session 5:
Deep Learning

M. Kundegorski

13th December 2019

IAFIG-RMS - Bioimage Analysis With Python

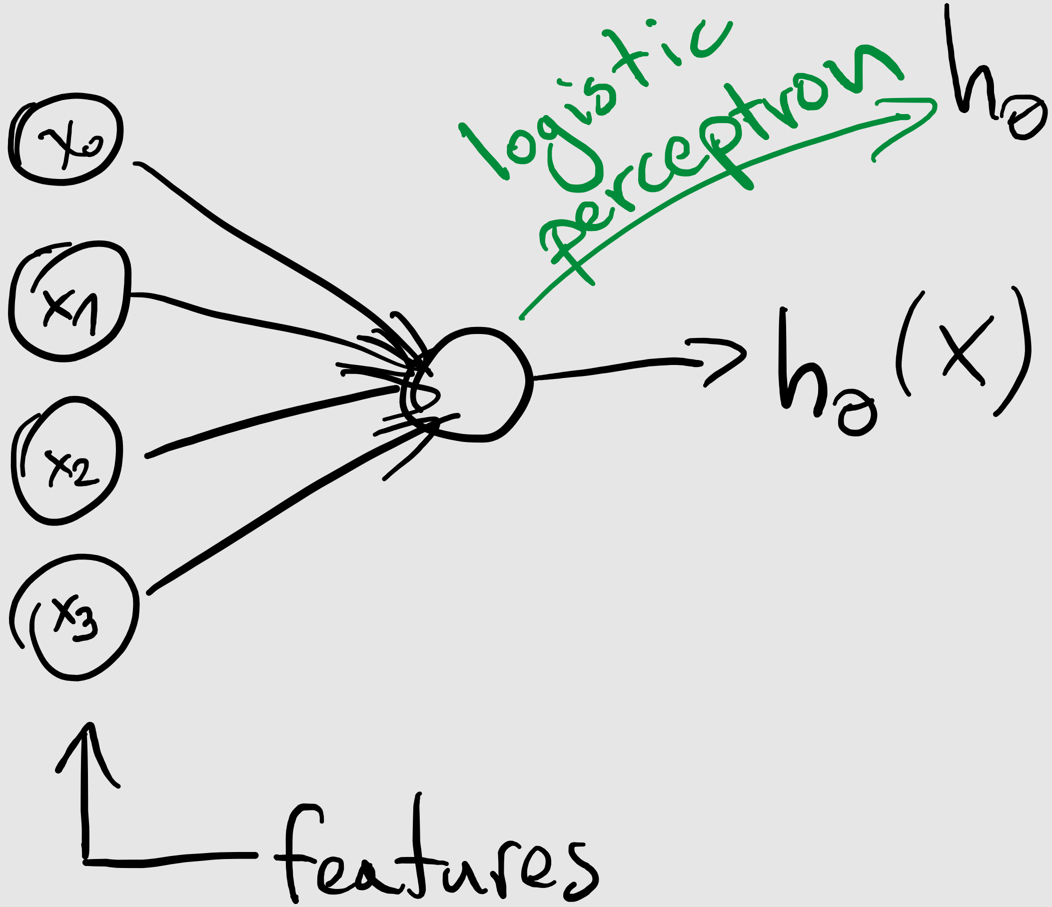
Cambridge Bioinformatics Training Centre

Royal Microscopical Society

Deep Learning

Accept that this is not magic.

From logistic regression to neural network

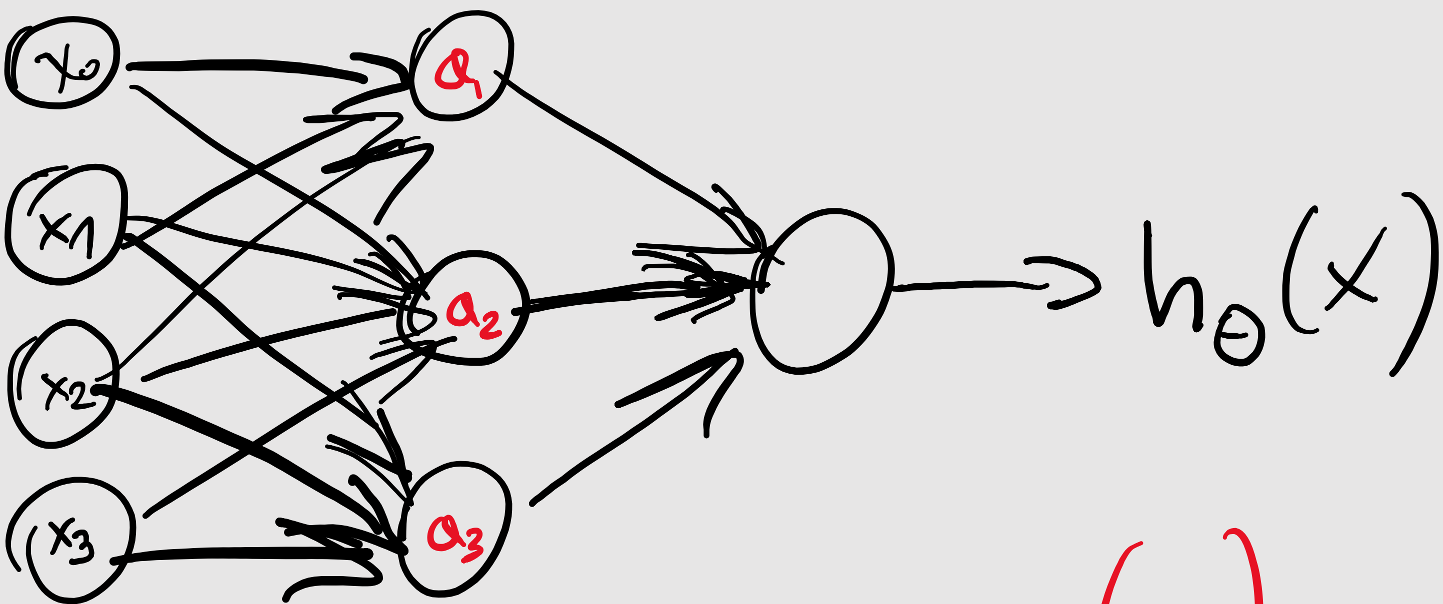


$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

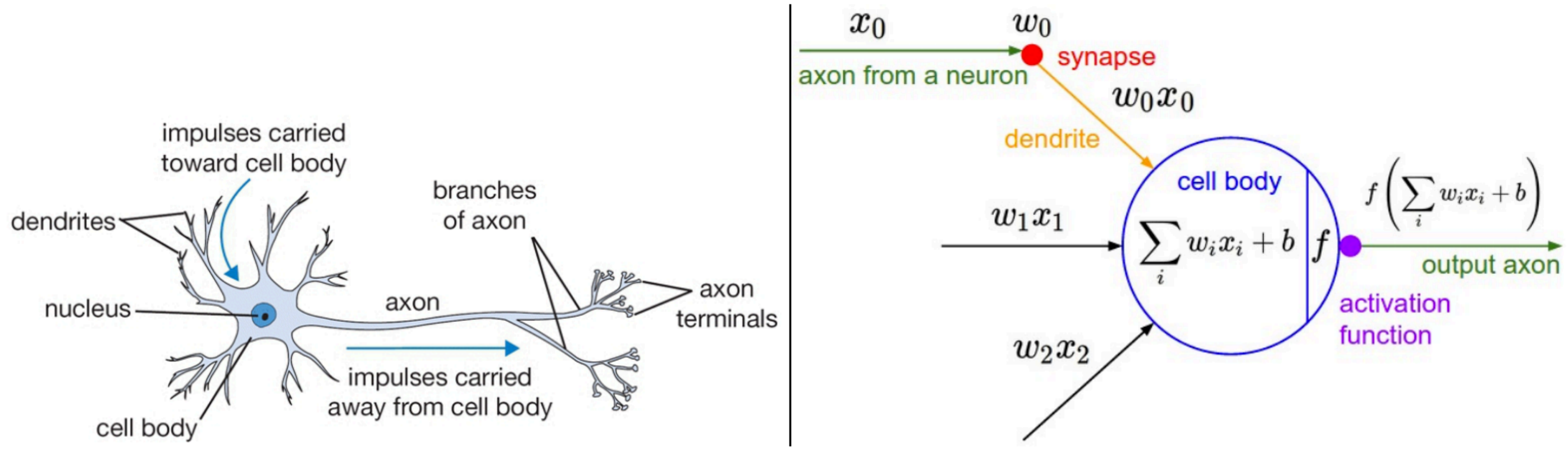
θ

parameters = weights

From logistic regression to neural network



$$a = g(x)$$



A cartoon drawing of a biological neuron (left) and its mathematical model (right).

source:CS231n Convolutional Neural Networks for Visual Recognition (https://cs231n.github.io)

