Using Deep Neural Networks for Discriminative Feature Localization

Javier Sánchez Rois
Daniel González Jiménez

Àgata Lapedriza García

Machine Learning Workhop Galicia 2016


Javier Sánchez Rois
Research Software Engineer
Daniel González
R&D Director, MMI Area
Gradiant
UOC
Universitat Oberta de Catalunya
SUnAI SCENE UNDERSTANDING & ARTIFICIAL INTELLIGENCE
Computer Vision & A.I Recognition for Objects and Gestures
CSAI
Àgata Lapedriza
Associate Professor (UOC)
Visiting Researcher (MIT)
Massachusetts Institute of Technology
Àgata Lapedriza
Associate Professor

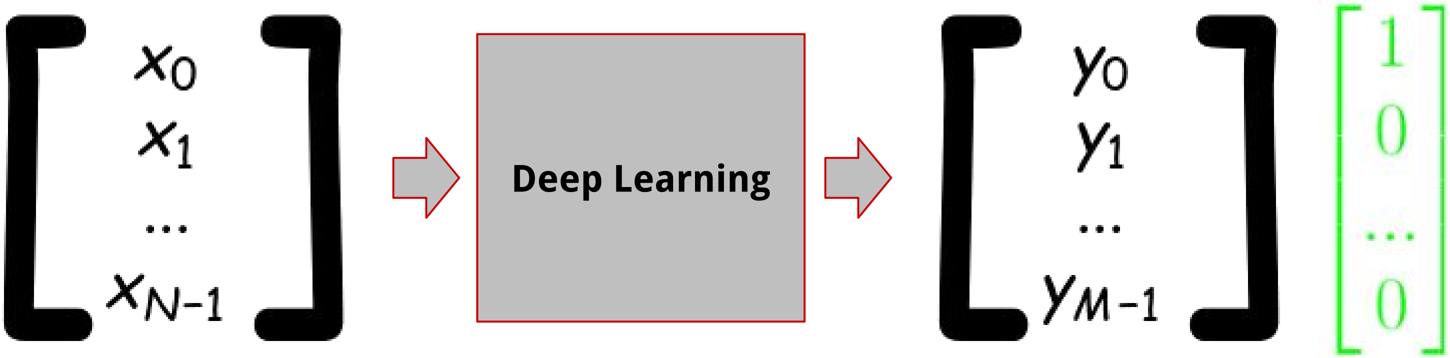
Deep {Neural Nets, Learning}

43 y.
1.70 m
Iowa
woman

$\begin{bmatrix} 43 \\ 170 \\ 29 \\ 1 \end{bmatrix} \Rightarrow$ **Deep Learning** $\Rightarrow \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

26 y.
1.68 m
Louisiana
man

$\begin{bmatrix} 26 \\ 168 \\ 18 \\ 0 \end{bmatrix} \Rightarrow$ **Deep Learning** $\Rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix}$


x_0
x_1
...
x_{N-1}
Deep Learning
y_0
y_1
...
y_{M-1}
1
0
...
0

$$\begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{N-1} \end{bmatrix} \rightarrow$$

$$\rightarrow \begin{bmatrix} y_0 \\ y_1 \\ \dots \\ y_{M-1} \end{bmatrix}$$

A Linear Model

$$\mathbf{W} \cdot \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{N-1} \end{bmatrix} + \mathbf{b} = \begin{bmatrix} y_0 \\ y_1 \\ \dots \\ y_{M-1} \end{bmatrix}$$

$$\mathbf{W} \cdot \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{N-1} \end{bmatrix} + \mathbf{b} = \begin{bmatrix} 13.2 \\ -1.3 \\ \dots \\ 0.01 \end{bmatrix}$$

(raw scores)

$$Softmax\left(\mathbf{W}\cdot\begin{bmatrix}x_0\\x_1\\\dots\\x_{N-1}\end{bmatrix}+\mathbf{b}\right)=\begin{bmatrix}1\\0\\\dots\\0\end{bmatrix}$$

(one-hot encoding)

$$Softmax\left(\mathbf{W}\cdot\begin{bmatrix}x_0\\x_1\\\dots\\x_{N-1}\end{bmatrix}+\mathbf{b}\right)=\begin{bmatrix}0.9\\\sim 0\\\dots\\\sim 0\end{bmatrix}$$

(... almost)

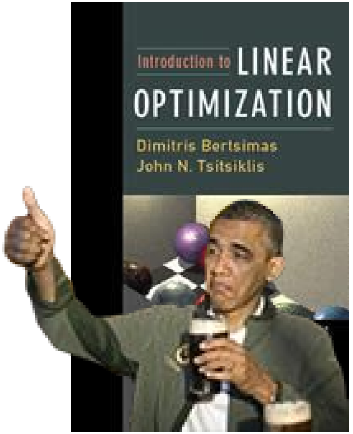
Logistic Regression

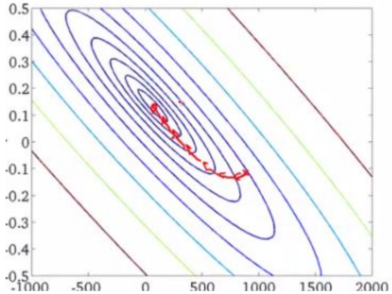
$$Softmax\left(\mathbf{W}\cdot\begin{bmatrix}x_0\\ x_1\\ \dots\\ x_{N-1}\end{bmatrix}+\mathbf{b}\right)$$

$$Error_i = f(\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.3 \\ 0.6 \\ 0.05 \\ 0.05 \end{bmatrix})$$

$$E_i = CE(\mathbf{l}_i, Softmax\left(\mathbf{W} \cdot \mathbf{x}_i + \mathbf{b}\right))$$

