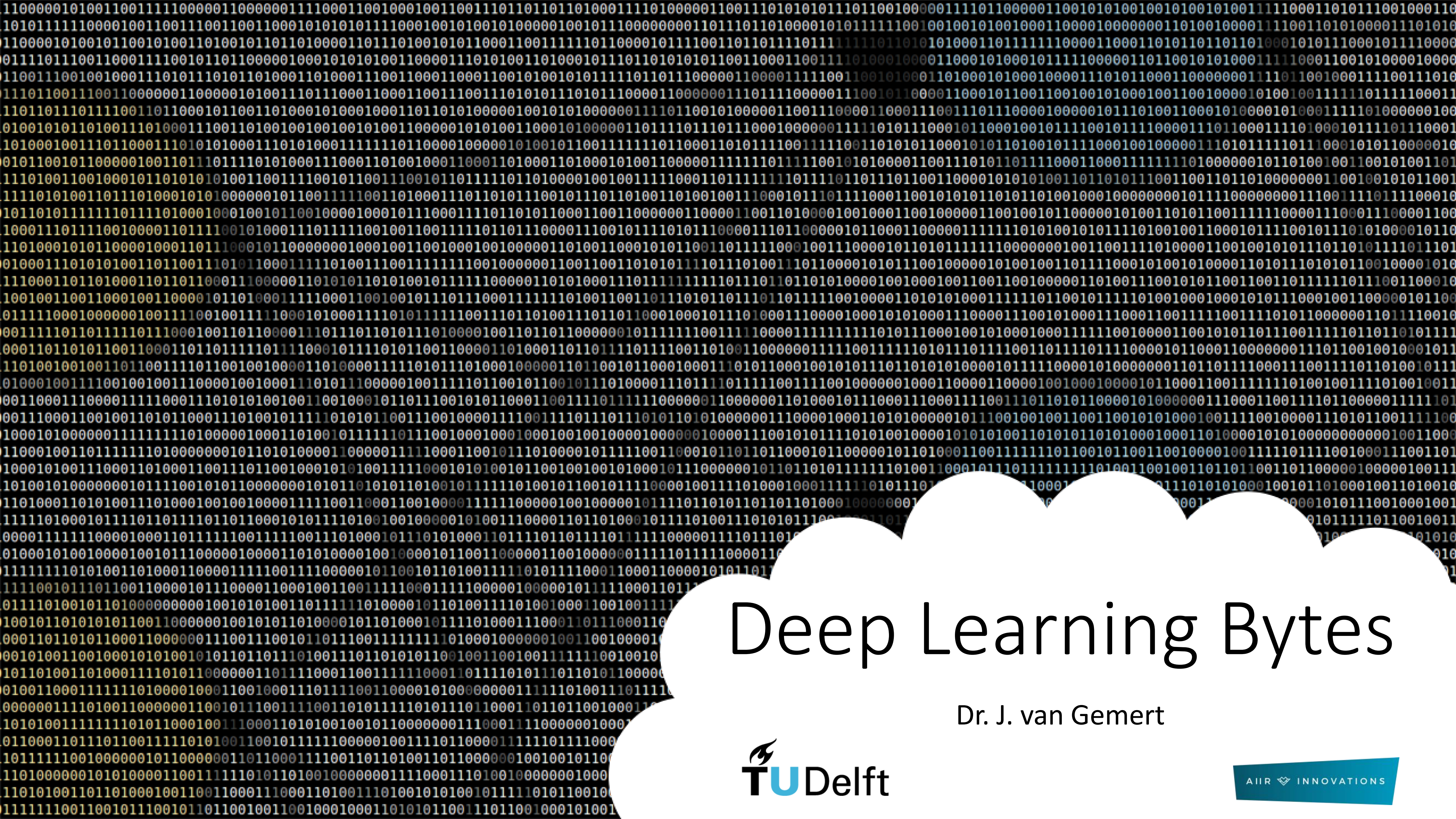


Deep Learning Bytes

Dr. J. van Gemert

TU Delft

AIIR INNOVATIONS

MonoPerfCap:
Human Performance Capture from Monocular Video

