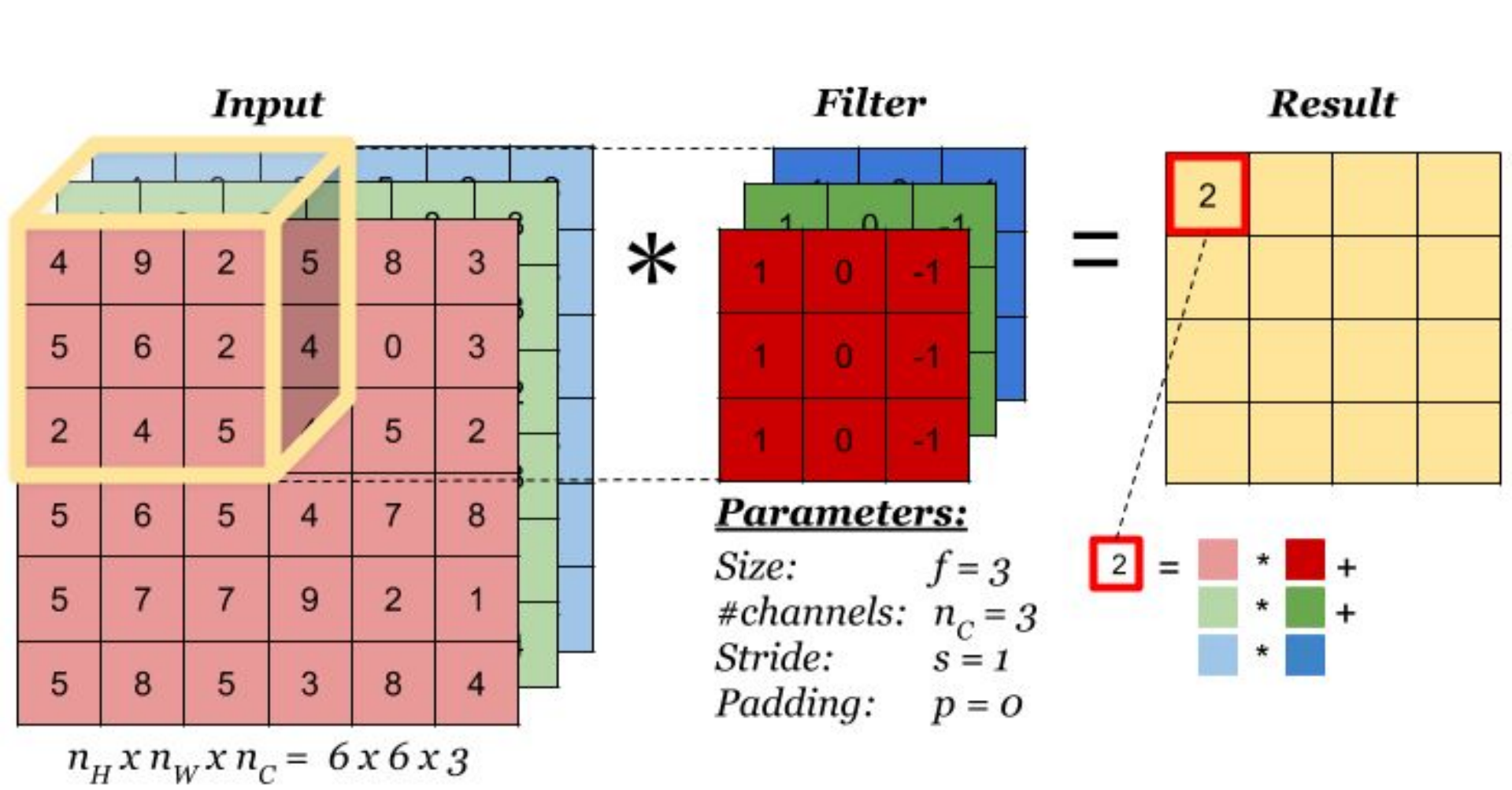


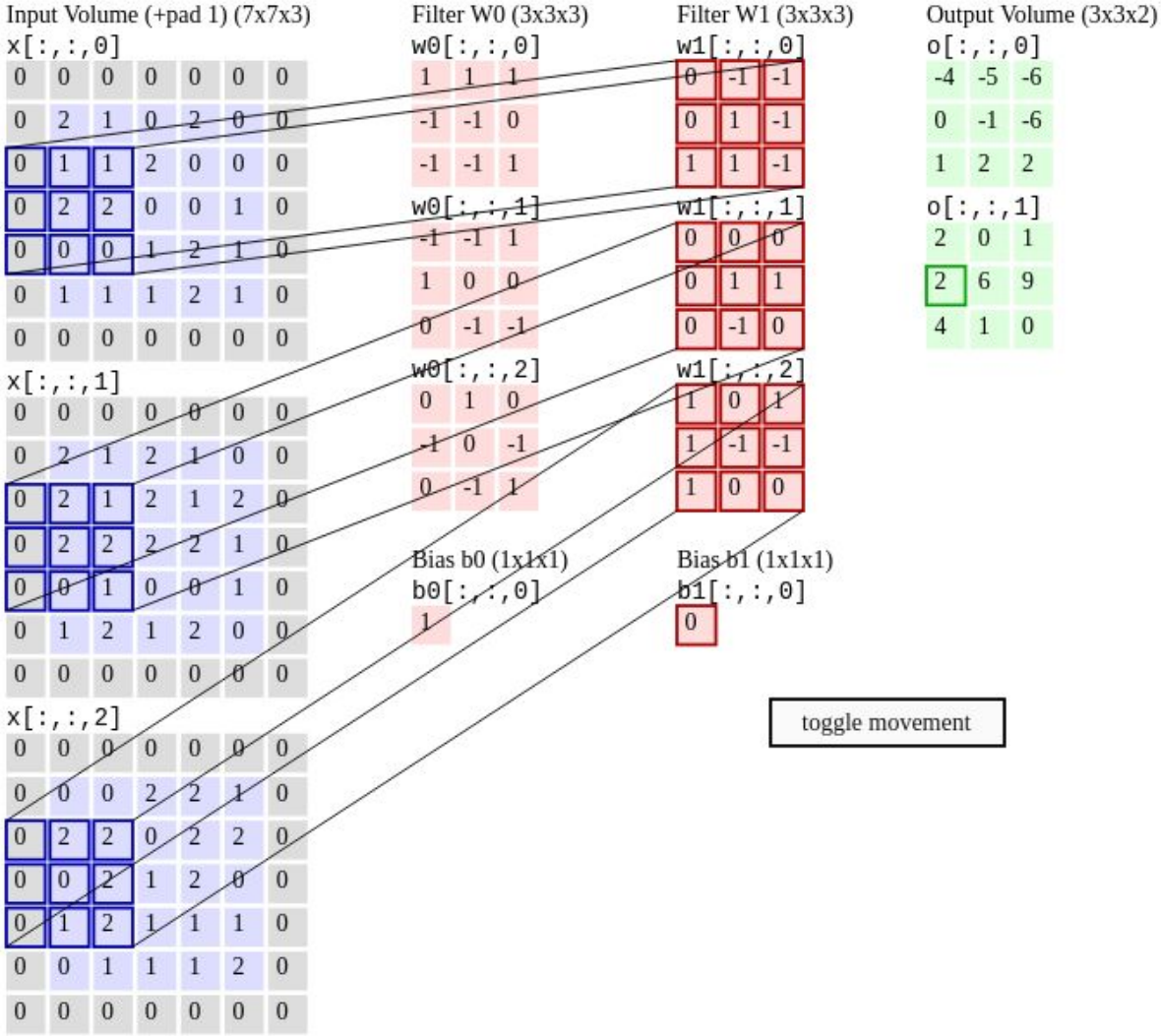

input
C1
feature maps
S1
feature maps
C2
feature maps
S2
feature maps
n1
n2
output
0
1
8
9
5x5
convolution
2x2
subsampling
5x5
convolution
2x2
subsampling
fully
connected
feature extraction
classification