$$\min_{W,b} \sum_i E_i \quad ?$$

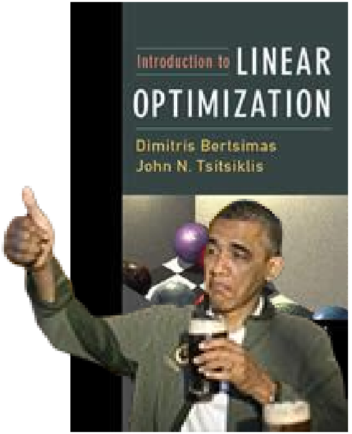




Gradient Descent

$$W_{i+1} \leftarrow W_i - \alpha \frac{\partial E}{\partial W}$$

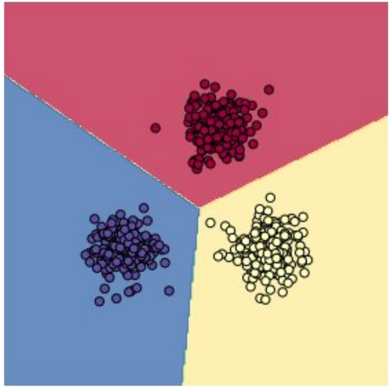
Linear Models: Pros



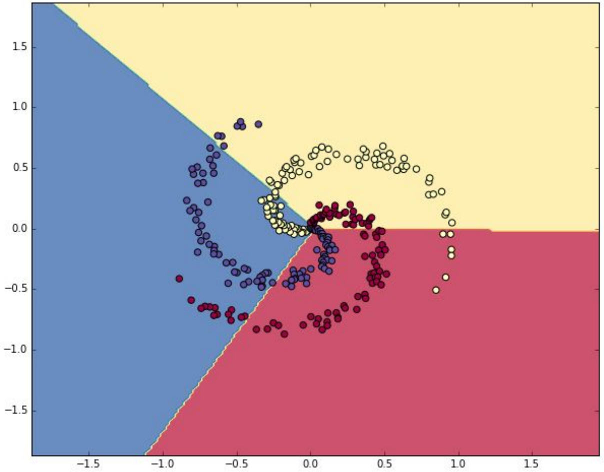
Easy to solve for minimal loss

GPUs for optimal implementation

Linear Models: Cons

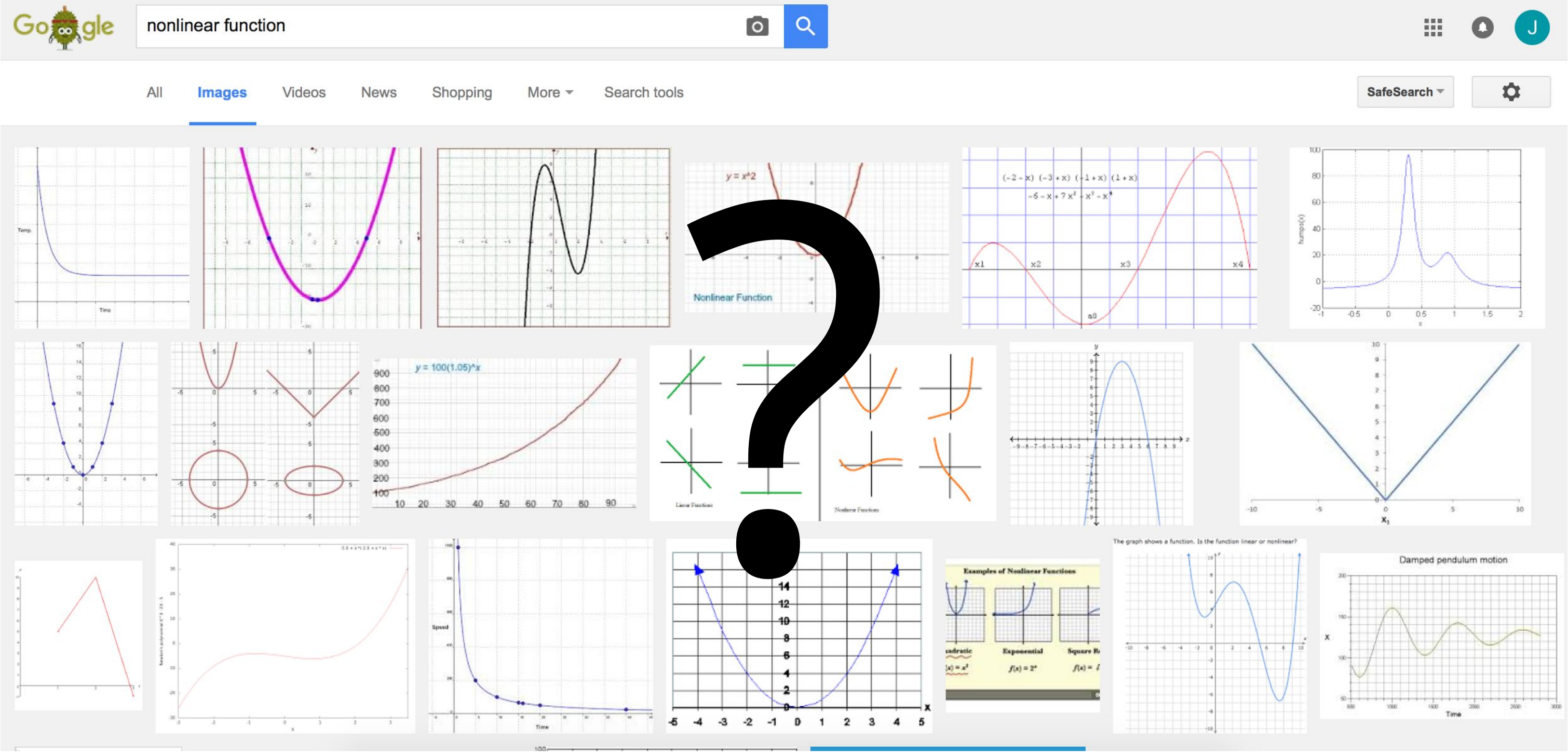


OK for *easy* scenarios

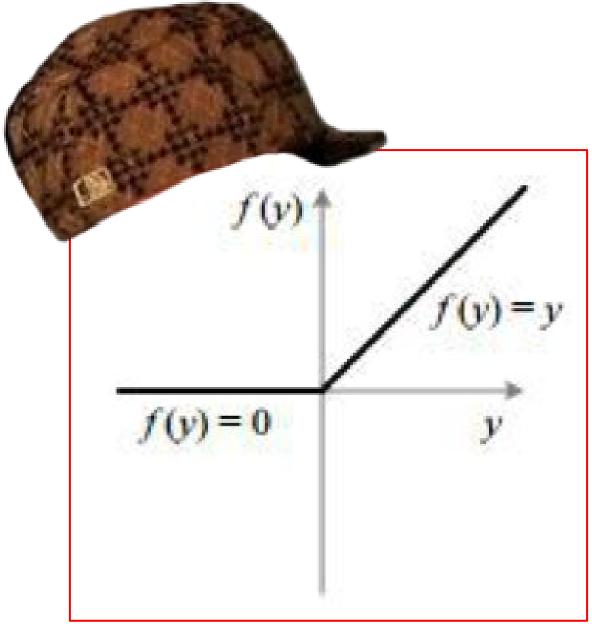


Underfitting complex data

We need to go *nonlinear*


Google
nonlinear function
All
Images
Videos
News
Shopping
More
Search tools
SafeSearch
y = x^2
Nonlinear Function
y = 100(1.05)^x
Damped pendulum motion
Examples of Nonlinear Functions
The graph shows a function. Is the function linear or nonlinear?
?

RELU: rectified linear unit

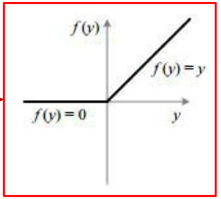


(scumbag nonlinear function)

The *perceptron*

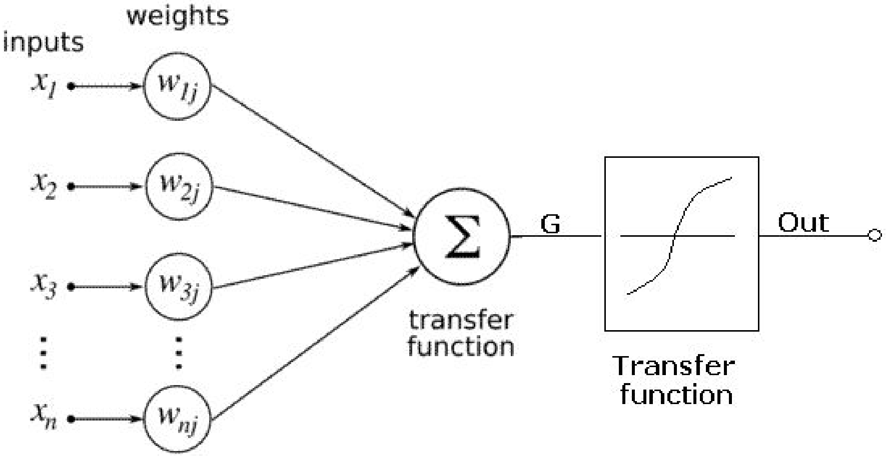
$$\mathbf{W}_0 \cdot [\bullet] + \mathbf{b}_0$$

Fully-connected *layer*

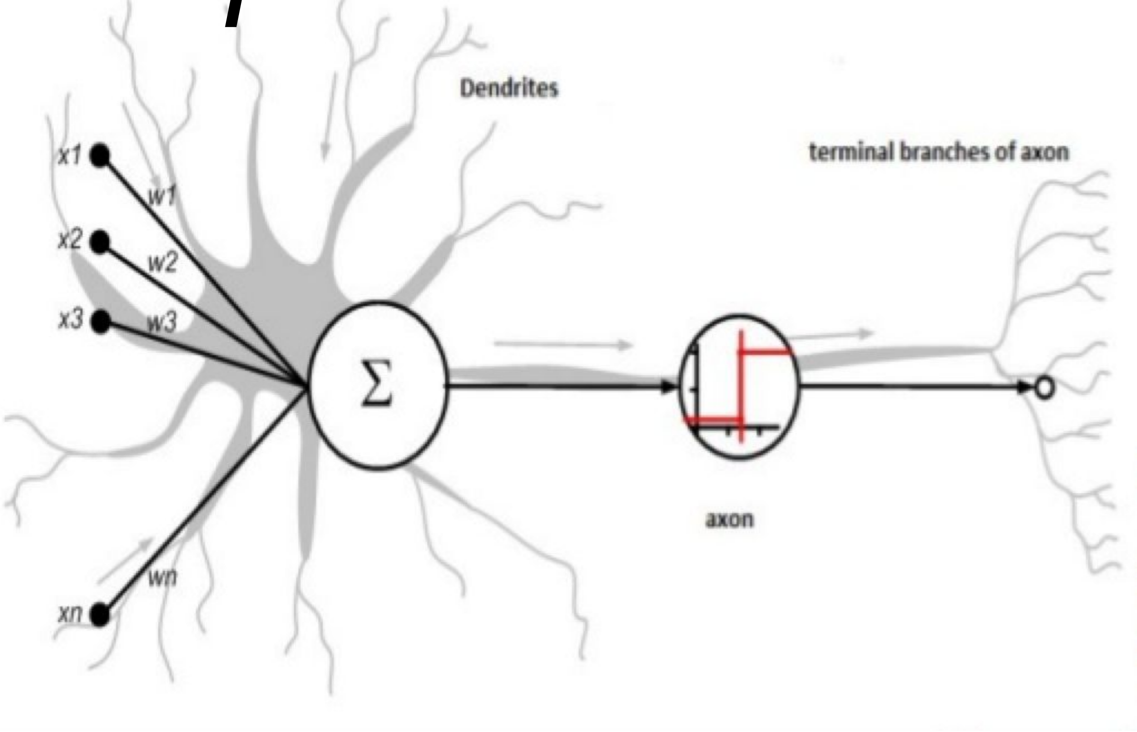


RELU

The *perceptron*

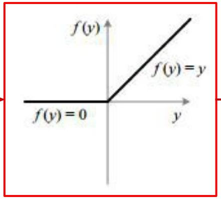


The *perceptron*

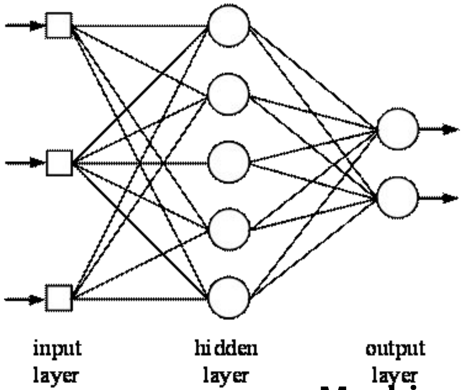


The *multilayer perceptron*

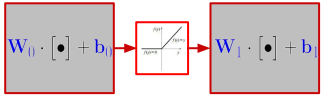
$$\mathbf{W}_0 \cdot [\bullet] + \mathbf{b}_0$$



$$\mathbf{W}_1 \cdot [\bullet] + \mathbf{b}_1$$



$$\min_{W_0,b_0,W_1,b_1} \sum_i E_i$$



too hard

Gradient Descent

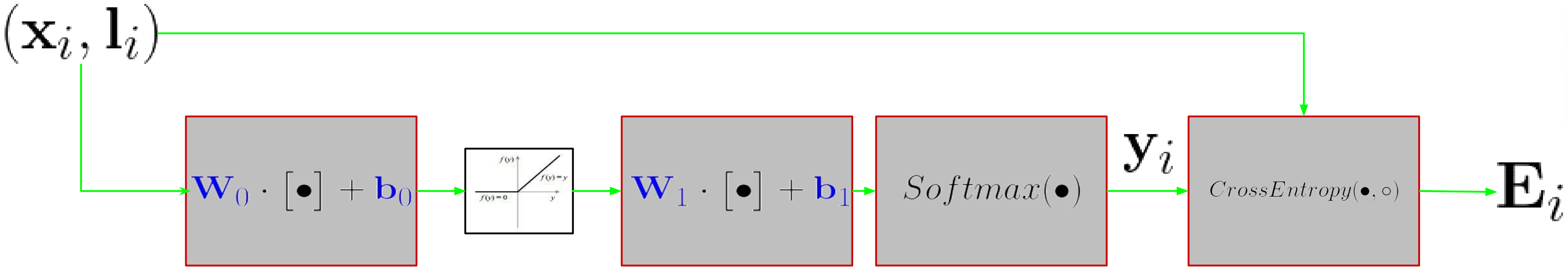
$$W_{i+1} \leftarrow W_i - \alpha \frac{\partial E}{\partial W}$$