Deep Learning Revolution

ImageNet Classification with Deep Convolutional Neural Networks

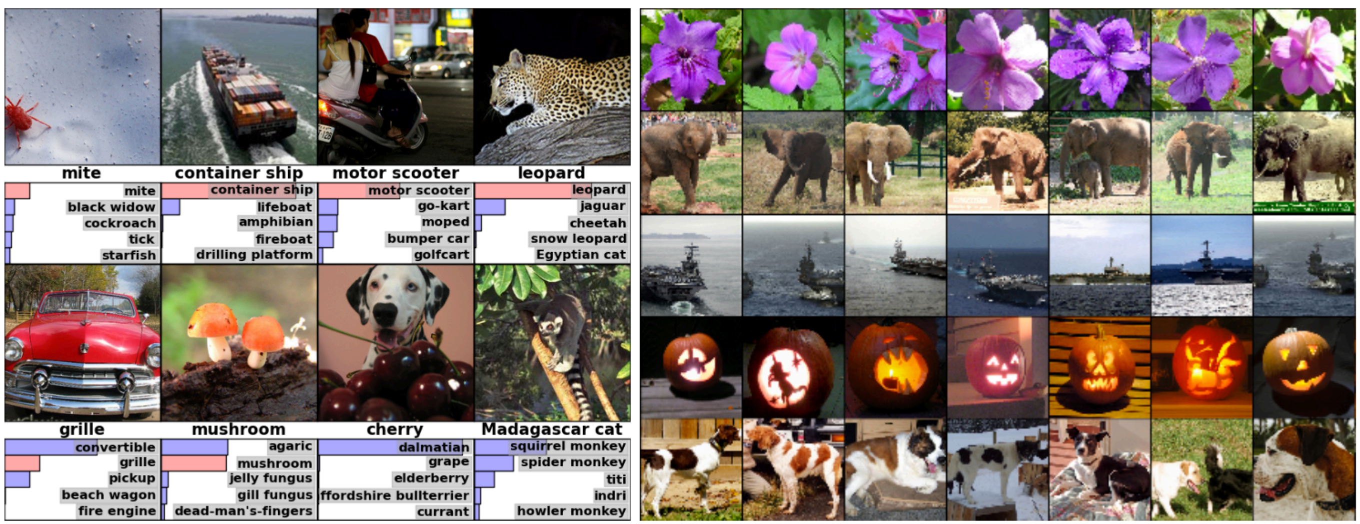
Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

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

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

Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

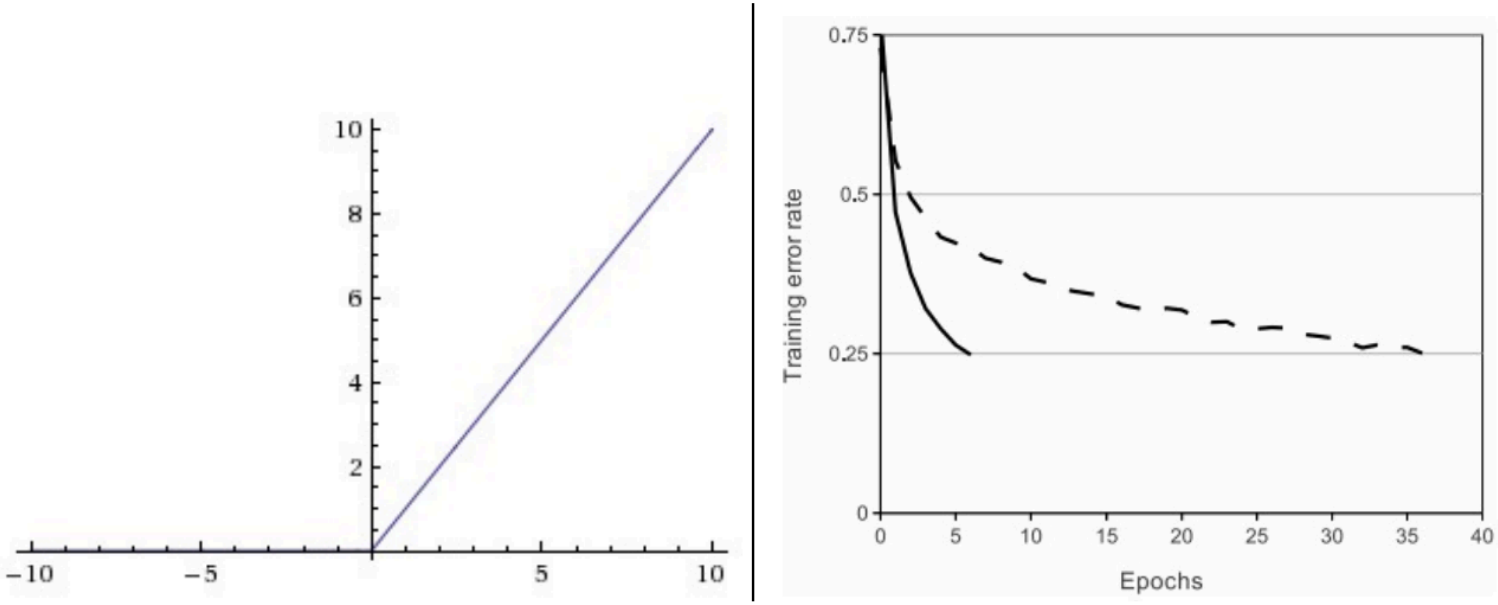
Table 1: Comparison of results on ILSVRC-2010 test set. In *italics* are best results achieved by others.



sc
(I

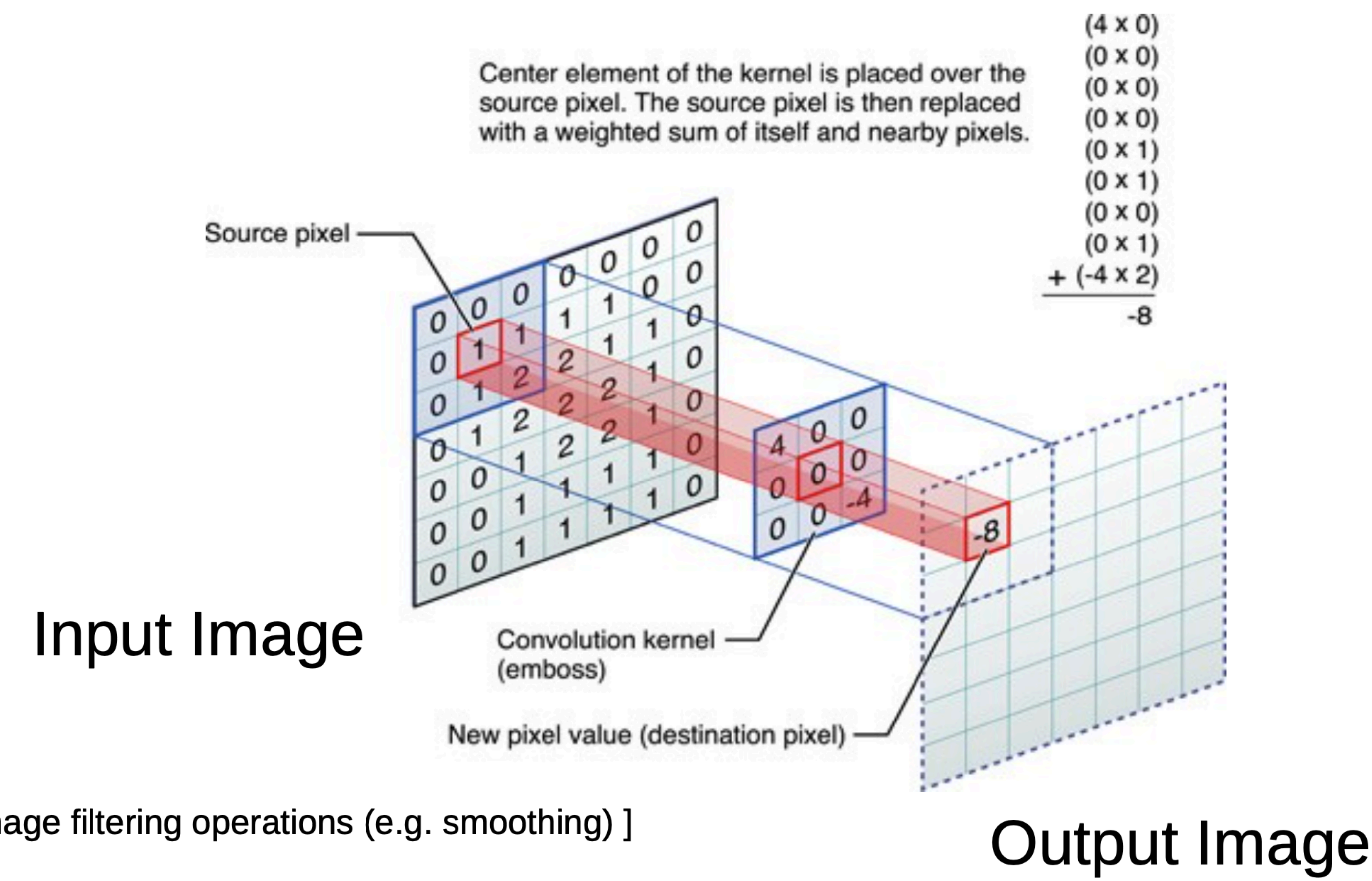
Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

ReLU activation – instead of Sigmoid



Left: Rectified Linear Unit (ReLU) activation function, which is zero when x < 0 and then linear with slope 1 when x > 0. **Right:** A plot from Krizhevsky et al. (pdf) paper indicating the 6x improvement in convergence with the ReLU unit compared to the tanh unit.