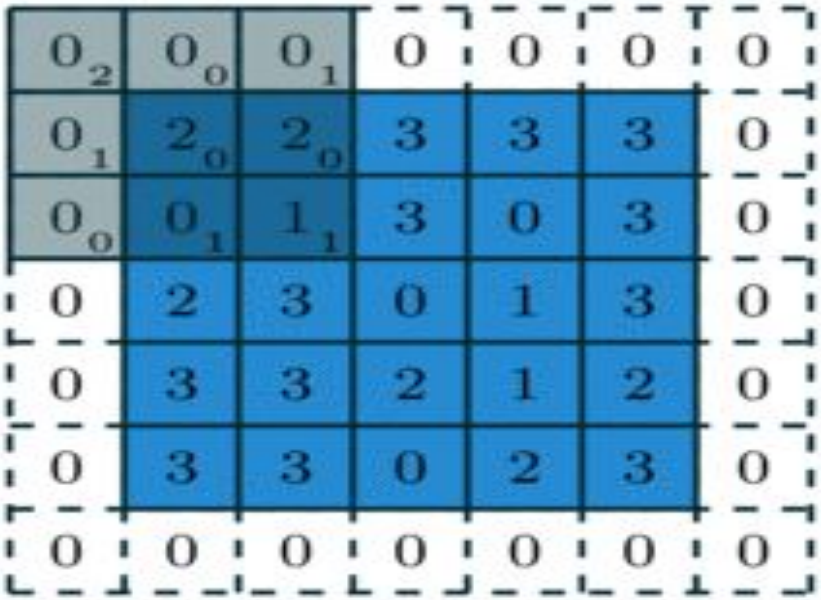

Faces
Cars
Elephants
Chairs

Jak rozwiązać te problemy? – konwolucja zamiast fully connected!



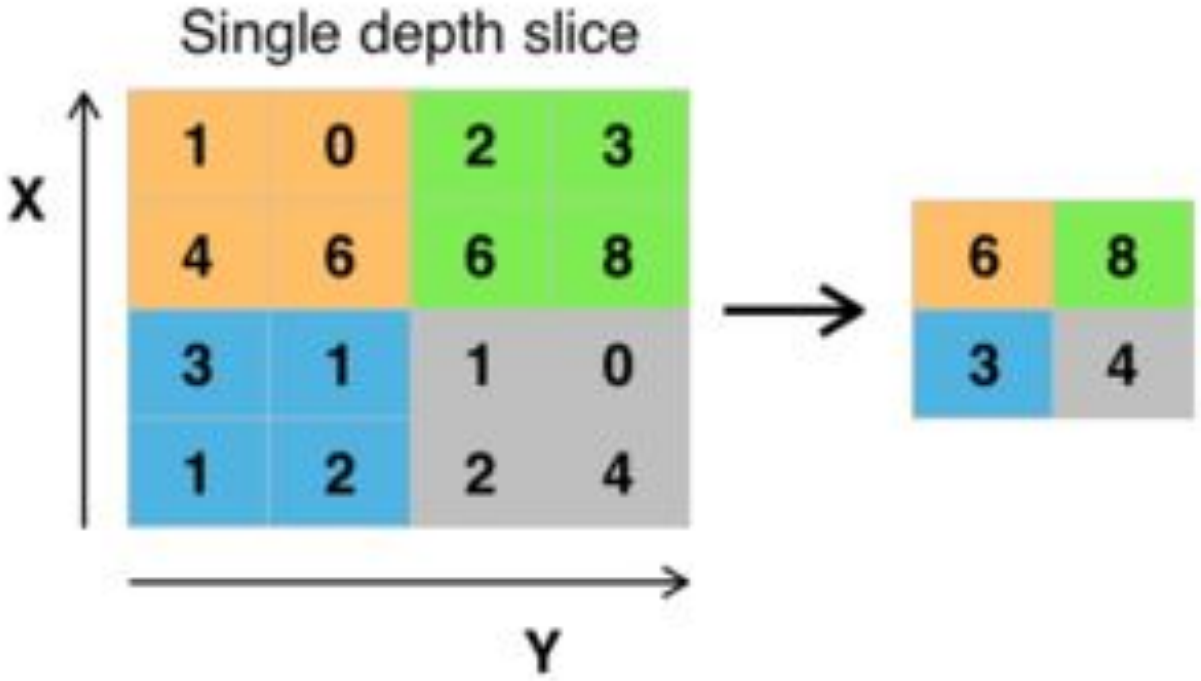

Input
Filter
Result
4 9 2 5 8 3
5 6 2 4 0 3
2 4 5 5 2
5 6 5 4 7 8
5 7 7 9 2 1
5 8 5 3 8 4
1 0 -1
1 0 -1
1 0 -1
2
Parameters:
Size: f = 3
#channels: n_C = 3
Stride: s = 1
Padding: p = 0
n_H x n_W x n_C = 6 x 6 x 3


Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 2 1 0 2 0 0
0 1 1 2 0 0 0
0 2 2 0 0 1 0
0 0 0 1 2 1 0
0 1 1 1 2 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 2 1 2 1 0 0
0 2 1 2 1 2 0
0 2 2 2 2 1 0
0 0 1 0 0 1 0
0 1 2 1 2 0 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 0 0 2 2 1 0
0 2 2 0 2 2 0
0 0 2 1 2 0 0
0 1 2 1 1 1 0
0 0 1 1 1 2 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
1 1 1
-1 -1 0
-1 -1 1
w0[:,:,1]
-1 -1 1
1 0 0
0 -1 -1
w0[:,:,2]
0 1 0
-1 0 -1
0 -1 1
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
0 -1 -1
0 1 -1
1 1 -1
w1[:,:,1]
0 0 0
0 1 1
0 -1 0
w1[:,:,2]
1 0 1
1 -1 -1
1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
-4 -5 -6
0 -1 -6
1 2 2
o[:,:,1]
2 0 1
2 6 9
4 1 0
toggle movement

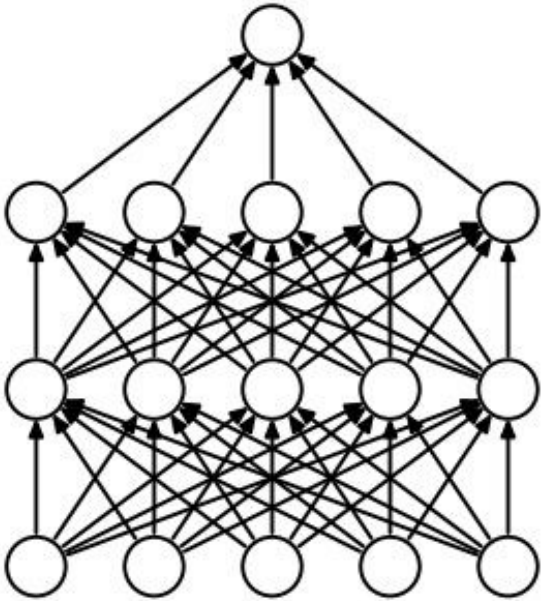


1	6	5
7	10	9
7	10	8

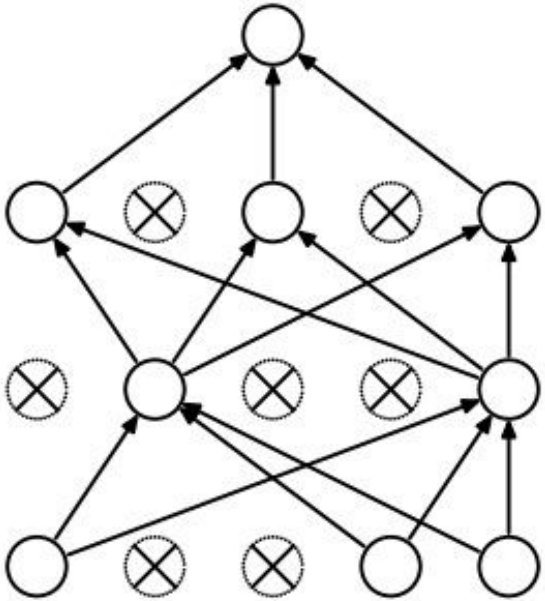
Max pooling



Dropout



(a) Standard Neural Net



(b) After applying dropout.

Obrazki ze stron

Hdimagegallery.net
Developer.apple.com
Deeplearning.net
cs231n.github.io
Aichengxu.com
Blog.christianperone.com
devblogs.nvidia.com