Backpropagation*

Backpropagation = Rule of Chain

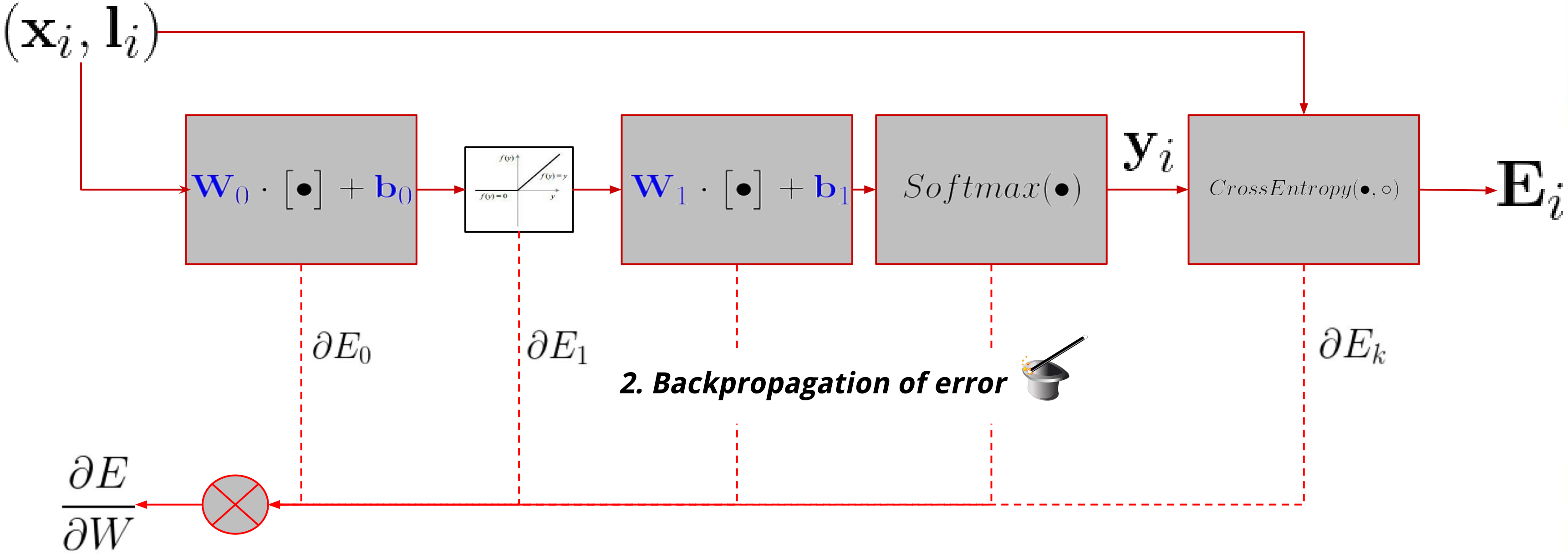
$$\frac{\partial}{\partial W} Error\left(\boxed{\mathbf{W}_0 \cdot [\bullet] + \mathbf{b}_0} \rightarrow \boxed{} \rightarrow \boxed{\mathbf{W}_1 \cdot [\bullet] + \mathbf{b}_1} \right) = \partial E_0 \cdot \partial E_1 \ldots \cdot \partial E_k$$

Training

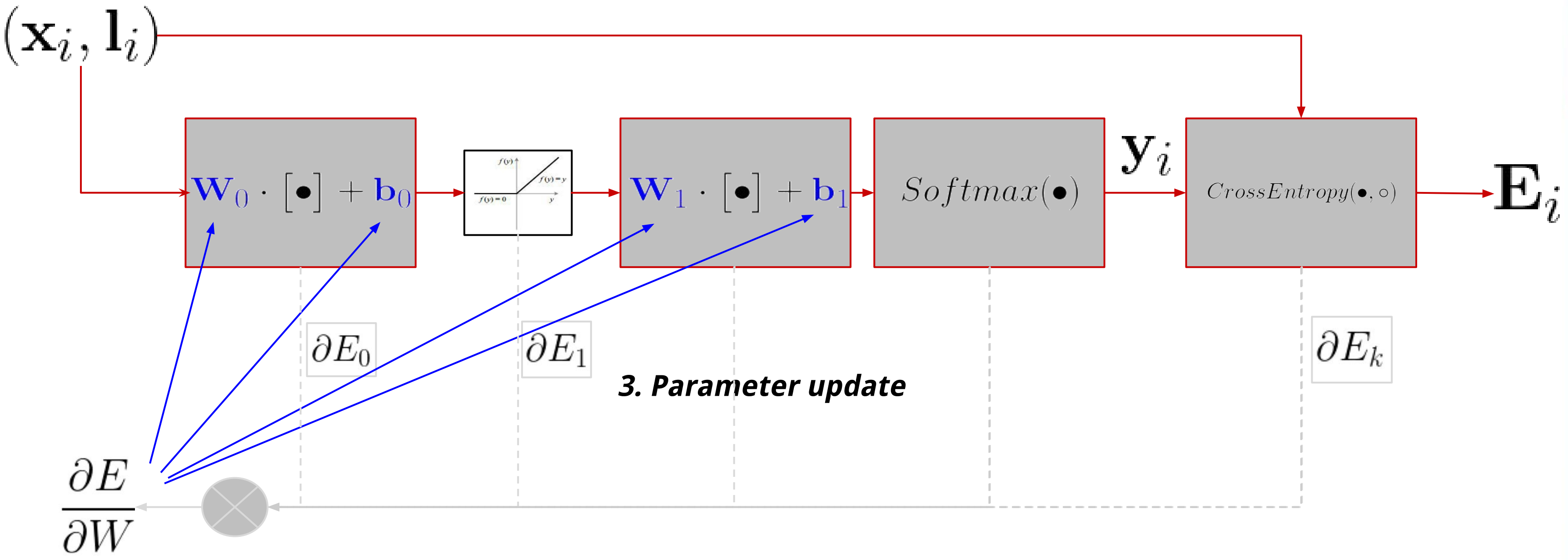


1. ***Forward Propagation***

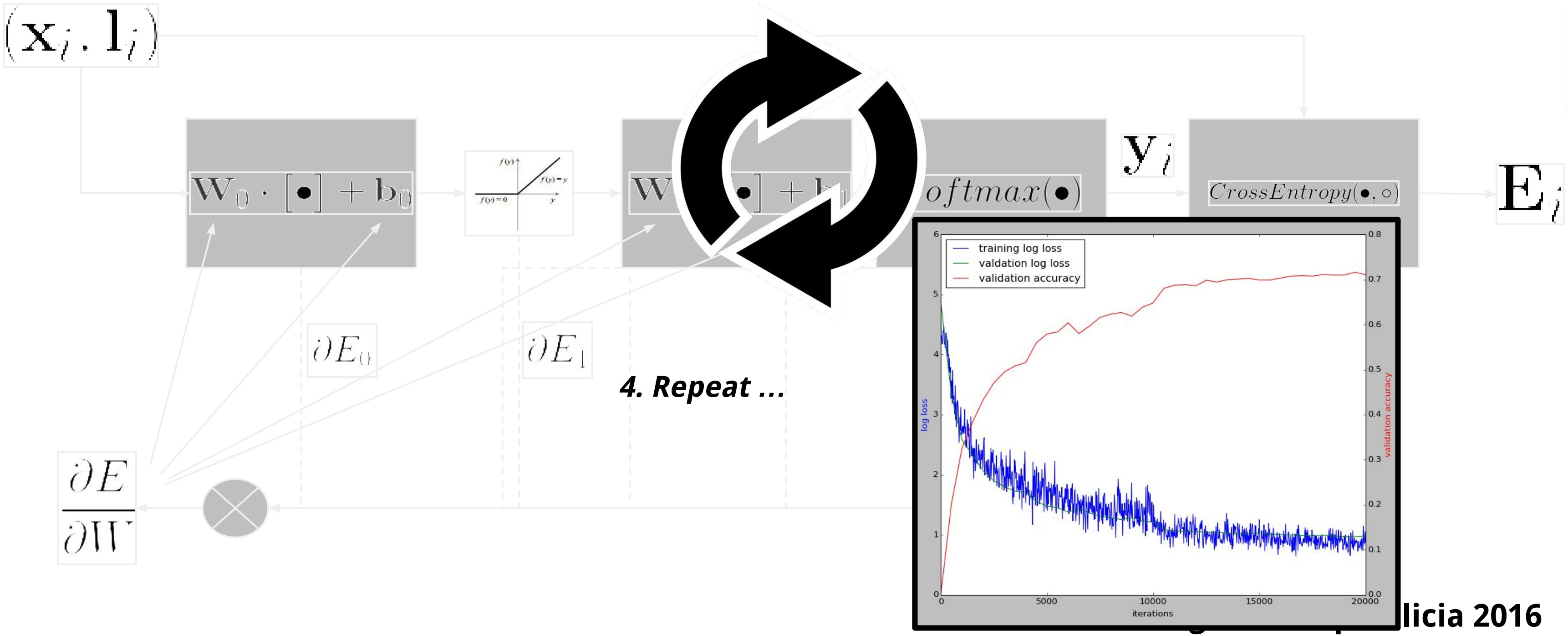
Training

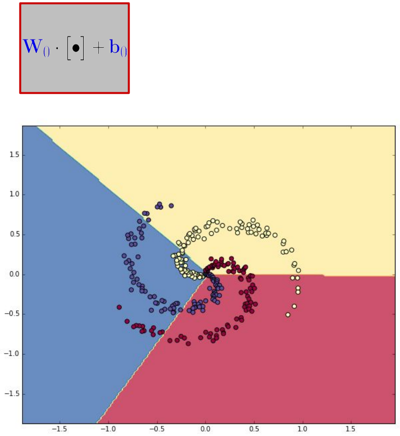

$(\mathbf{x}_i, \mathbf{l}_i)$
$\mathbf{W}_0 \cdot [\bullet] + \mathbf{b}_0$
$f(y)$
$f(y) = y$
$f(y) = 0$
y
$\mathbf{W}_1 \cdot [\bullet] + \mathbf{b}_1$
$Softmax(\bullet)$
$\mathbf{y}_i$
$CrossEntropy(\bullet, \circ)$
$\mathbf{E}_i$
∂E_0
∂E_1
∂E_k
2. Backpropagation of error
$\frac{\partial E}{\partial W}$

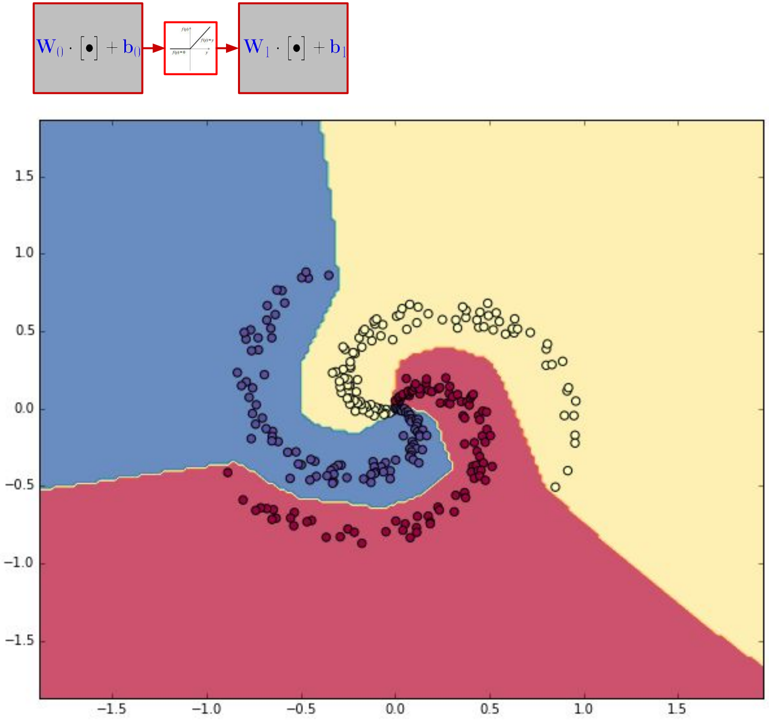
Training



$(\mathbf{x}_i, \mathbf{l}_i)$
$\mathbf{W}_0 \cdot [\bullet] + \mathbf{b}_0$
$\mathbf{W}_1 \cdot [\bullet] + \mathbf{b}_1$
$Softmax(\bullet)$
$\mathbf{y}_i$
$CrossEntropy(\bullet, \circ)$
$\mathbf{E}_i$
∂E_0
∂E_1
∂E_k
3. Parameter update
$\frac{\partial E}{\partial W}$