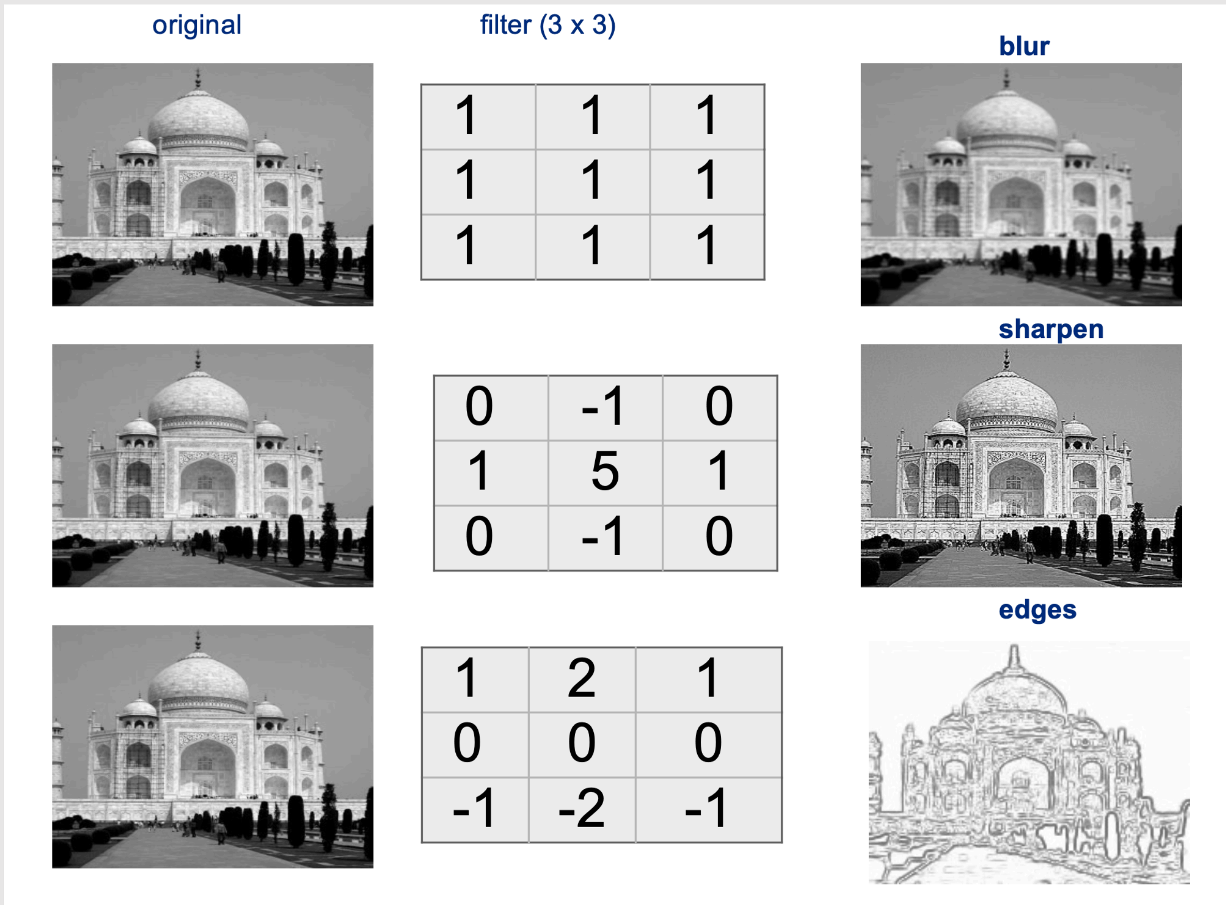
Convolution



[used in image filtering operations (e.g. smoothing)]

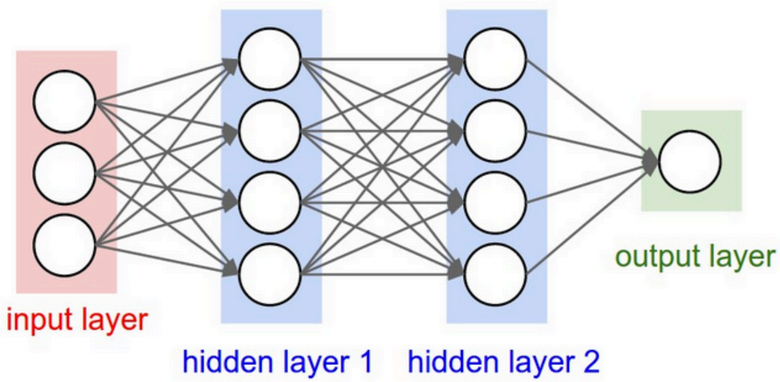
Image source: developer.apple.com

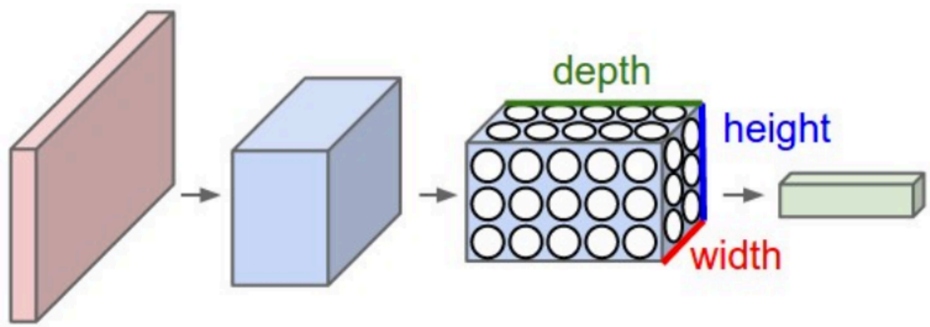
convolution

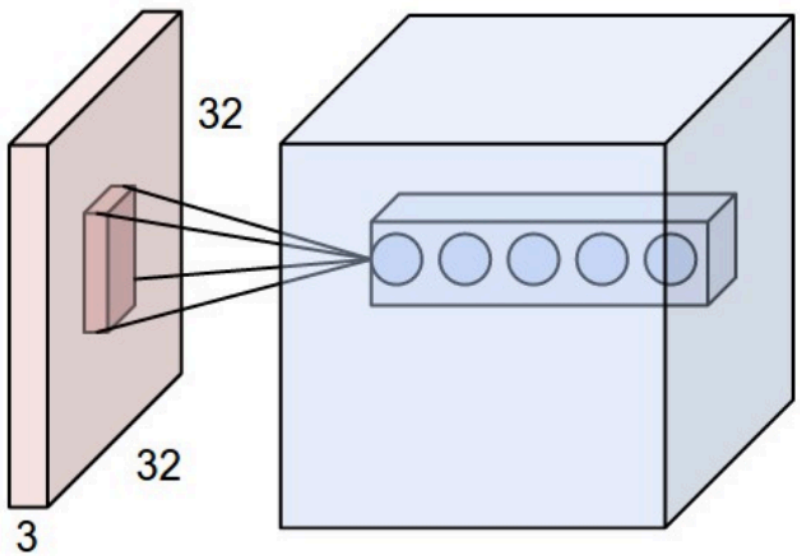


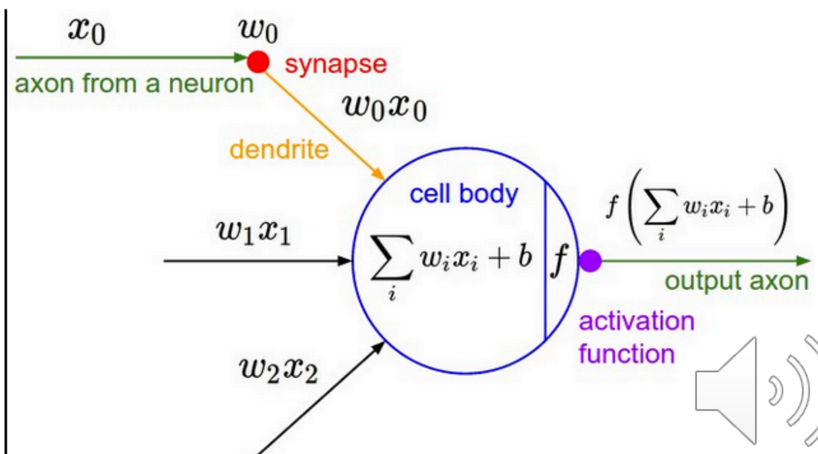
source: http://breckon.eu/toby/teaching/mltutorial/