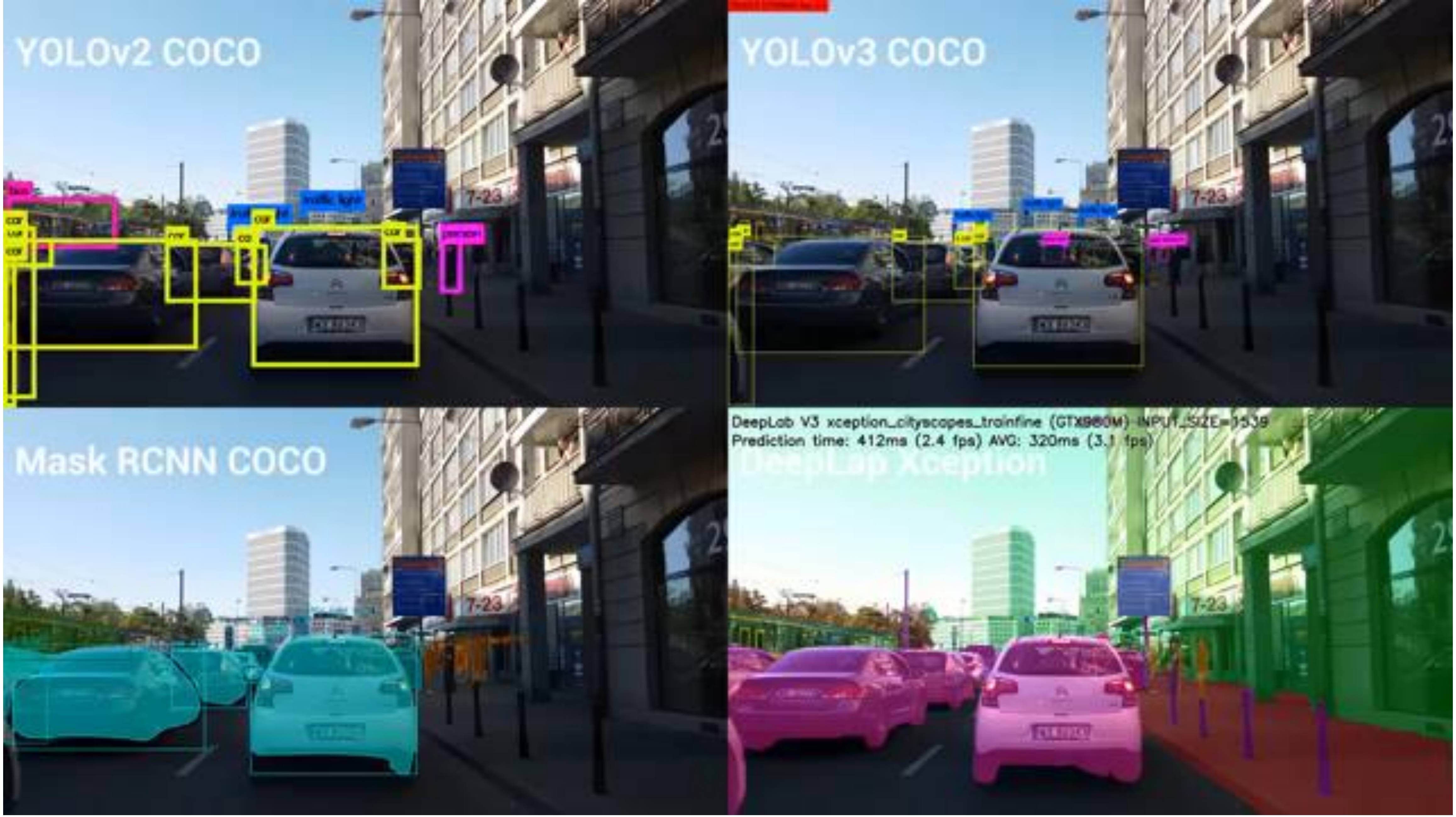
(with voiceover)

Weipeng Xu[1] Avishek Chatterjee[1] Michael Zollhöfer[1] Helge Rhodin[2]
Dushyant Mehta[1] Hans-Peter Seidel[1] Christian Theobalt[1]