$(\mathbf{x}_i, \mathbf{l}_i)$
$\mathbf{W}_0 \cdot [\bullet] + \mathbf{b}_0$
$Softmax(\bullet)$
$\mathbf{y}_i$
$CrossEntropy(\bullet, \circ)$
$\mathbf{E}_i$
∂E_0
∂E_1
$\frac{\partial E}{\partial W}$
4. Repeat ...
training log loss
valdation log loss
validation accuracy
log loss
validation accuracy
iterations





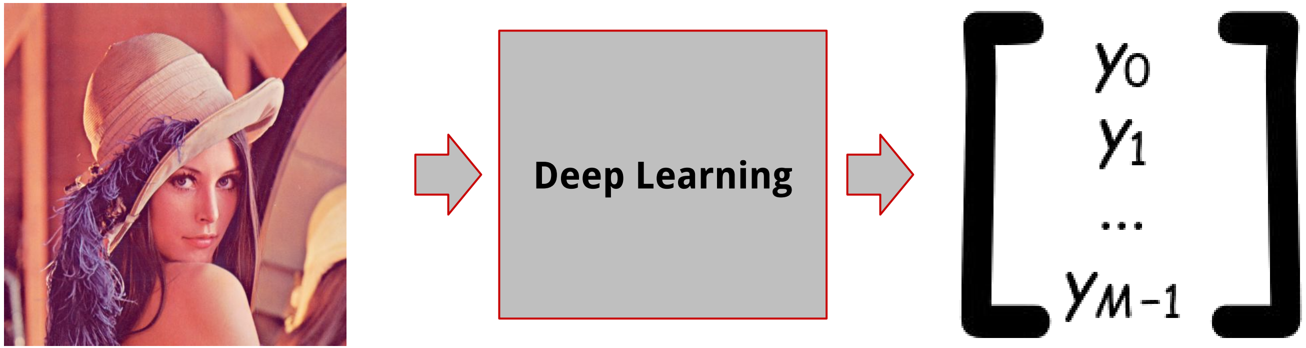
Andrej Karpathy - CS231n: Convolutional Neural Networks for Visual Recognition.

Machine Learning Workhop Galicia 2016

Neural Nets are machine learning network characterized by

- Use of **nonlinear** models
- **Backpropagation** algorithm for training

DNN’ing images



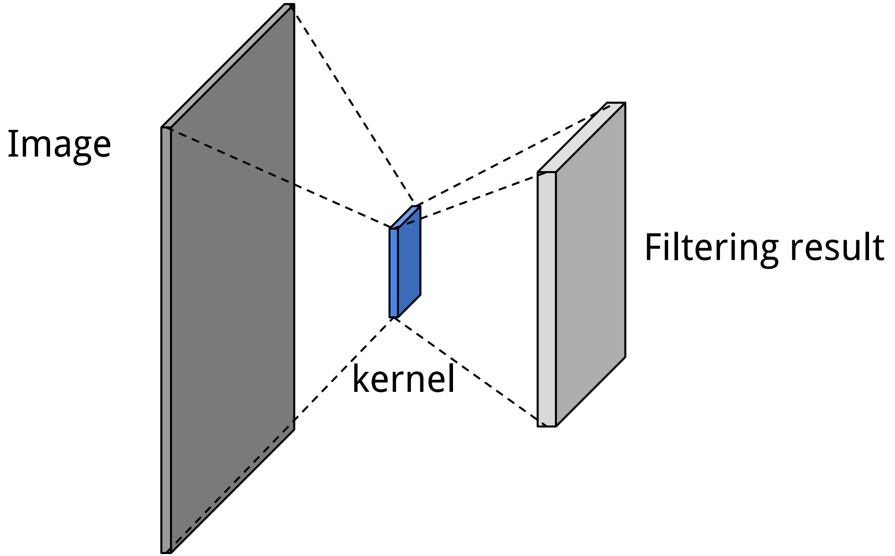
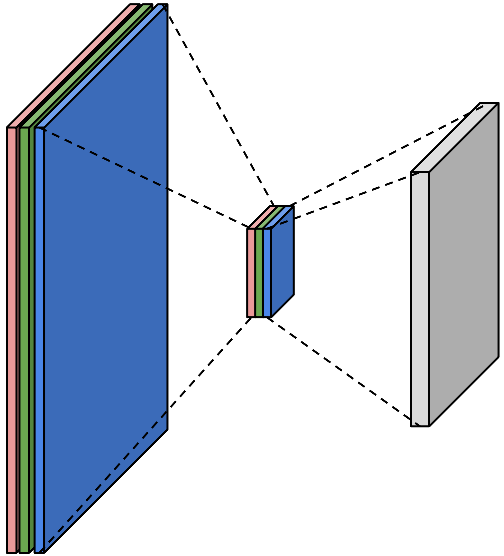
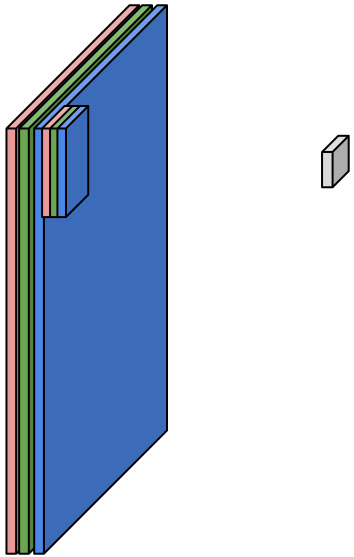
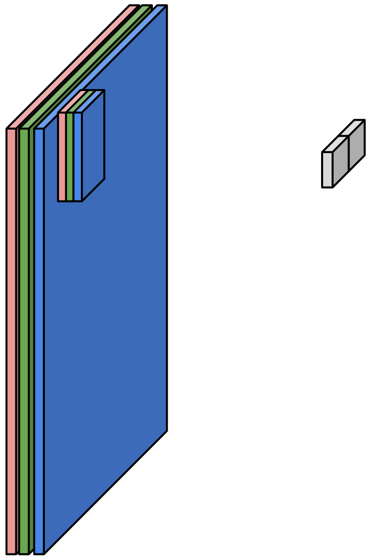
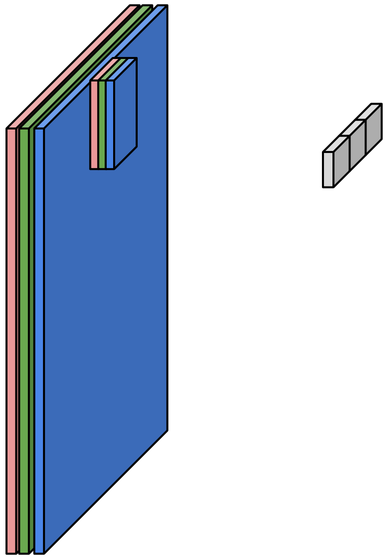


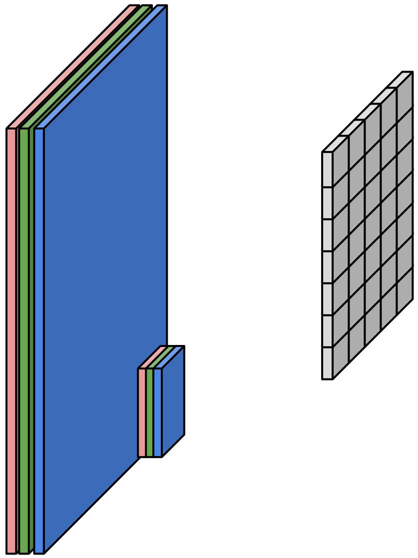
Image
kernel
Filtering result

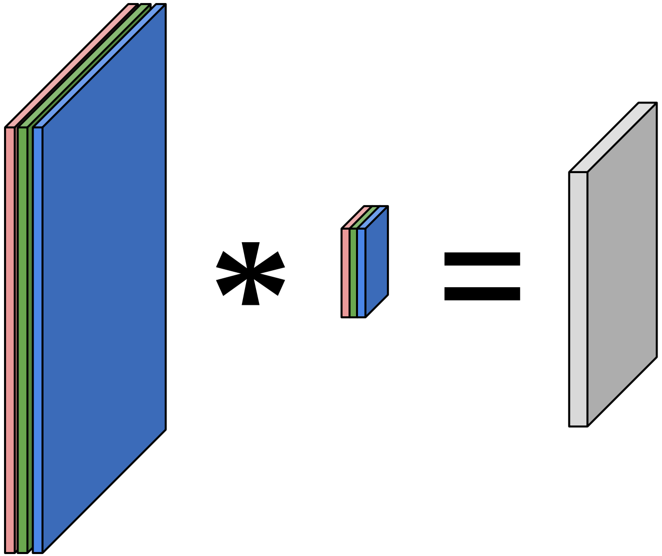


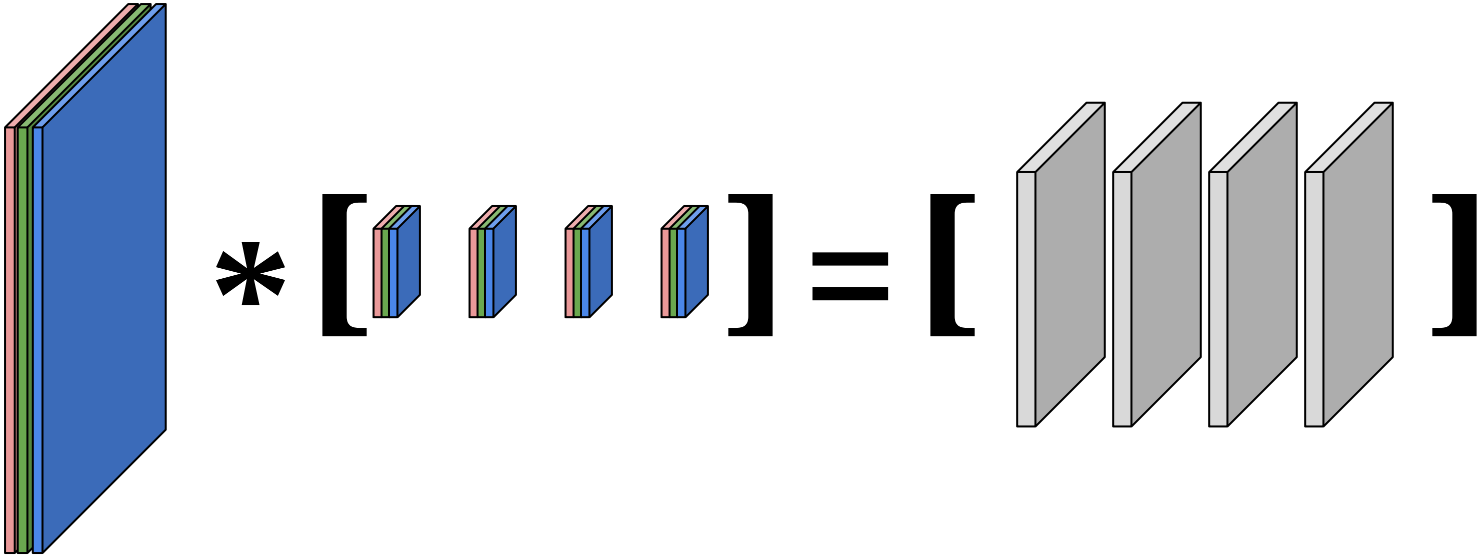


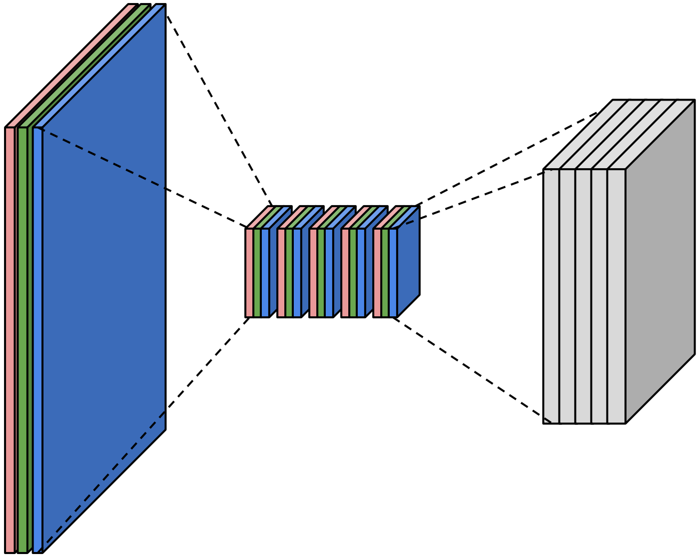


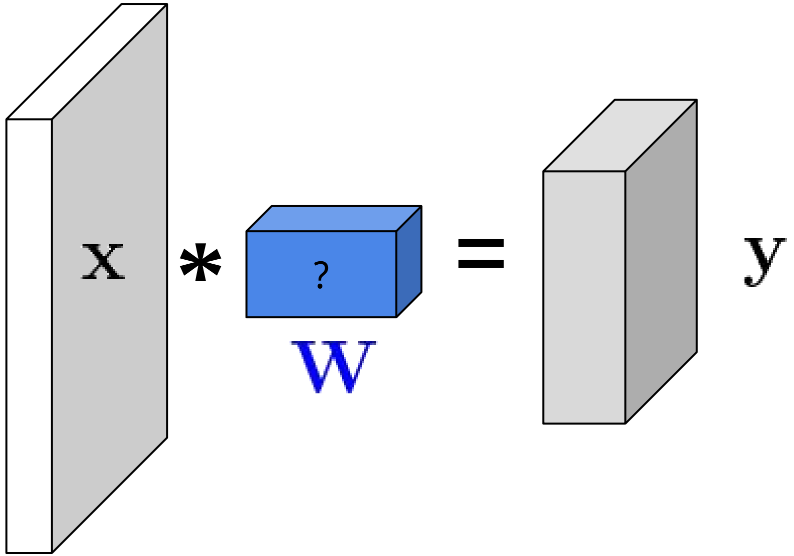


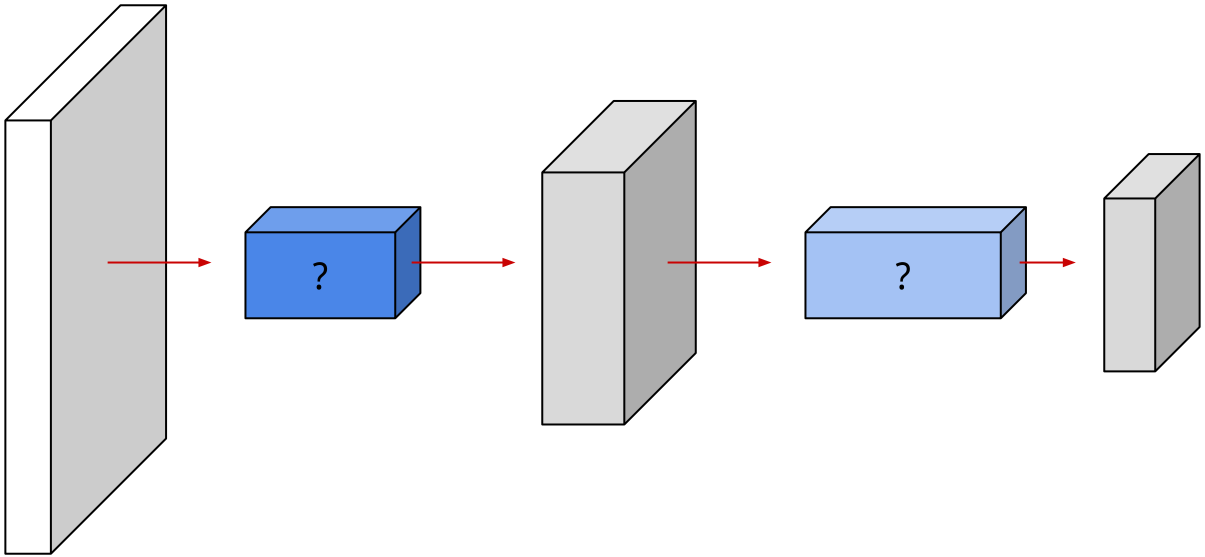


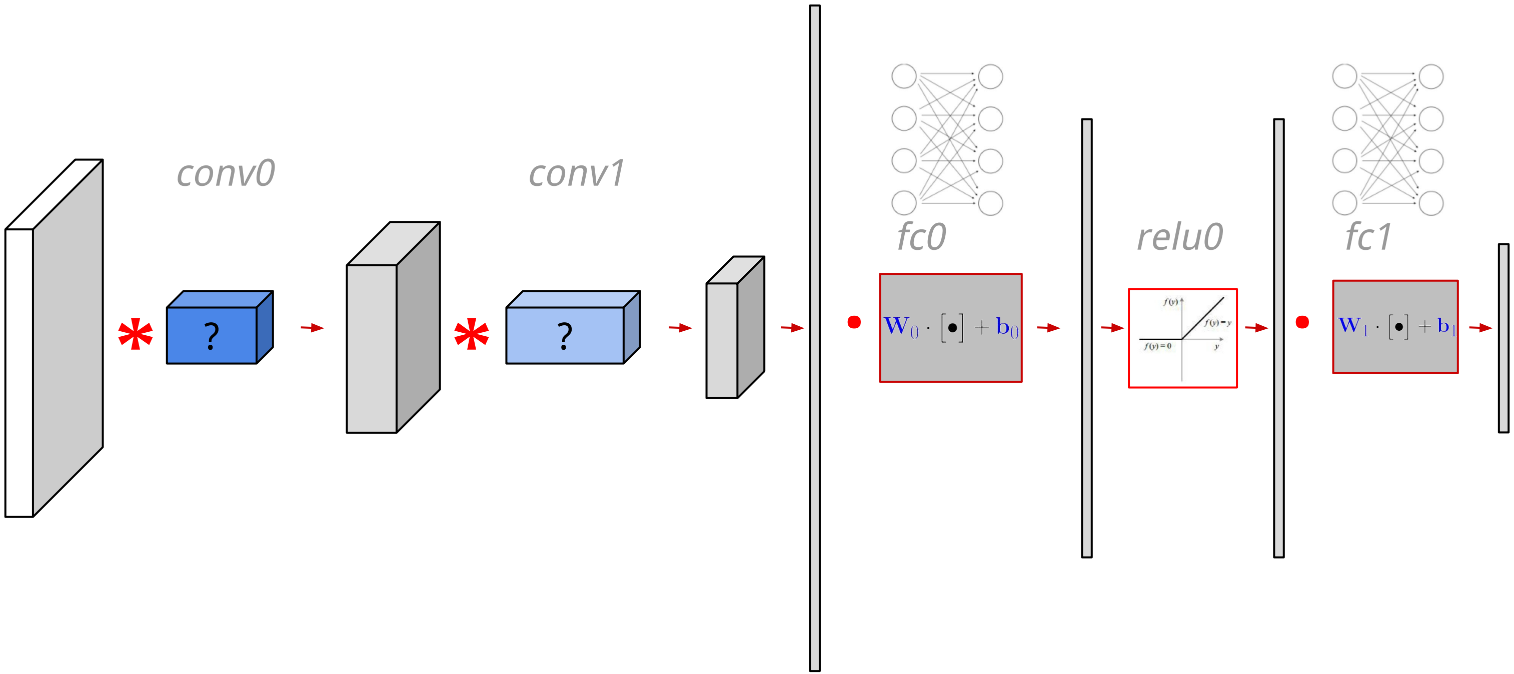







x
*
?
W
=
y




conv0
?
conv1
?
fc0
$W_0 \cdot [\bullet] + b_0$
relu0
fc1
$W_1 \cdot [\bullet] + b_1$

Network Architecture

How many convolutional layers?
How many fully connected layers?
Parameters?
...

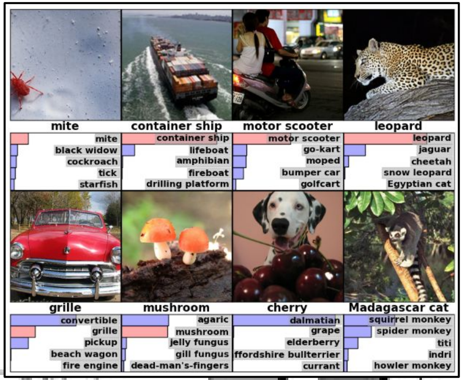
CNN architectures: AlexNet (2012)

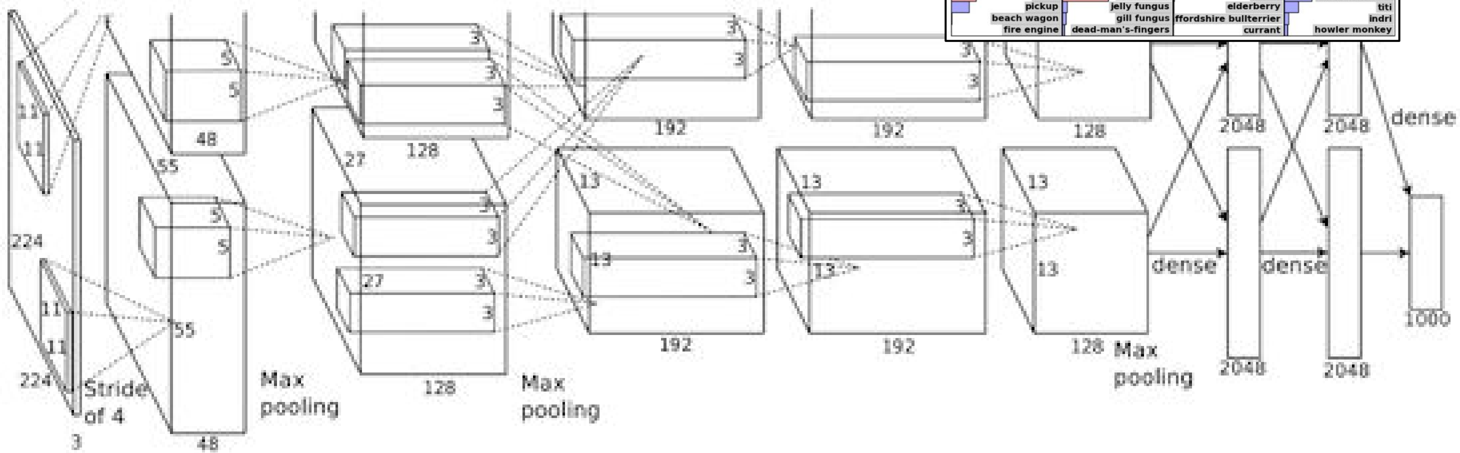
ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

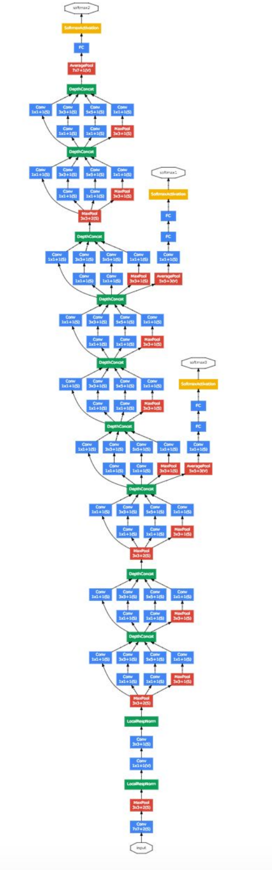
Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca



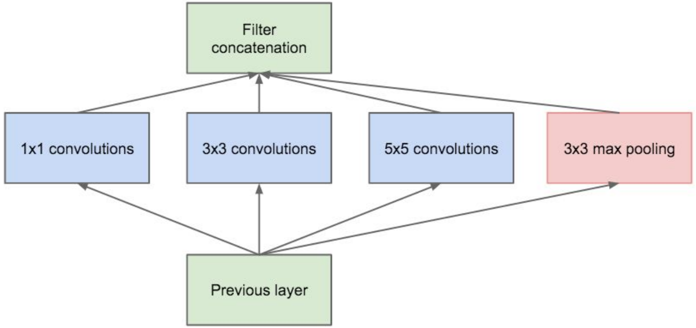


CNN architectures: GoogLeNet (2014)



Going Deeper with Convolutions

Christian Szegedy[1], Wei Liu[2], Yangqing Jia[1], Pierre Sermanet[1], Scott Reed[3],
Dragomir Anguelov[1], Dumitru Erhan[1], Vincent Vanhoucke[1], Andrew Rabinovich[4]
[1]Google Inc. [2]University of North Carolina, Chapel Hill
[3]University of Michigan, Ann Arbor [4]Magic Leap Inc.
[1]{szegedy,jiayq,sermanet,dragomir,dumitru,vanhoucke}@google.com
[2]wliu@cs.unc.edu, [3]reedscott@umich.edu, [4]arabinovich@magicleap.com



Caffe

So what is Caffe?

- Pure C++ / CUDA architecture for deep learning
    - command line, Python, MATLAB interfaces
- Fast, well-tested code
- Tools, reference models, demos, and recipes
- Seamless switch between CPU and GPU
    - `Caffe::set_mode(Caffe::GPU);`

Prototype

Training

Deployment

All with essentially the same code!

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  param {
    lr_mult: 1
    decay_mult: 1
  }
  param {
    lr_mult: 2
    decay_mult: 0
  }
  convolution_param {
    num_output: 96
    kernel_size: 11
    stride: 4
    weight_filler {
      type: "gaussian"
      std: 0.01
    }
    bias_filler {
      type: "constant"
      value: 0
    }
  }
}
layer {
  name: "relu1"
```

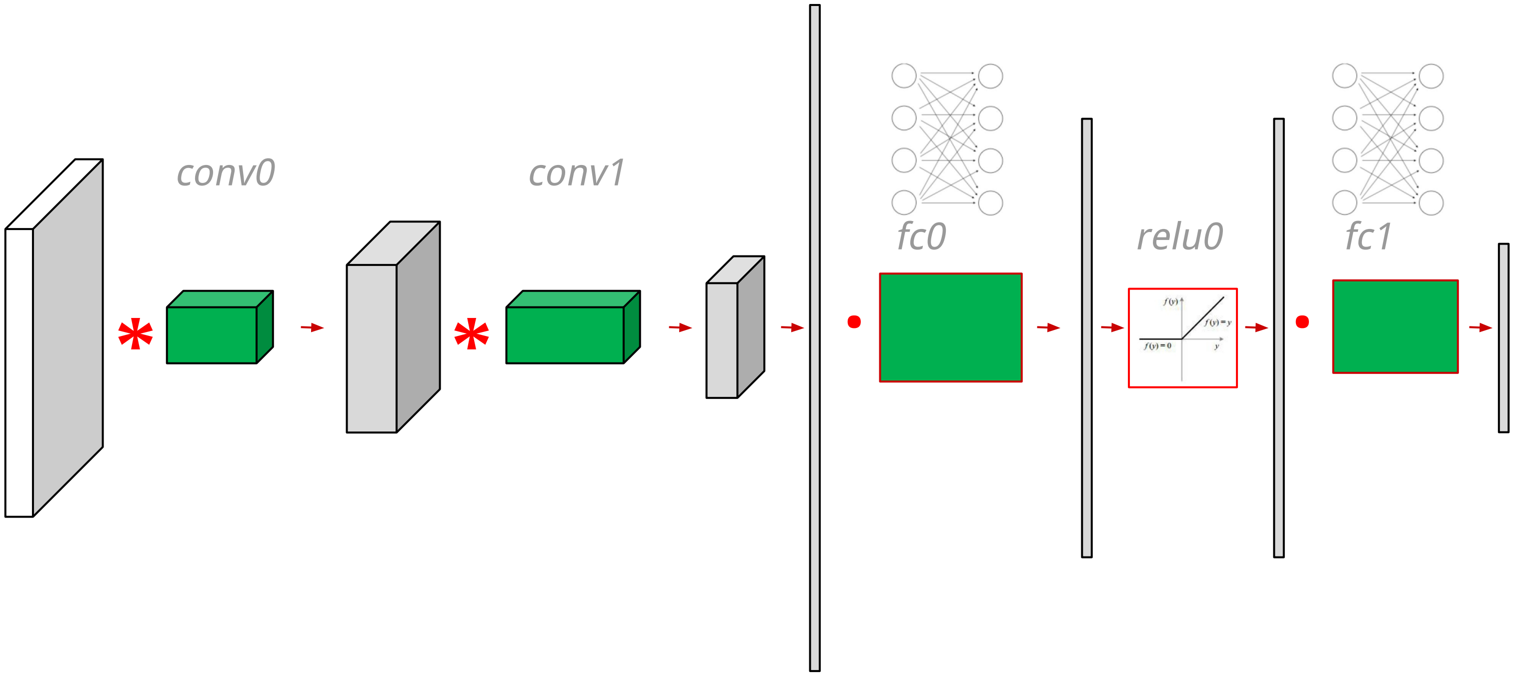
Caffe
Model Zoo

Discriminative Feature Localization

Fine-tuning

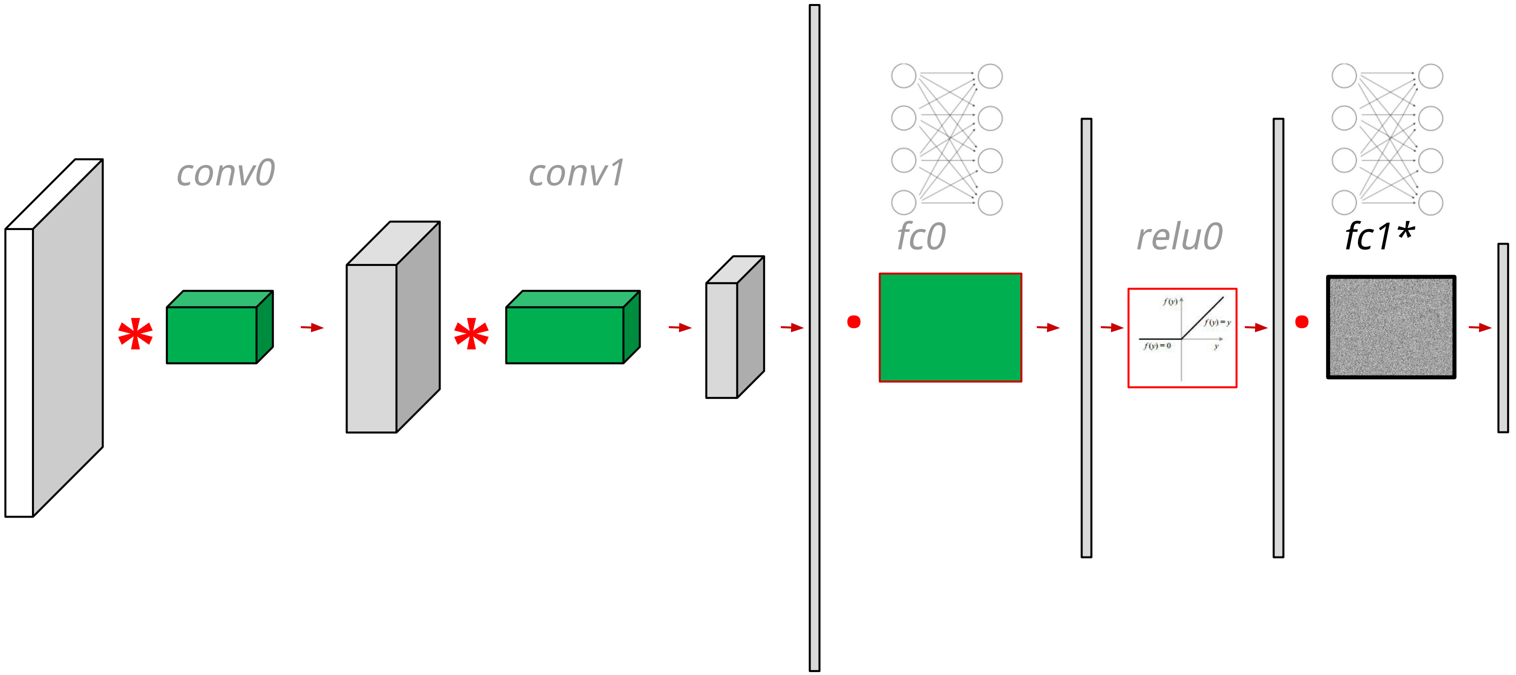
Adjusting the parameters of an existing, pre-trained network to solve a different problem with “*few*” images from the new problem domain.

Fine-tuning



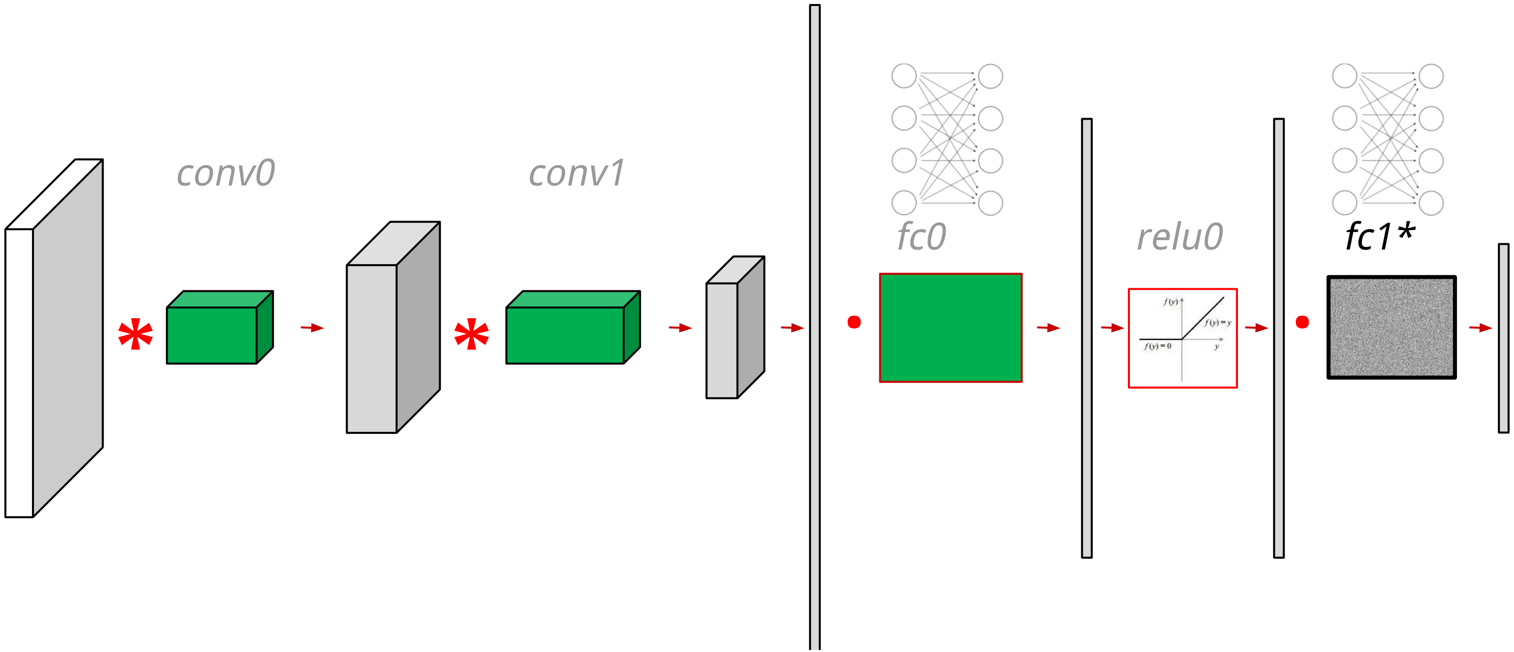
Pretrained network for problem P

Fine-tuning



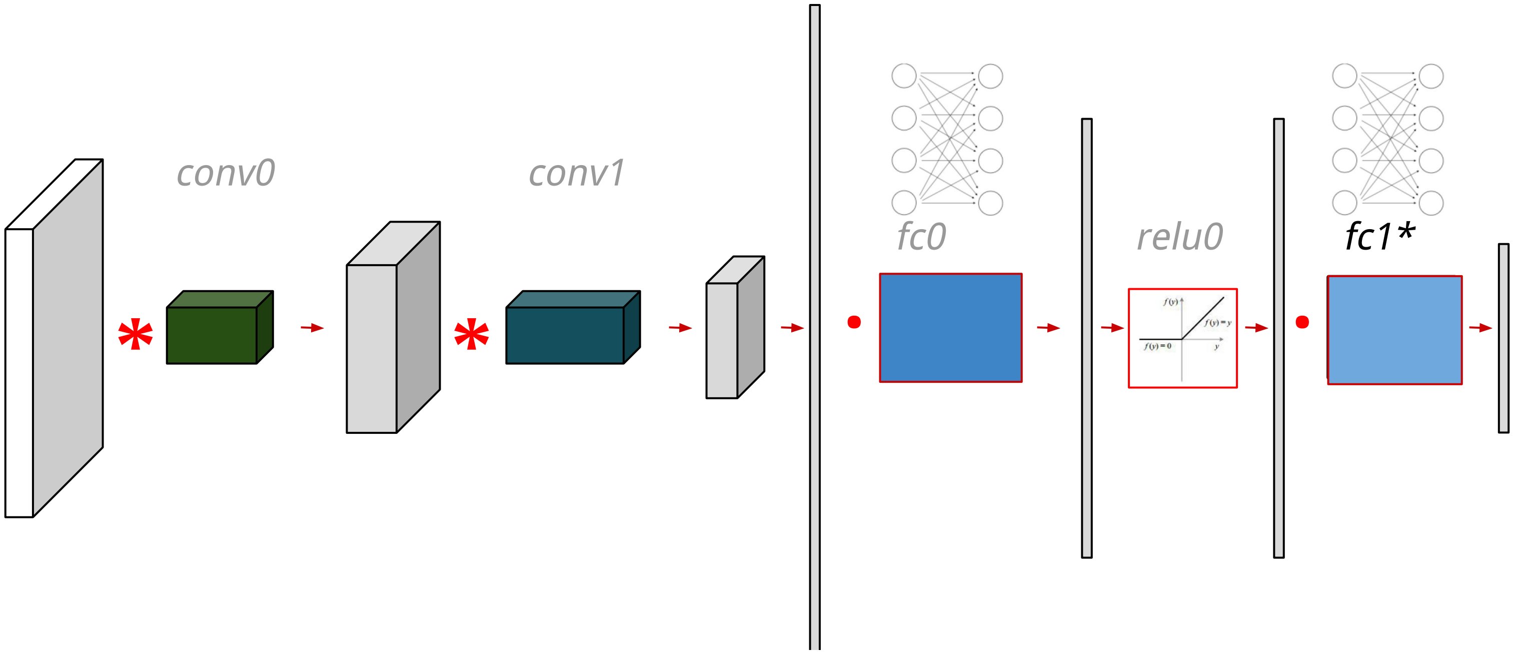
Substitute layer(s)

Fine-tuning



Train with new samples for problem Q

Fine-tuning



Train with new samples for problem Q

Image Semantics

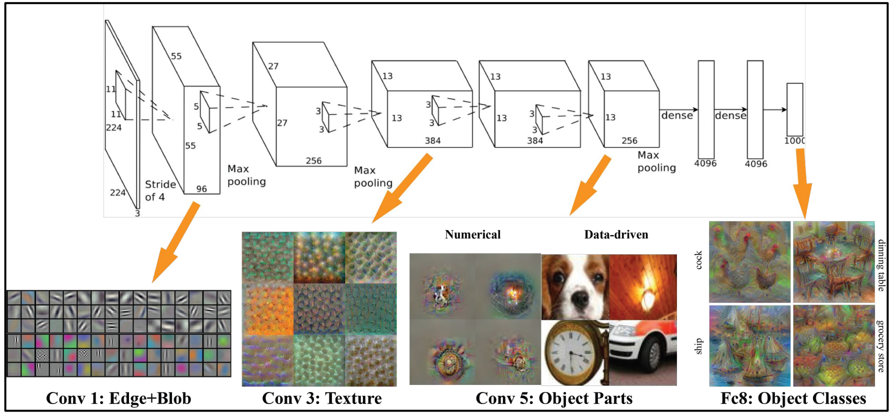
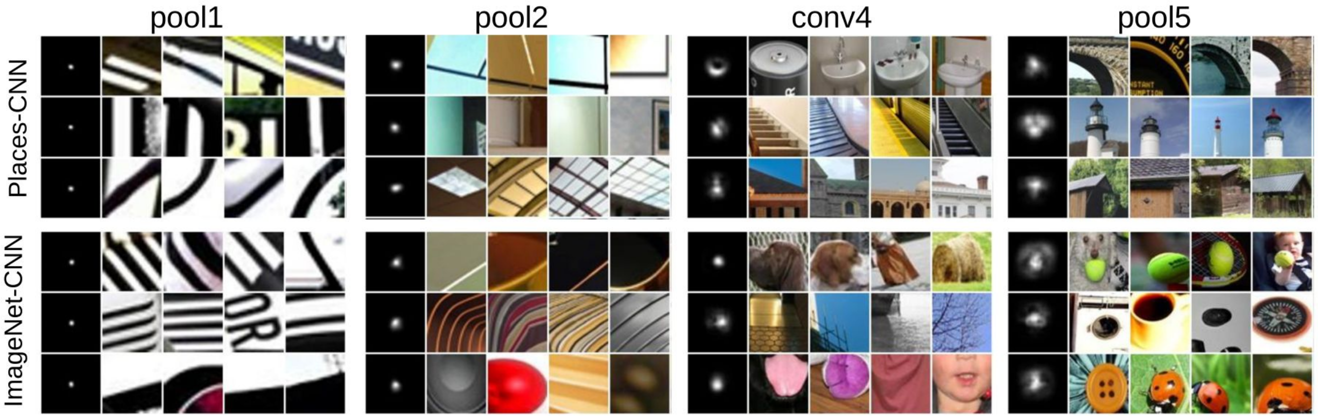


Image Semantics



Published as a conference paper at ICLR 2015

OBJECT DETECTORS EMERGE IN DEEP SCENE CNNS

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba
Computer Science and Artificial Intelligence Laboratory, MIT
{bolei,khosla,agata,oliva,torralba}@mit.edu

Learning Deep Features for Discriminative Localization

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba
Computer Science and Artificial Intelligence Laboratory, MIT
{bzhou,khosla,agata,oliva,torralba}@csail.mit.edu

Abstract

In this work, we revisit the global average pooling layer proposed in [13], and shed light on how it explicitly enables the convolutional neural network to have remarkable localization ability despite being trained on image-level labels. While this technique was previously proposed as a means for regularizing training, we find that it actually builds a generic localizable deep representation that can be applied to a variety of tasks. Despite the apparent simplicity of global average pooling, we are able to achieve 37.1% top-5

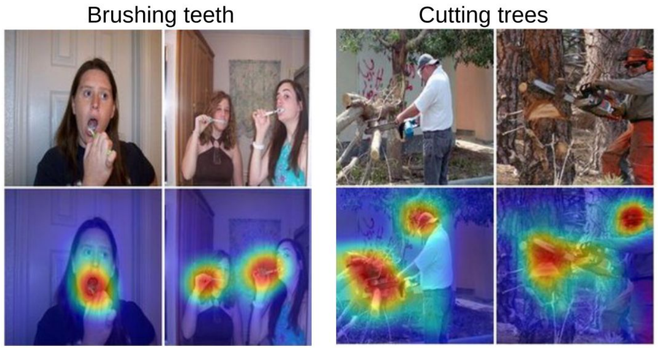
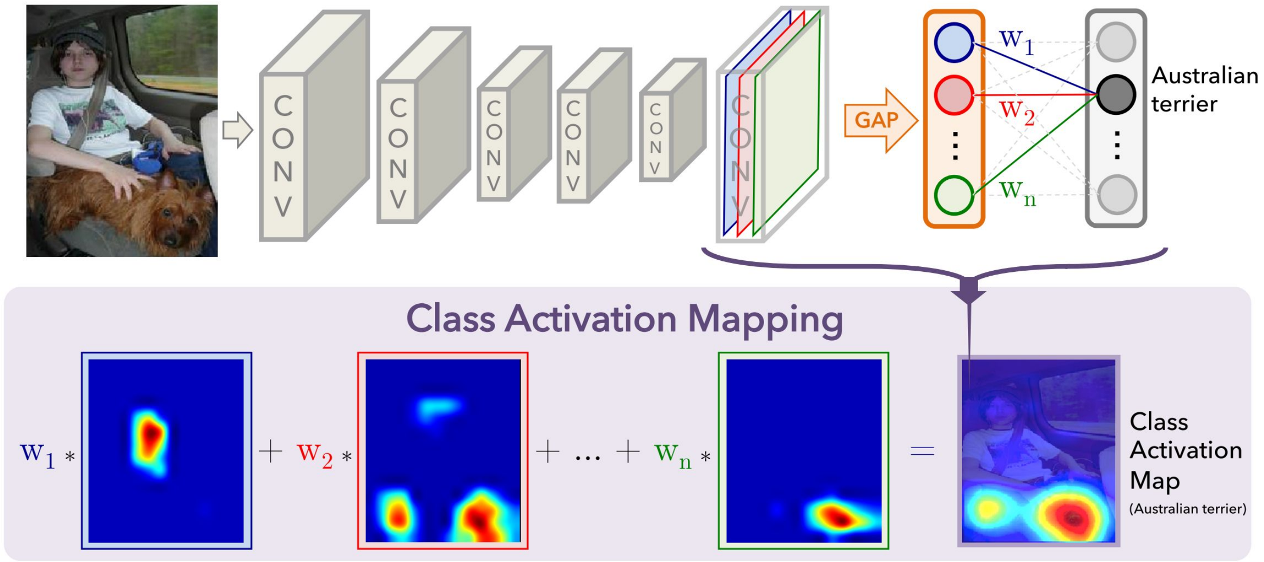


Figure 1. A simple modification of the global average pool-

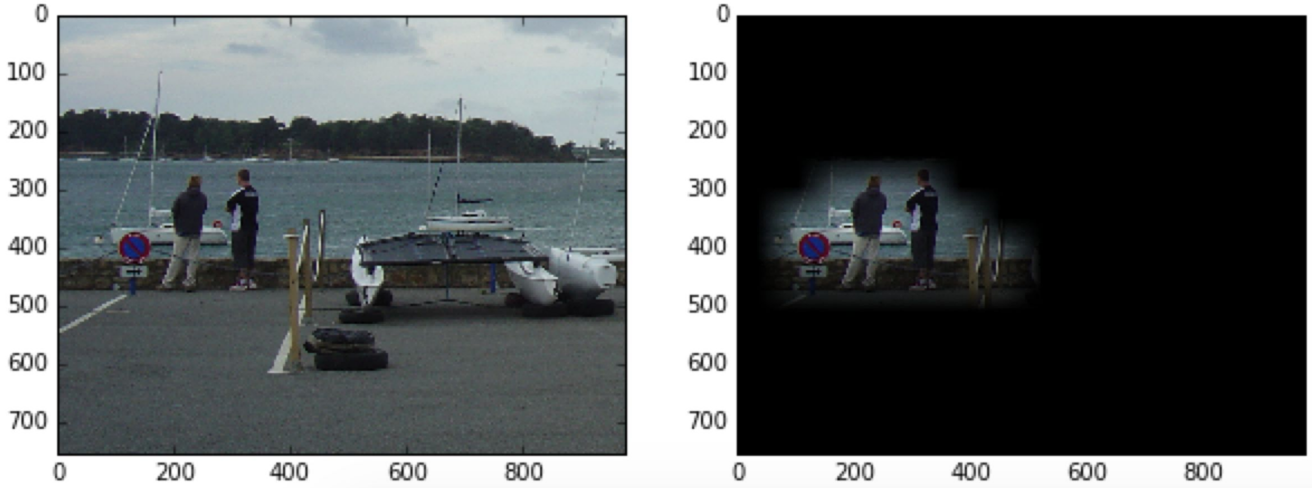


CONV
CONV
CONV
CONV
CONV
CONV
GAP
w_1
w_2
w_n
Australian terrier
Class Activation Mapping
$w_1 *$
$+ w_2 *$
$+ \dots + w_n *$
$=$
Class Activation Map
(Australian terrier)

Experiment: Localization using CAM

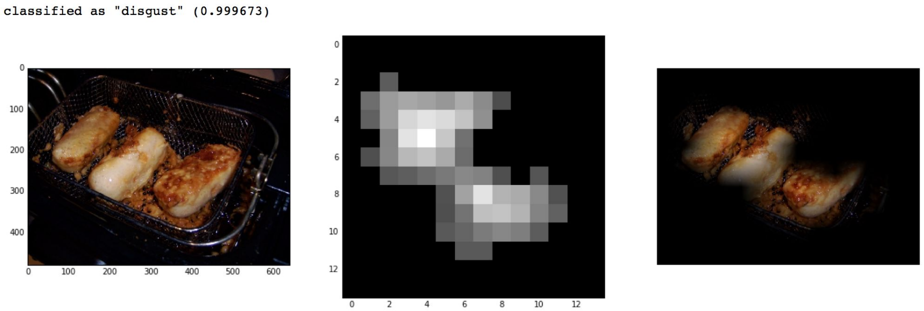
2 classes : {person, not person}

```
Image classified as 1 with probability 0.999978
Class no. 1 Activation Mapping
```

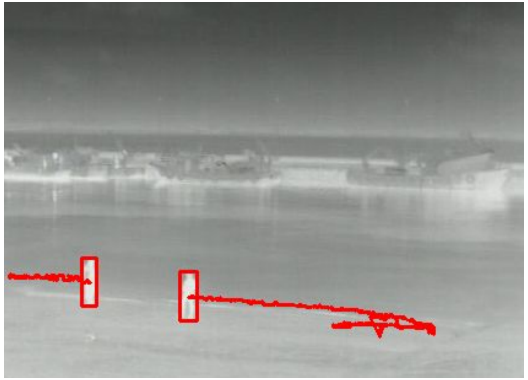


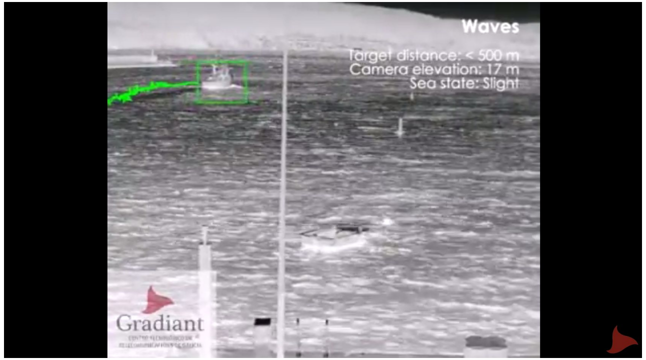
Experiment: Sentiment Analysis + CAM

6 classes : {anger, disgust, fear, joy, sadness, surprise}



New Challenges: Stable Object Detection

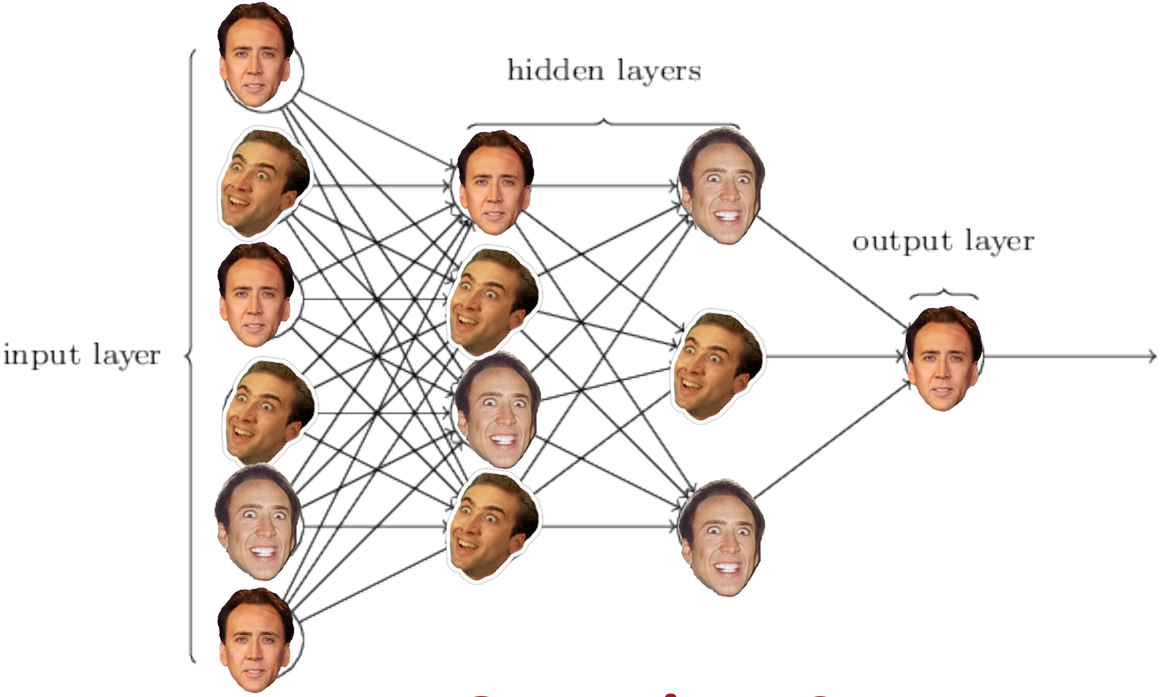




SEERS
Snapshot Spectral Imager for IR Surveillance

HORIZON 2020

THANK YOU!



Questions?

Machine Learning Workhop Galicia 2016