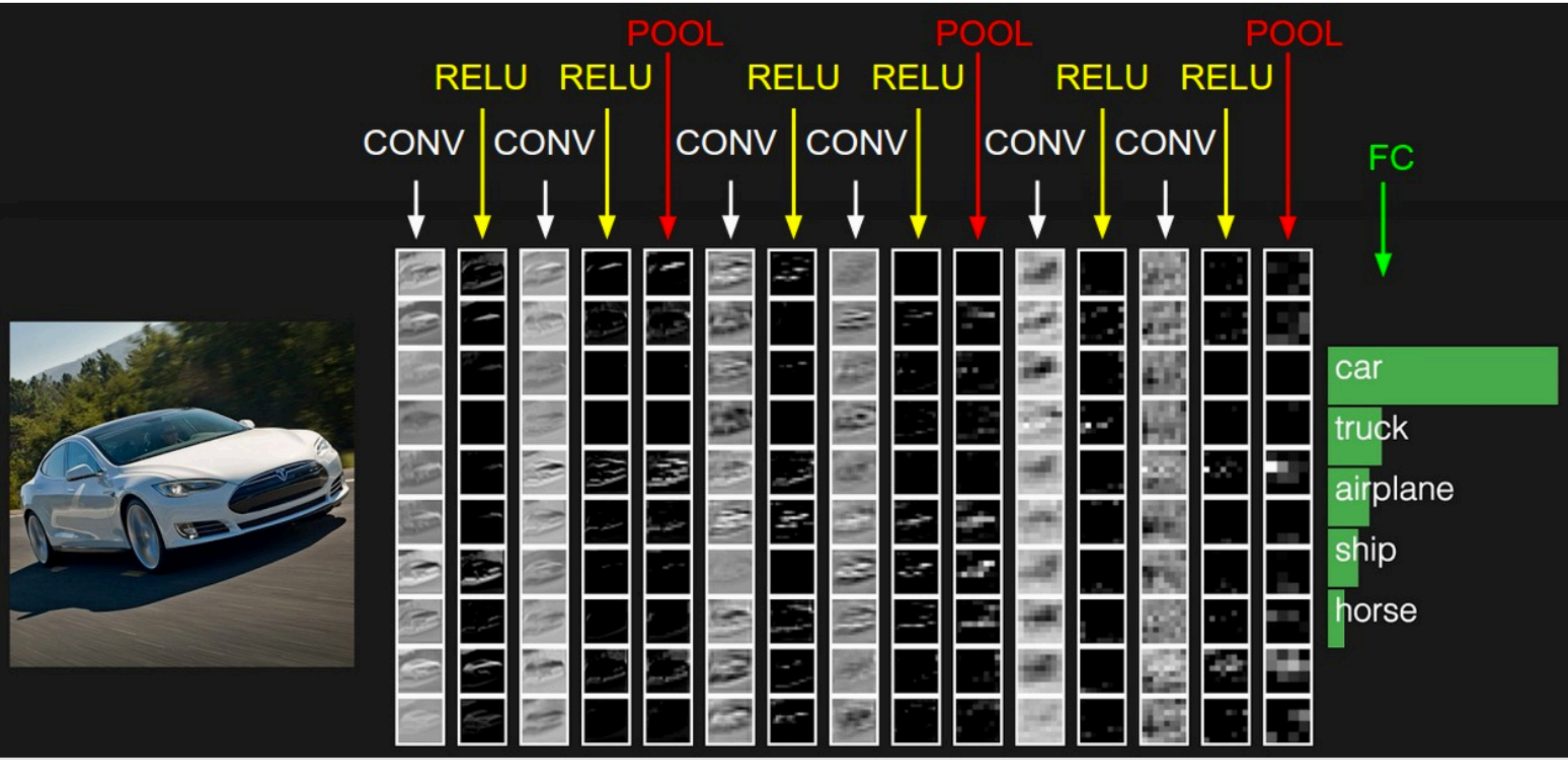
Convolutional Neural Networks



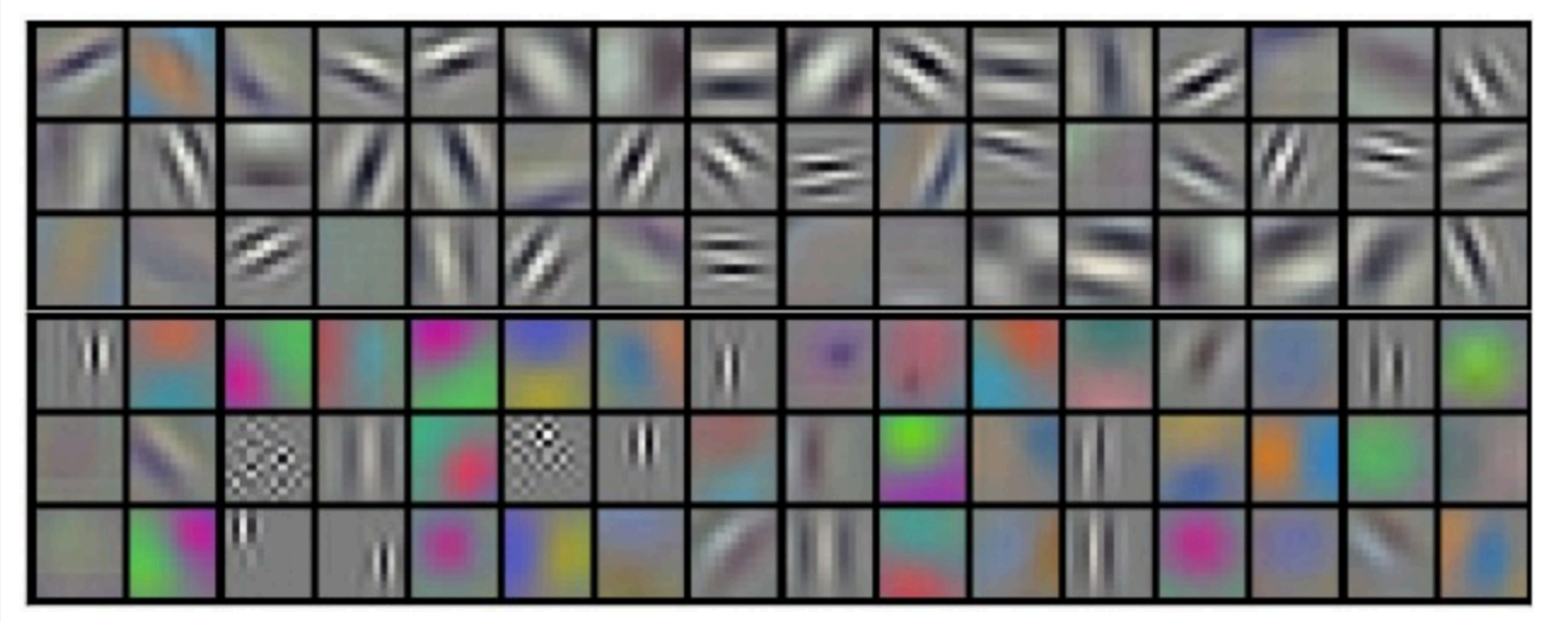




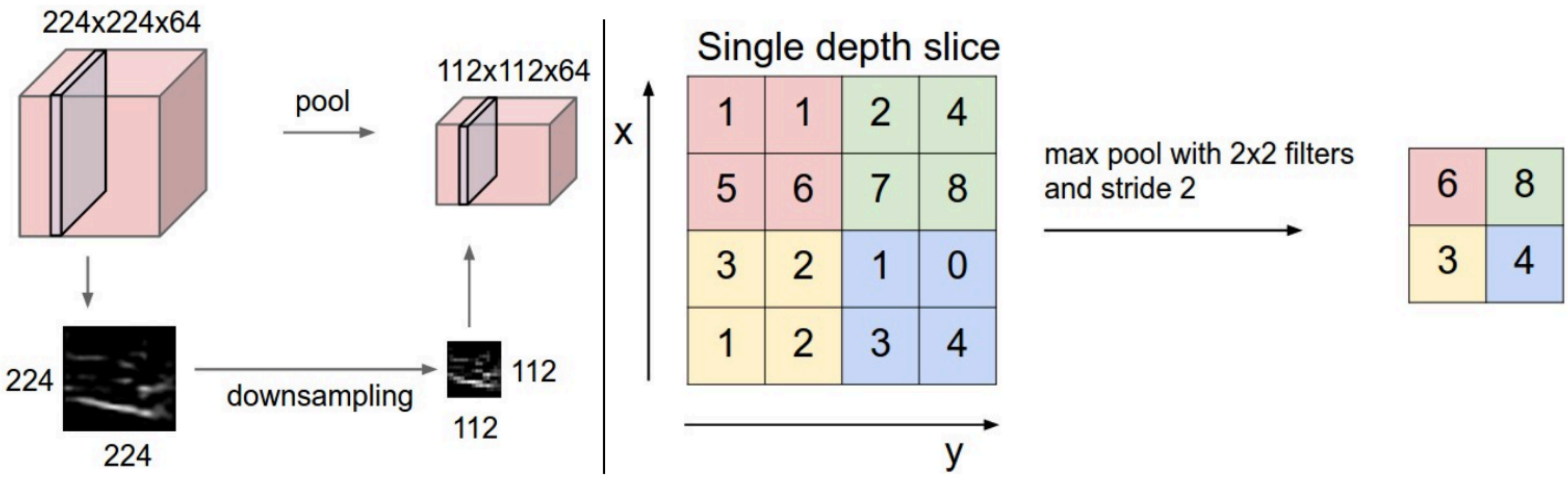



POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

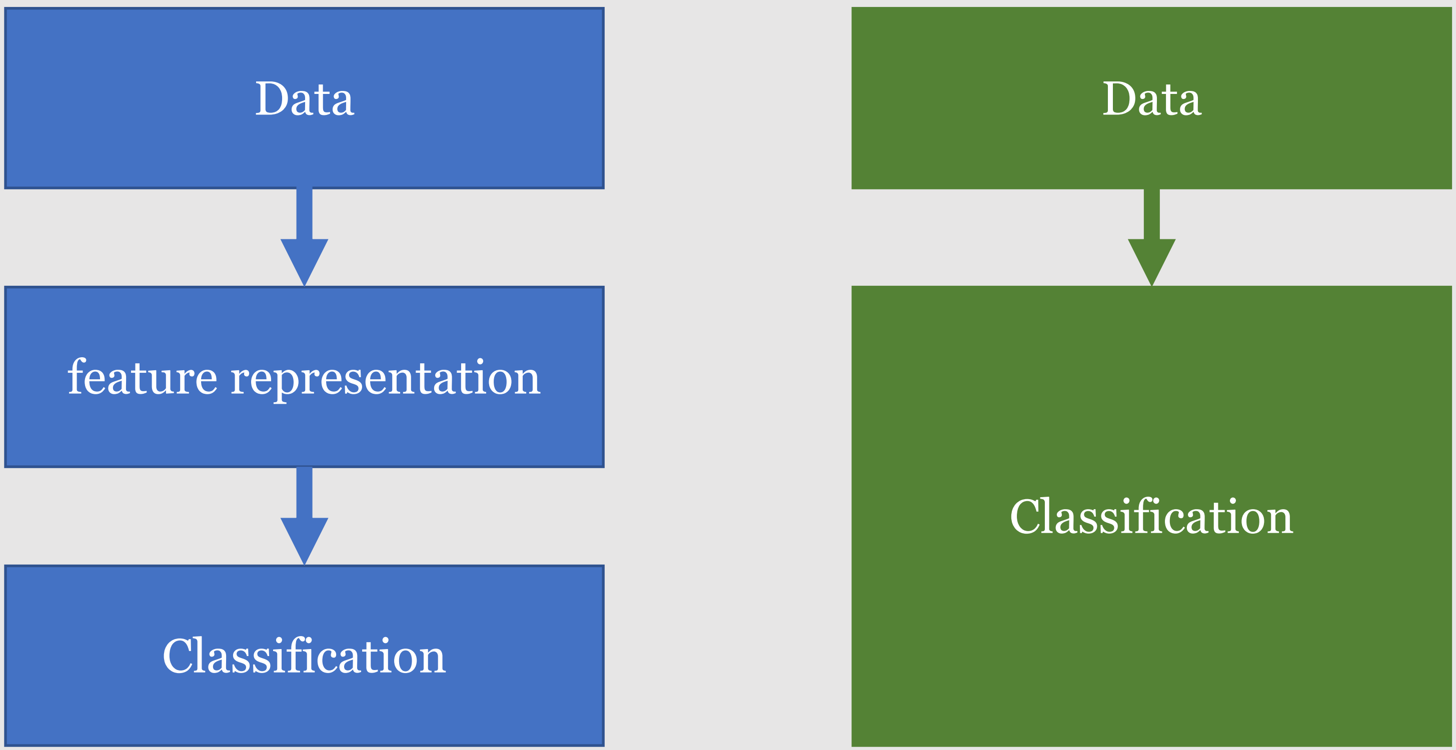
Features from Deep Learning



Max pooling


224x224x64
pool
112x112x64
224
224
downsampling
112
112