[1]Max Planck Institute for Informatics, Saarland Informatics Campus [2]EPFL

Video credits: “MonoPerfCap”, Xu et al., 2018

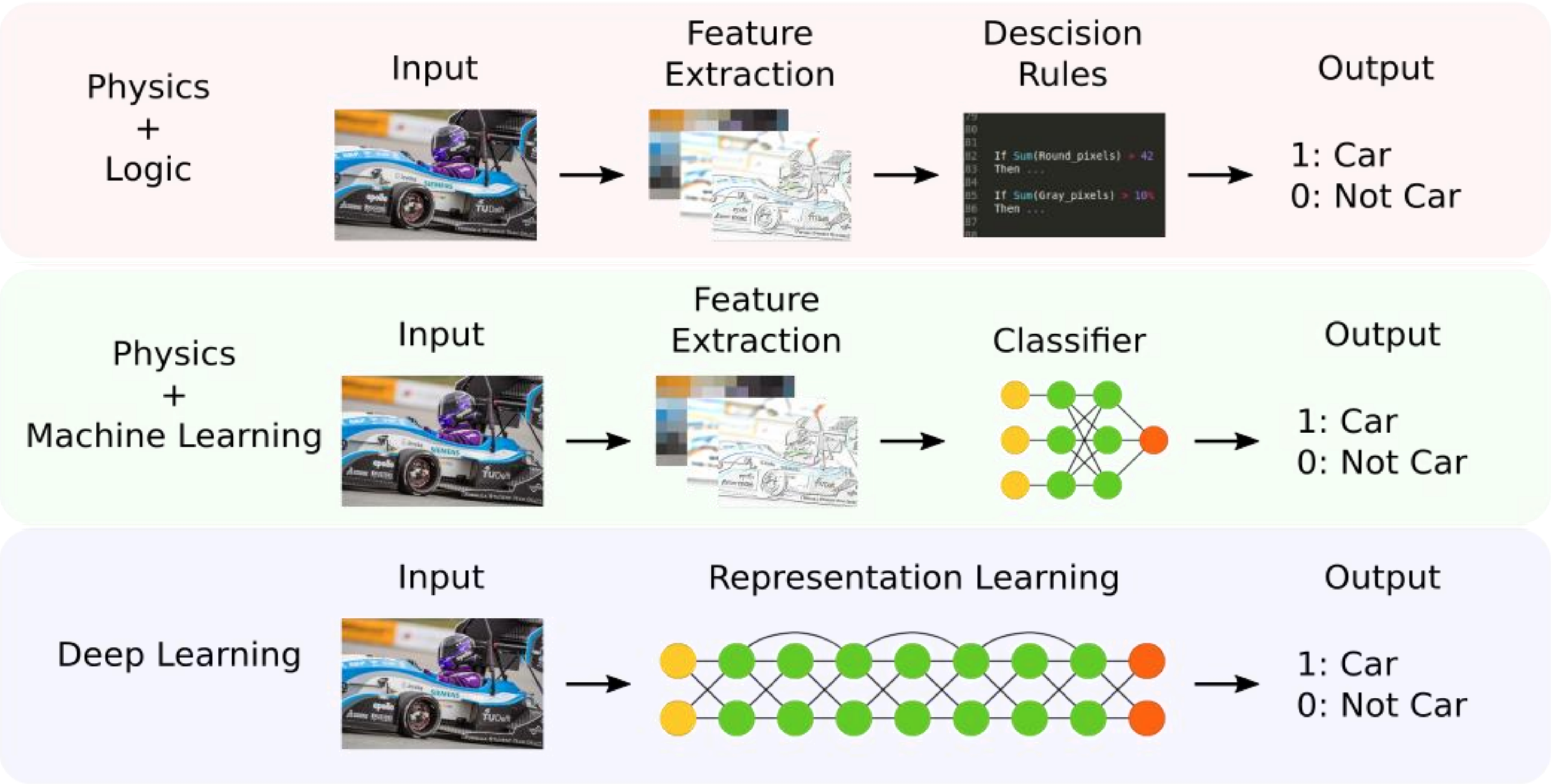
Video credits: “Deep video portraits”, Kim et al., 2018



```
DeepLab V3 xception_cityscapes_trainfine (GTX980M) INPUT_SIZE=1539
Prediction time: 412ms (2.4 fps) AVG: 320ms (3.1 fps)
```

Video credits: youtube, K. Majek, 2018

What is Deep Learning?



Physics + Logic

Input → Feature Extraction → Descision Rules → Output

```
If Sum(Round_pixels) > 42
Then ...

If Sum(Gray_pixels) > 10%
Then ...
```