Single depth slice
x
y
max pool with 2x2 filters
and stride 2
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
6 8
3 4

Shallow vs Deep



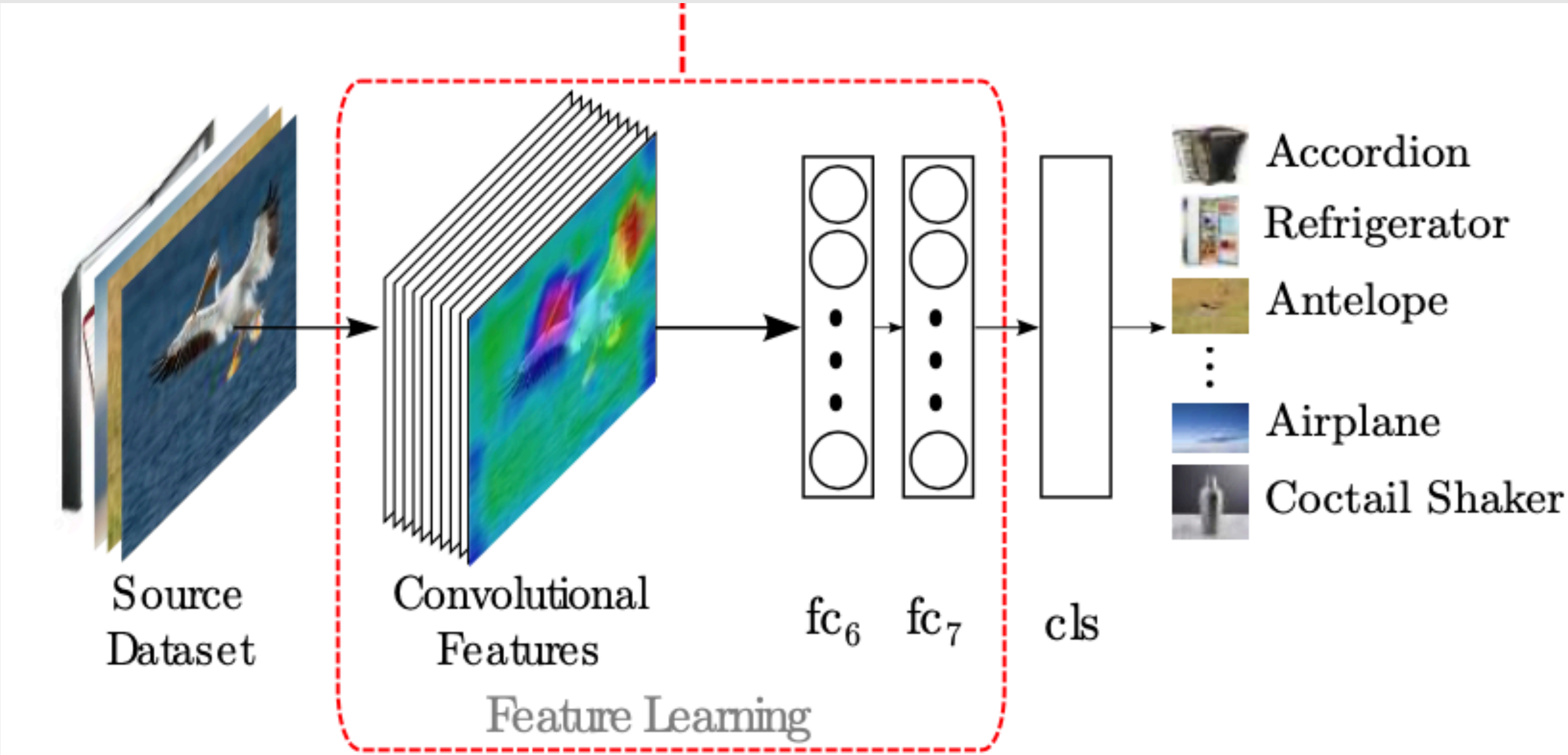
Why not to use deep learning

- requires big datasets
- takes long training time to learn hyperparameters
- Can a better classification be achieved using SVM on learned deep-learned features?

“Big” dataset

14 million annotated images

Transfer Learning



ImageNet Classification with Deep Convolutional Neural Networks
Krizhevsky , 2012 / Illustration Samet Ackay

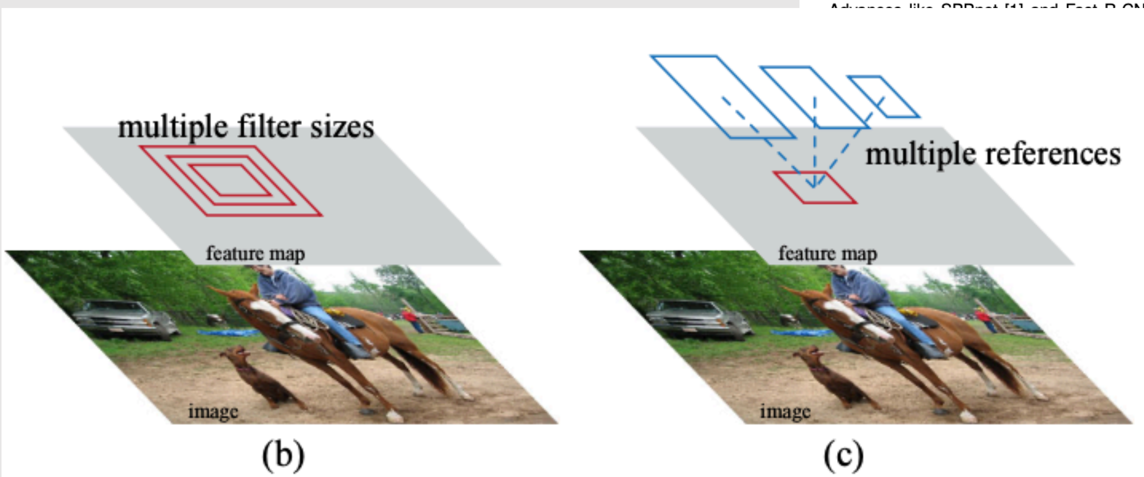
Applications

Object detection

Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun

Abstract—State-of-the-art object detection networks depend on region proposal algorithms to hypothesize object locations. Advances like SPPnet [1] and Fast R-CNN [2] have reduced the running time of these detection networks, exposing region
his work, we introduce a *Region Proposal Network* (RPN) that shares full-image
work, thus enabling nearly cost-free region proposals. An RPN is a fully convolutional
bounds and objectness scores at each position. The RPN is trained end-to-end to
ch are used by Fast R-CNN for detection. We further merge RPN and Fast R-CNN
olutional features—using the recently popular terminology of neural networks with



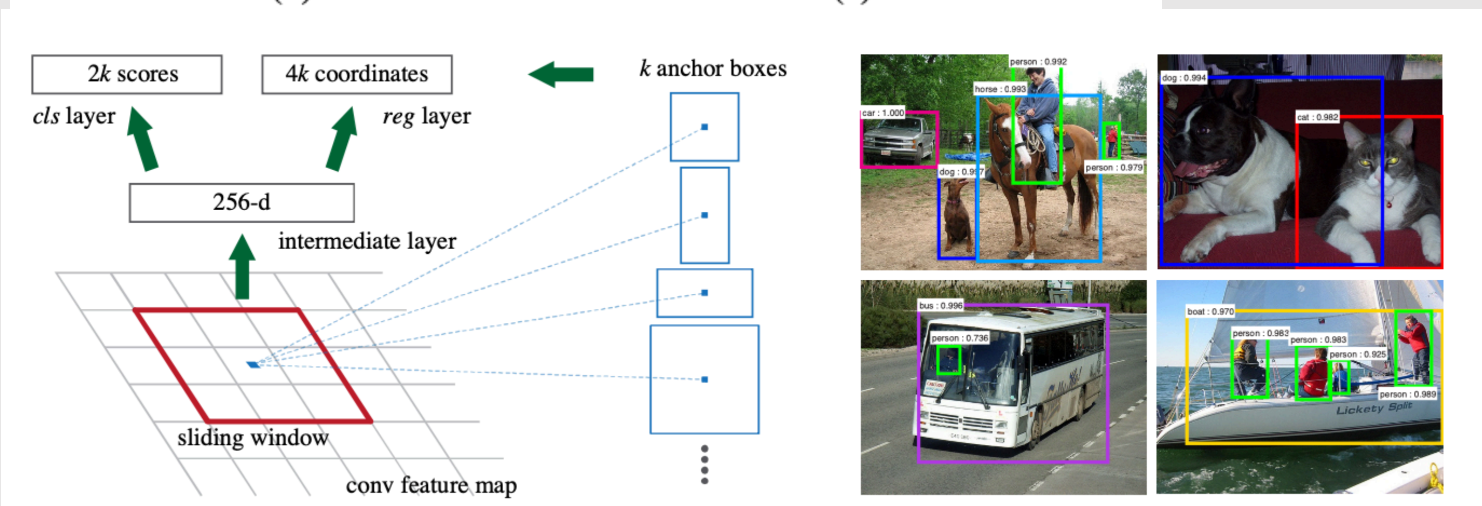
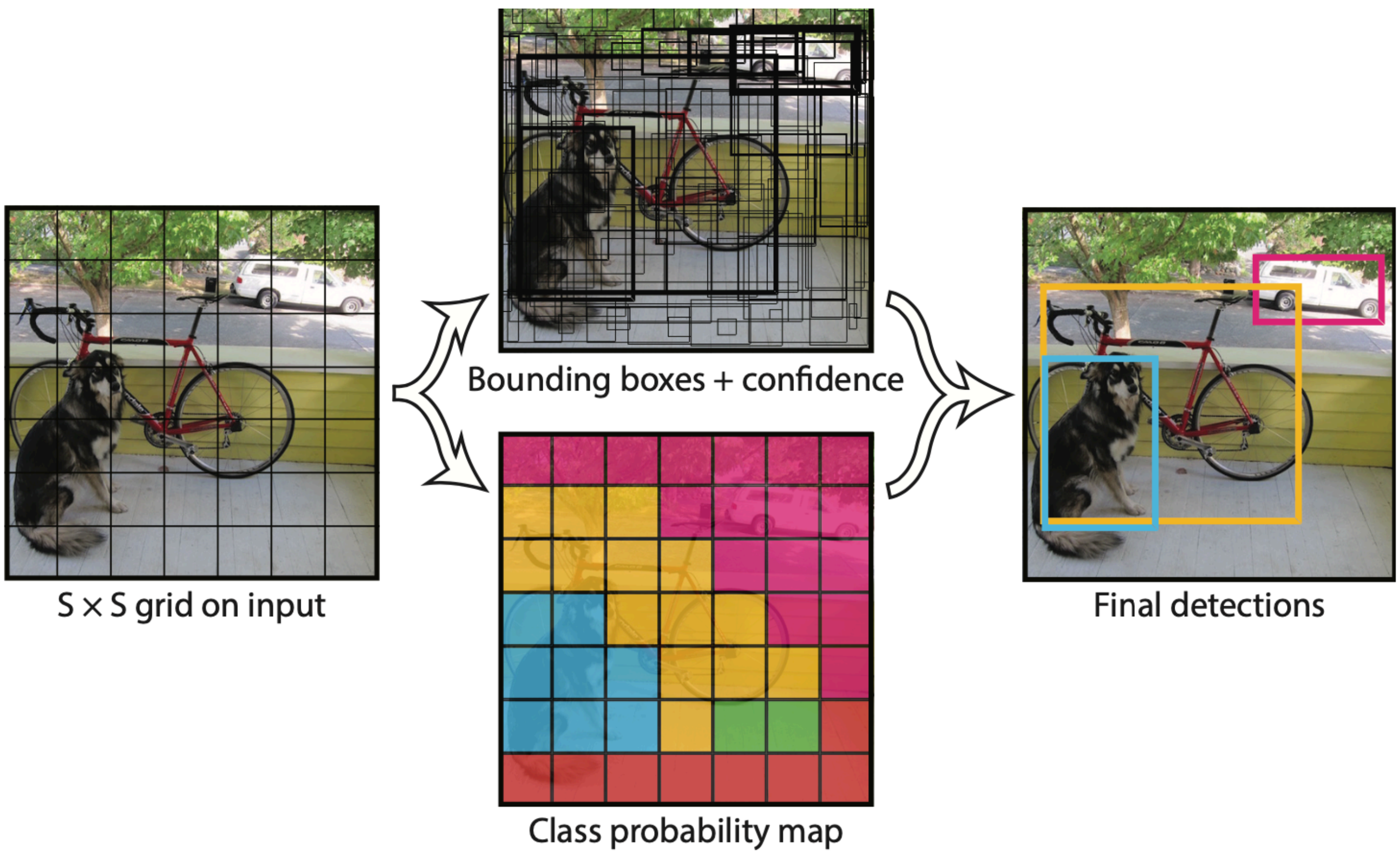


Figure 3: **Left**: Region Proposal Network (RPN). **Right**: Example detections using RPN proposals on PASCAL VOC 2007 test. Our method detects objects in a wide range of scales and aspect ratios.