1: Car
0: Not Car

Physics + Machine Learning

Input → Feature Extraction → Classifier → Output

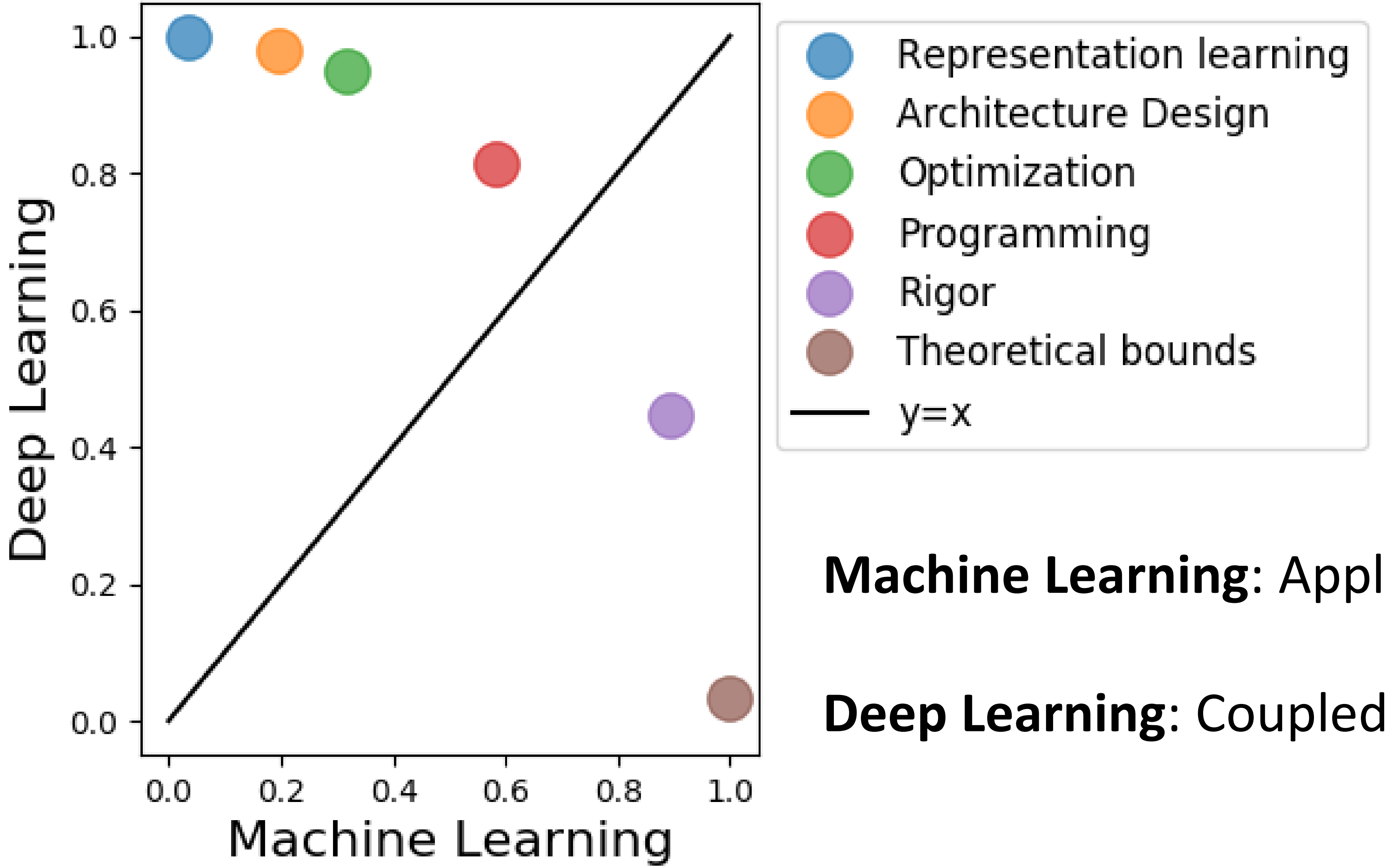
1: Car
0: Not Car

Deep Learning

Input → Representation Learning → Output

1: Car
0: Not Car

Machine Learning vs Deep Learning