You Only Look Once: Unified, Real-Time Object Detection

Joseph Redmon*, Santosh Divvala*†, Ross Girshick¶, Ali Farhadi*†
University of Washington*, Allen Institute for AI†, Facebook AI Research¶

http://pjreddie.com/yolo/

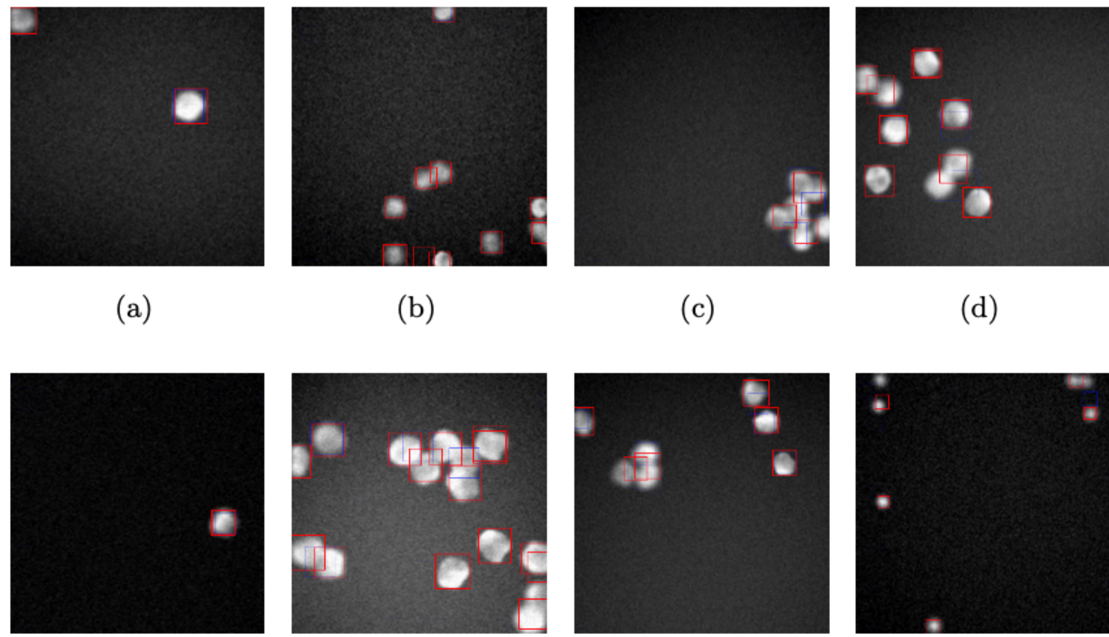


cell counting

Avoiding Over-Detection: Towards Combined Object Detection and Counting

Philip T. G. Jackson and Boguslaw Obara*

School of Engineering and Computing Sciences
Durham University
South Road, DH1 3LE, Durham, UK
{p.t.g.jackson,boguslaw.obara}@durham.ac.uk



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 2: A sample of detection results on SIMCEP images. Confi[illegible]ce is repre-
sented in the transparency of the boxes; all output boxes with con[illegible]ence abov
0.1 are shown. Instead of post-processing with NMS, we simply take boxes wit
confidence above 0.5 (shown in red) as positive detections. Boxes with confidenc

la

The iNaturalist Species Classification and Detection Dataset

Grant Van Horn[1] Oisin Mac Aodha[1] Yang Song[2] Yin Cui[3] Chen Sun[2]
Alex Shepard[4] Hartwig Adam[2] Pietro Perona[1] Serge Belongie[3]

[1]Caltech [2]Google [3]Cornell Tech [4]iNaturalist

	Super-Class	Class	Train	Val	BBoxes
	Plantae	2,101	158,407	38,206	-
	Insecta	1,021	100,479	18,076	125,679
	Aves	964	214,295	21,226	311,669
	Reptilia	289	35,201	5,680	42,351
	Mammalia	186	29,333	3,490	35,222
	Fungi	121	5,826	1,780	-
	Amphibia	115	15,318	2,385	18,281
	Mollusca	93	7,536	1,841	10,821
	Animalia	77	5,228	1,362	8,536
	Arachnida	56	4,873	1,086	5,826
	Actinopterygii	53	1,982	637	3,382
	Chromista	9	398	144	-
	Protozoa	4	308	73	-
	Total	5,089	579,184	95,986	561,767

Table 2. Number of images, classes, and bounding boxes in iNat2017 broken down by super-class. 'Animalia' is a catch-all category that contains species that do not fit in the other super-classes. Bounding boxes were collected for nine of the super-classes. In addition, the public and private test sets contain 90,427 and 92,280 images, respectively.

Two-spotted ladybug *Adalia bipunctata*

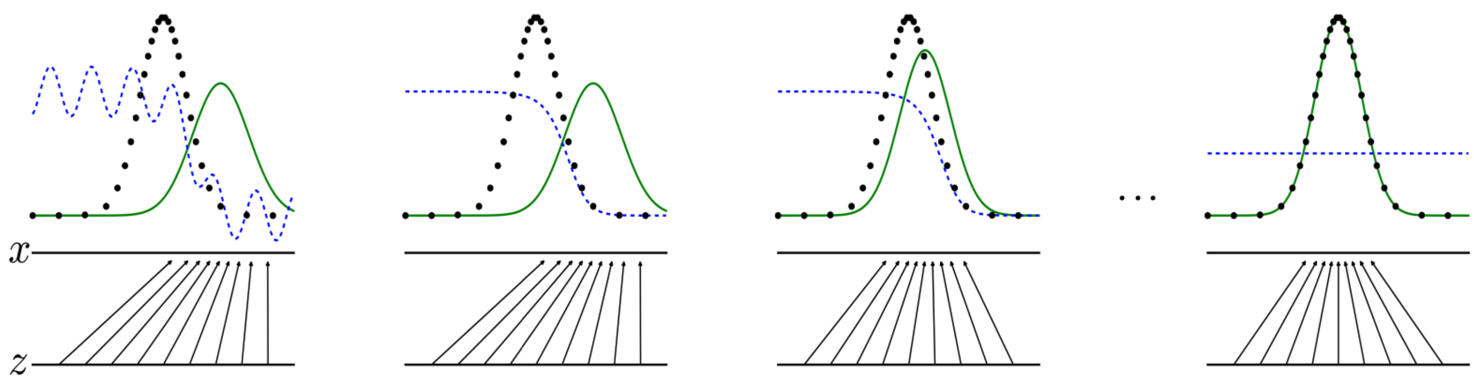
Seven-spotted ladybug *Coccinella septempunctata*

Figure 1. Two visually similar species from the iNat2017 dataset.

GANs

Generative Adversarial Nets

Ian J. Goodfellow,* Jean Pouget-Abadie,† Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,‡ Aaron Courville, Yoshua Bengio§
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7



(a) (b) (c) (d)

Figure 1: Generative adversarial nets are trained by simultaneously updating the **d**iscriminative distribution (D, blue, dashed line) so that it discriminates between samples from the data generating distribution (black, dotted line) $p_{\boldsymbol{x}}$ from those of the **g**enerative distribution p_g (G) (green, solid line). The lower horizontal line is the domain from which $\boldsymbol{z}$ is sampled, in this case uniformly. The horizontal line above is part of the domain of $\boldsymbol{x}$. The upward arrows show how the mapping $\boldsymbol{x} = G(\boldsymbol{z})$ imposes the non-uniform distribution p_g on transformed samples. G contracts in regions of high density and expands in regions of low density of p_g. (a) Consider an adversarial pair near convergence: p_g is similar to p_{data} and D is a partially accurate classifier. (b) In the inner loop of the algorithm D is trained to discriminate samples from data, converging to $D^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x})+p_g(\boldsymbol{x})}$. (c) After an update to G, gradient of D has guided $G(\boldsymbol{z})$ to flow to regions that are more likely to be classified as data. (d) After several steps of training, if G and D have enough capacity, they will reach a point at which both cannot improve because $p_g = p_{\text{data}}$. The discriminator is unable to differentiate between the two distributions, i.e. $D(\boldsymbol{x}) = \frac{1}{2}$.

GANs

Fig. 3. The three rows show the synthetic bird images of TSGAN-IMG and GAN-IMG, and the real images of Real-IMG.

SYNTHESIS OF IMAGES BY TWO-STAGE GENERATIVE ADVERSARIAL NETWORKS

Qiang Huang, Philip J.B. Jackson, Mark D. Plumbley, Wenwu Wang

Centre for Vision, Speech and Signal Processing
University of Surrey, Guildford, UK
Email: {q.huang, p.jackson, m.plumbley, w.wang}@surrey.ac.uk

Detect to Track and Track to Detect

Christoph Feichtenhofer *
Graz University of Technology
feichtenhofer@tugraz.at

Axel Pinz
Graz University of Technology
axel.pinz@tugraz.at

Andrew Zisserman
University of Oxford
az@robots.ox.ac.uk

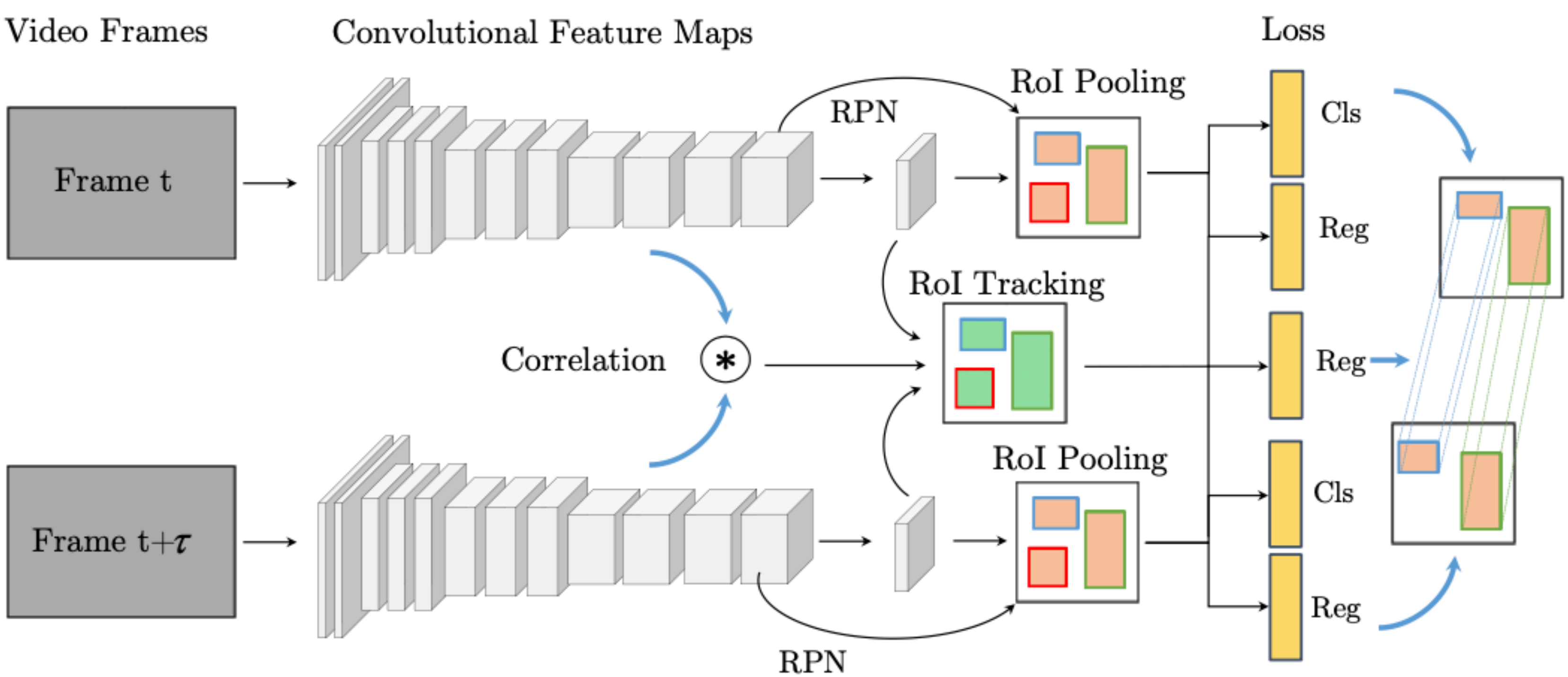
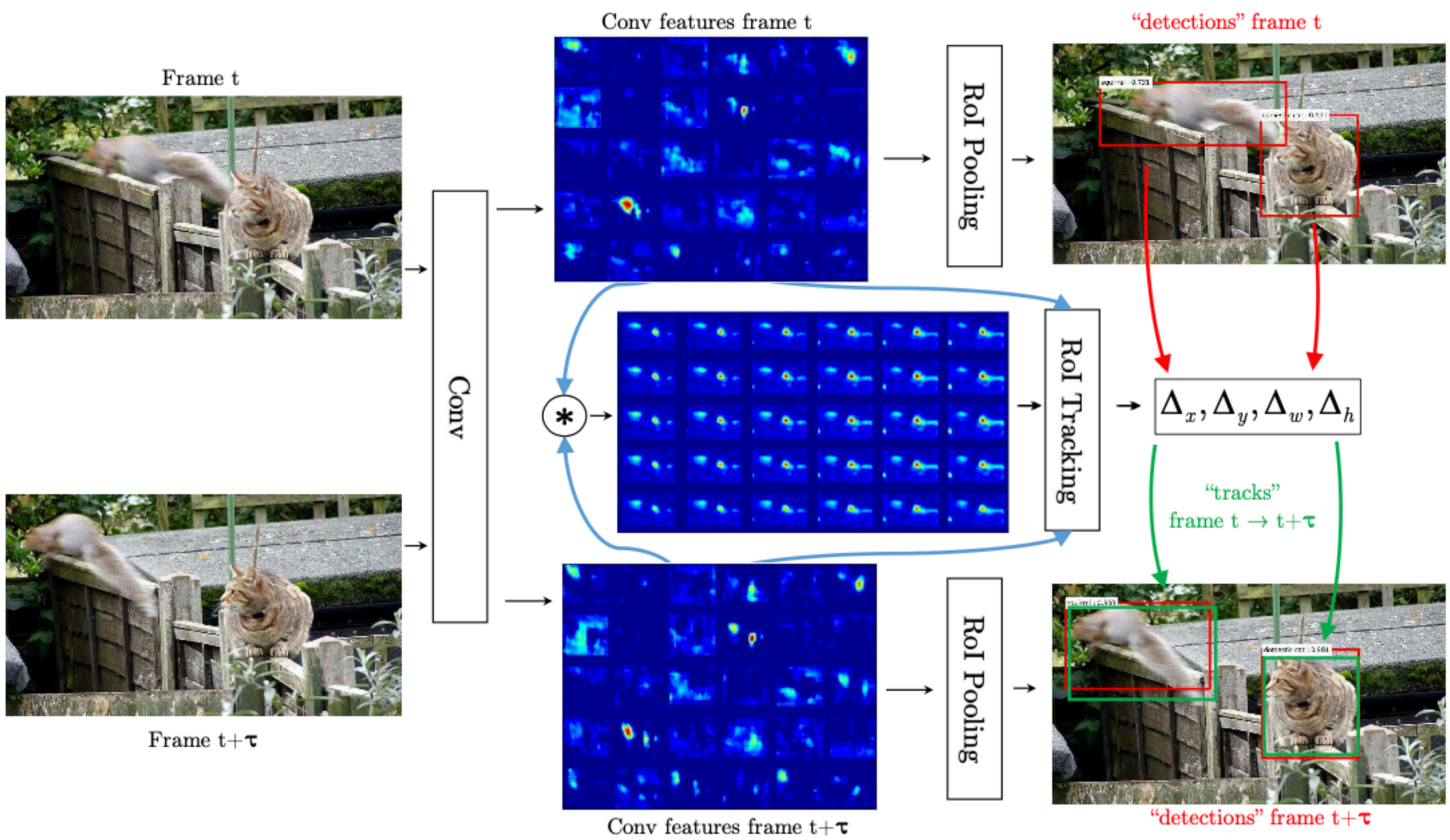


Figure 2. Architecture of our Detect and Track (D&T) approach (see Section 3 for details).

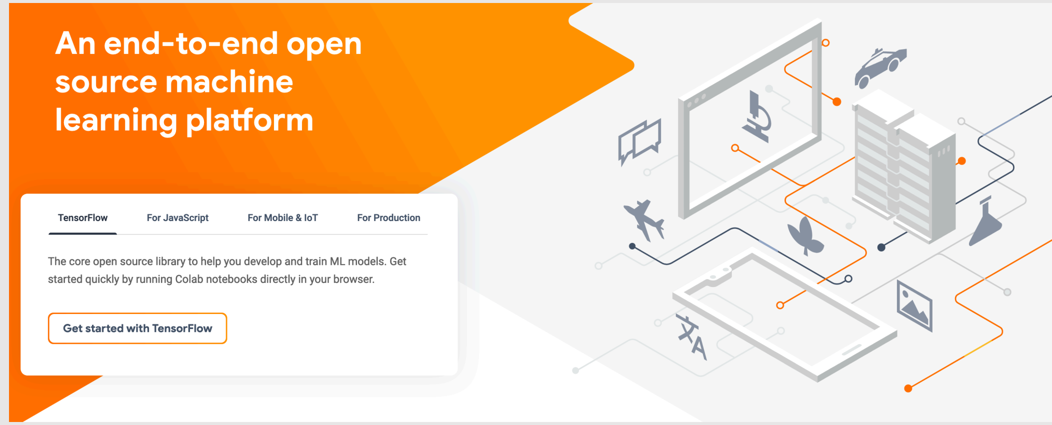
Deep Tracking



Hardware requirements

- GPU vs CPU – order of magnitude speed-up
- High memory requirements
- CUDA programming

Software



- Tensorflow
- Keras: ”easy” python library
- PyTorch

PYTORCH