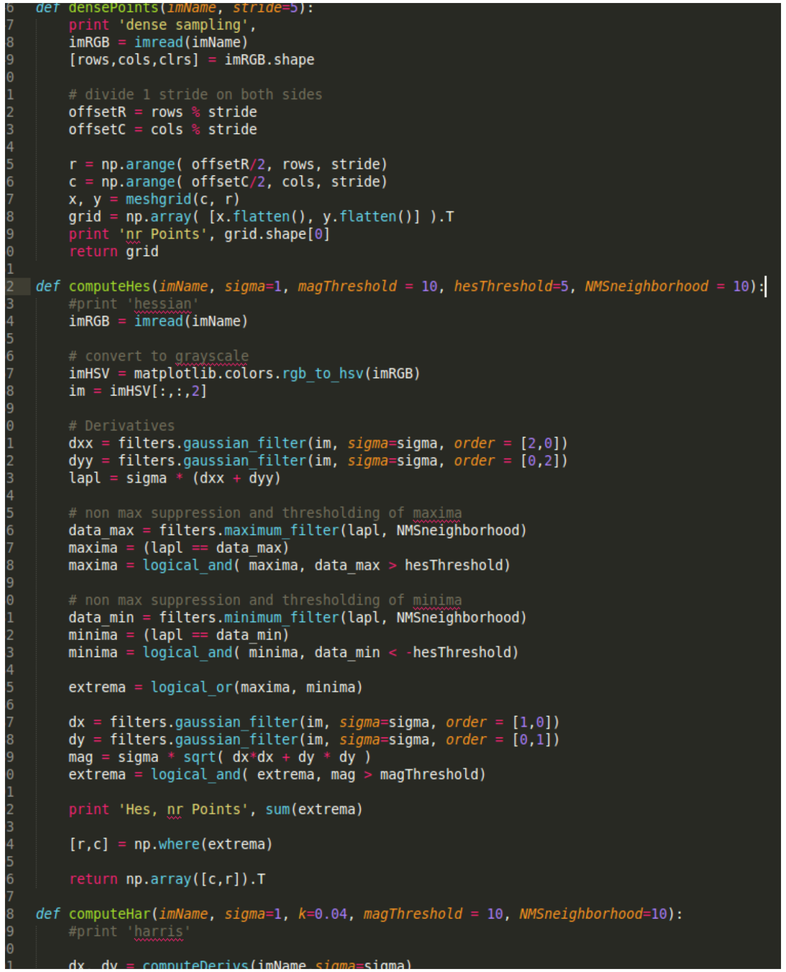


Machine Learning: Application independent

Deep Learning: Coupled to the application

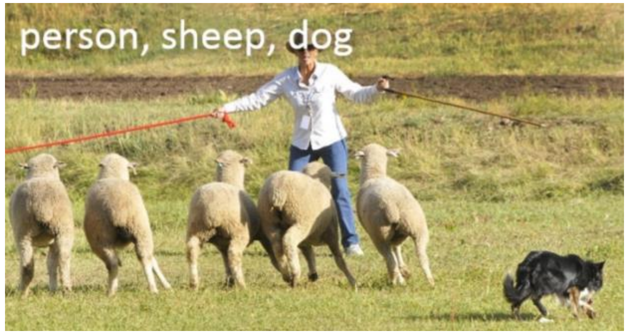
Computer programming by Deep Learning

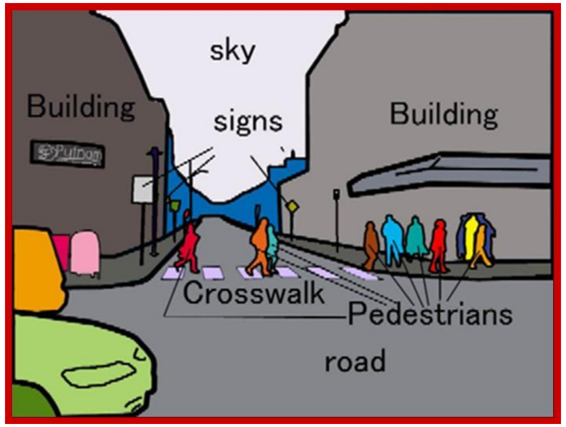
Instead of how to do a task:

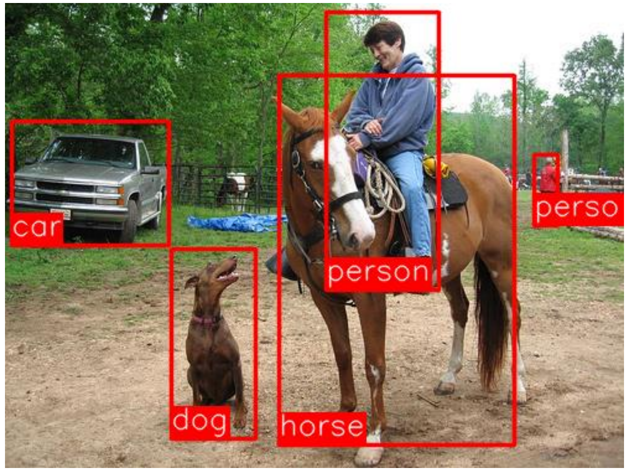


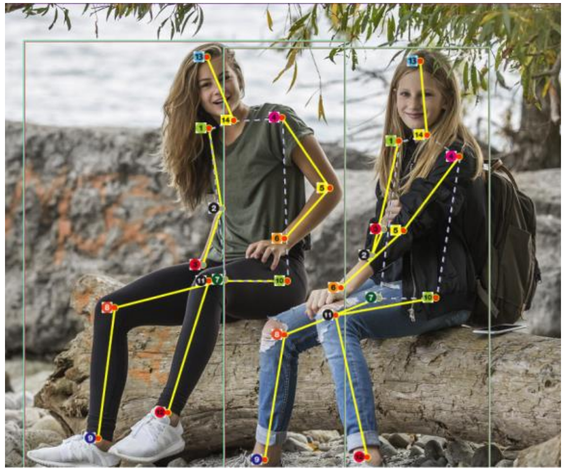
```
def densePoints(imName, stride=5):
    print 'dense sampling',
    imRGB = imread(imName)
    [rows,cols,clrs] = imRGB.shape

    # divide 1 stride on both sides
    offsetR = rows % stride
    offsetC = cols % stride

    r = np.arange( offsetR/2, rows, stride)
    c = np.arange( offsetC/2, cols, stride)
    x, y = meshgrid(c, r)
    grid = np.array( [x.flatten(), y.flatten()] ).T
    print 'nr Points', grid.shape[0]
    return grid

def computeHes(imName, sigma=1, magThreshold = 10, hesThreshold=5, NMSneighborhood = 10):
    #print 'hessian'
    imRGB = imread(imName)

    # convert to grayscale
    imHSV = matplotlib.colors.rgb_to_hsv(imRGB)
    im = imHSV[:,:,2]

    # Derivatives
    dxx = filters.gaussian_filter(im, sigma=sigma, order = [2,0])
    dyy = filters.gaussian_filter(im, sigma=sigma, order = [0,2])
    lapl = sigma * (dxx + dyy)

    # non max suppression and thresholding of maxima
    data_max = filters.maximum_filter(lapl, NMSneighborhood)
    maxima = (lapl == data_max)
    maxima = logical_and( maxima, data_max > hesThreshold)

    # non max suppression and thresholding of minima
    data_min = filters.minimum_filter(lapl, NMSneighborhood)
    minima = (lapl == data_min)
    minima = logical_and( minima, data_min < -hesThreshold)

    extrema = logical_or(maxima, minima)

    dx = filters.gaussian_filter(im, sigma=sigma, order = [1,0])
    dy = filters.gaussian_filter(im, sigma=sigma, order = [0,1])
    mag = sigma * sqrt( dx*dx + dy * dy )
    extrema = logical_and( extrema, mag > magThreshold)

    print 'Hes, nr Points', sum(extrema)

    [r,c] = np.where(extrema)

    return np.array([c,r]).T

def computeHar(imName, sigma=1, k=0.04, magThreshold = 10, NMSneighborhood=10):
    #print 'harris'

    dx, dy = computeDerivs(imName, sigma=sigma)
```

Give examples of what to do:





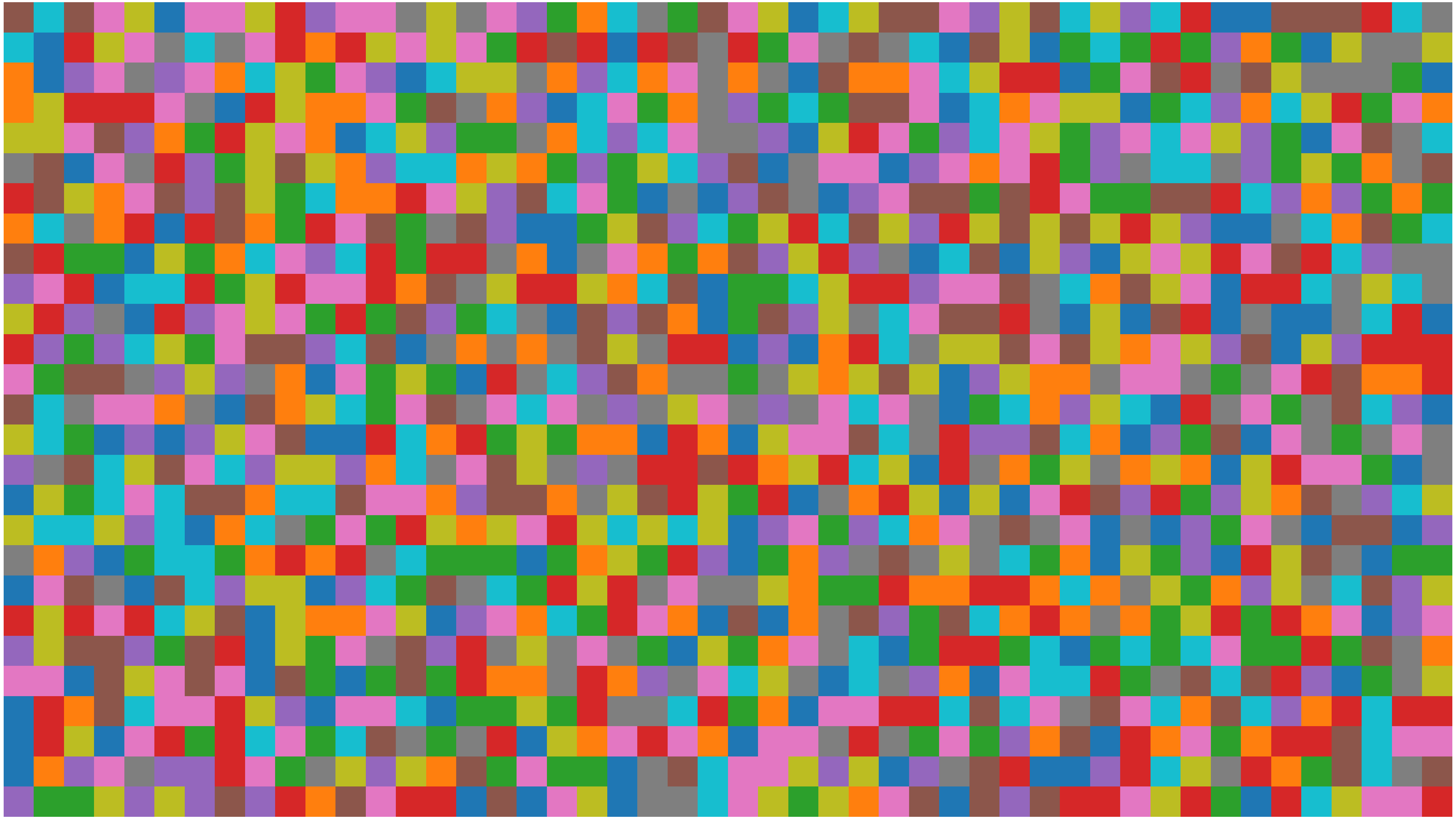




Big Labels

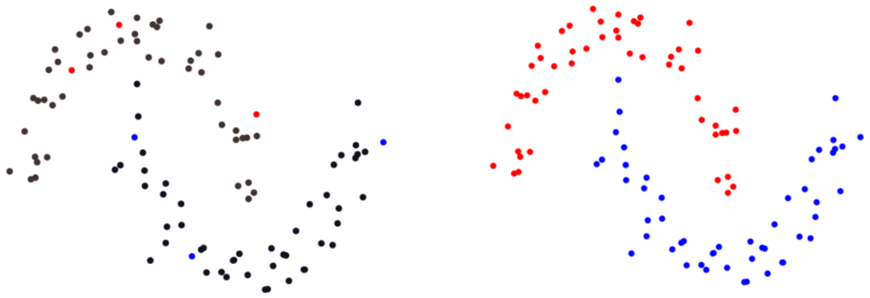
Big Labels

Big Labels

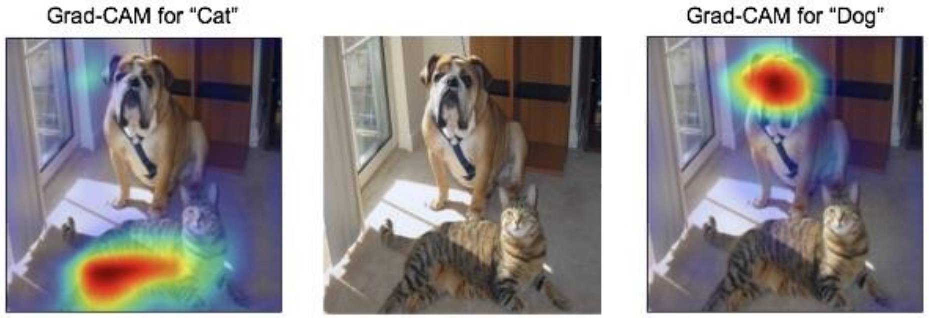


Current research state-of-the art

Reduce annotation effort:

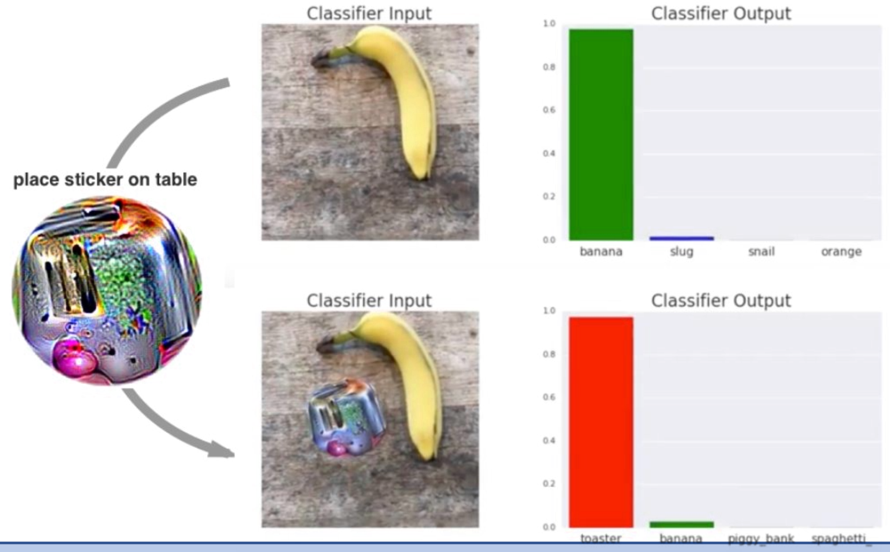


Interpretability:

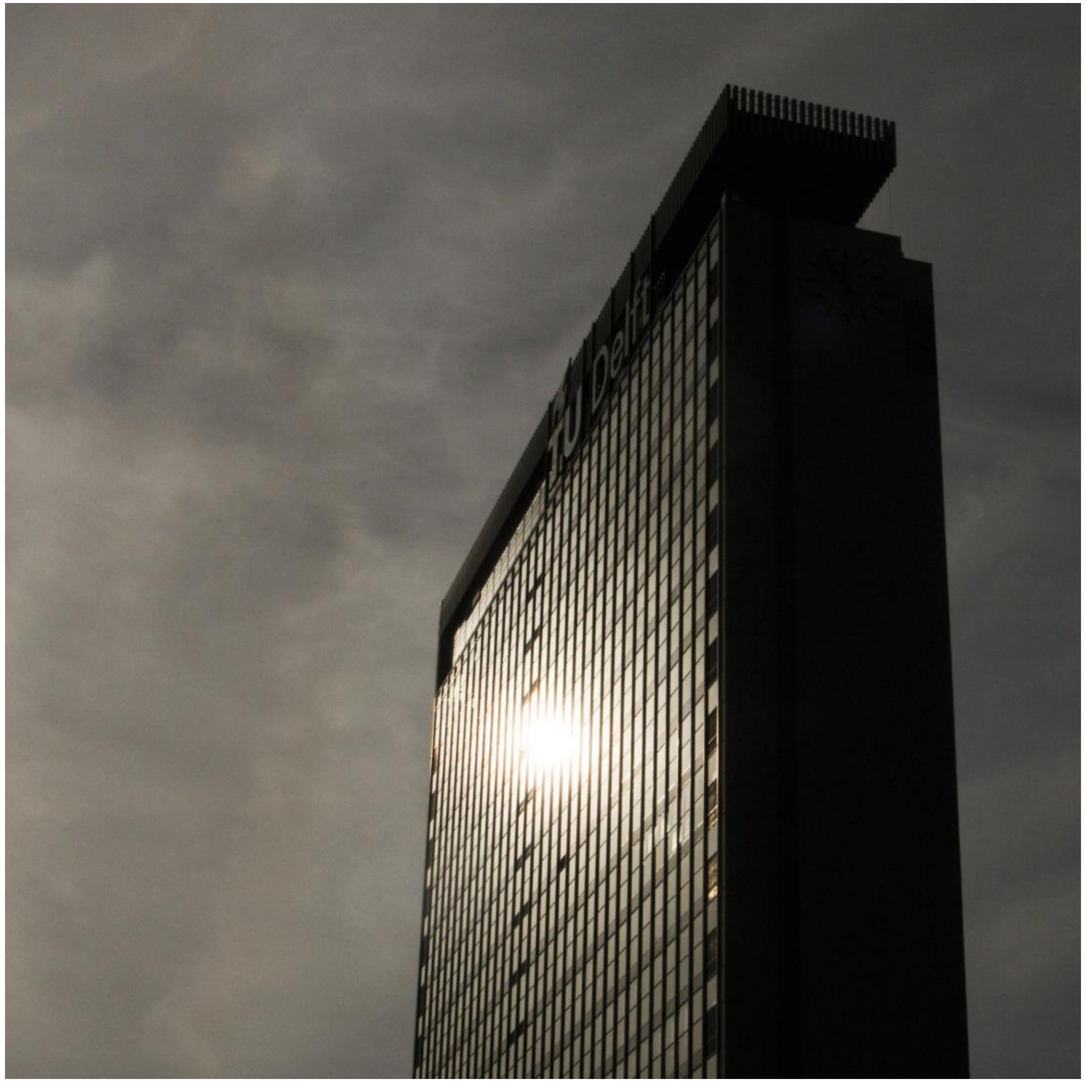


Efficiency:

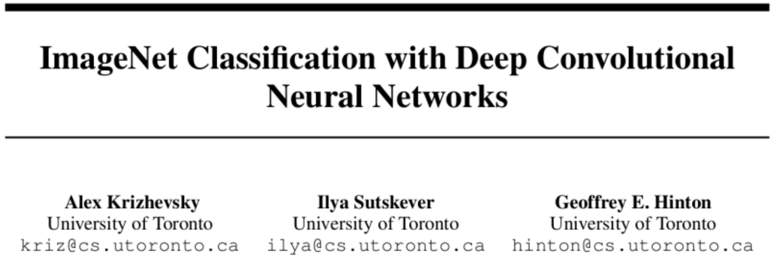
Security against adversarial attacks:



Role of the university



DL revolution came from academia:



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

No products; generic solutions:

$$\mathcal{L} = \mu \sum_{i=1}^{N} \|\mathbf{b}_i - \mathbf{V}\mathbf{y}_i\|_2^2 + \nu\|\mathbf{V}^\intercal\mathbf{V} - A\|_F^2 + \eta\|\mathbf{C} - \mathbf{V}\|_F^2$$

Knowledge powers innovation: