# Computer vision

## introduction and overview of current practice

motivational lecture for the Mozgalo contest

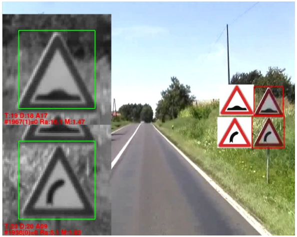
Siniša Šegvić
Faculty of Electrical Engineering and Computing
Sveučilište u Zagrebu

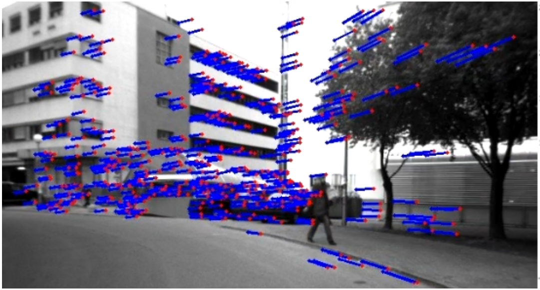
# About: computer vision

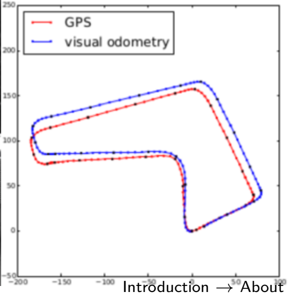
We study techniques to recover useful information from images

Two main approaches towards that goal are:

- **recognition**: *learn* to recover symbolic information, eg. is there an object at (x,y)?
- **reconstruction**: *estimate* numerical information related to geometry, eg. scene structure, camera motion, ...







# About: methods

Both approaches heavily rely on advanced optimization:

- **recognition**: optimize a mapping from images to probabilities
  - criterion: performance on the training data (+ regularization...)
  - optimization performed during training, inference (relatively) simple
  - examples: localization, object classification, image classification
- **reconstruction**: optimize a geometrical model
  - criterion: agreement with the observed image
  - optimization is performed during inference, there is no training
  - often require more than one image

Focus of this talk: **recognition**

A little unfair to reconstruction methods, however:

- Mozgalo targets recognition problems
- recognition methods appear to develop faster today

# About: timeliness

Until recently, image recognition was an academic discipline

- results were reproducible only in carefully controlled environments
- however, the desire to understand machine perception motivated people to carry on

Today, vision by recognition is a technology with clear industrial value

- companies like Google, Facebook, Intel, Microsoft, Nvidia, and many Chinese giants invest billions of dollars each year
- for instance, Google buyed DeepMind for 0.6 B$ (2014)
- for instance, Intel buyed MobileEye for 15.3 B$ (2017)
- NVidia invested 20 M$ in TuSimple (2017)

Let's see what's so hot about vision!

# Applications: autonomous cars


https://www.youtube.com/watch?v=9ydhDQaLAqM

Society of automotive engineers defines 6 autonomy levels (2014)

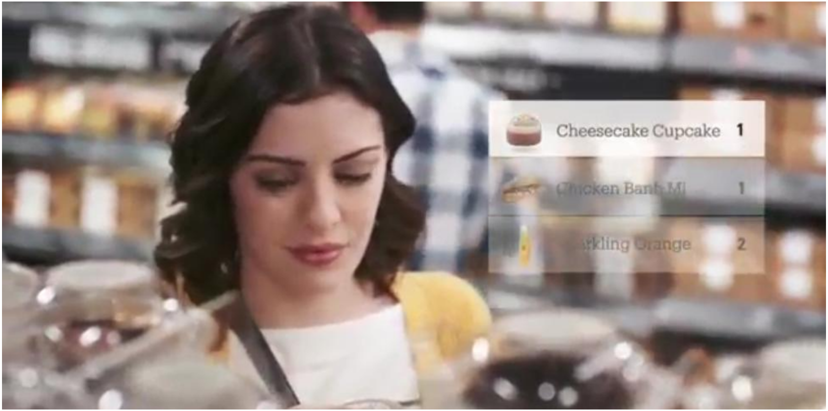
- Tesla autopilot qualifies as level 2 (2016).
- First level-4 vehicles appear: Google, Tesla, Uber, Audi, Renault ...

# Applications: automated diagnostics


http://www.nature.com/nature/journal/v542/n7639/abs/nature21056.html

Esteva et al. Dermatologist-level classification of skin cancer with deep neural networks. Nature 542, 115–118 (02 February 2017)

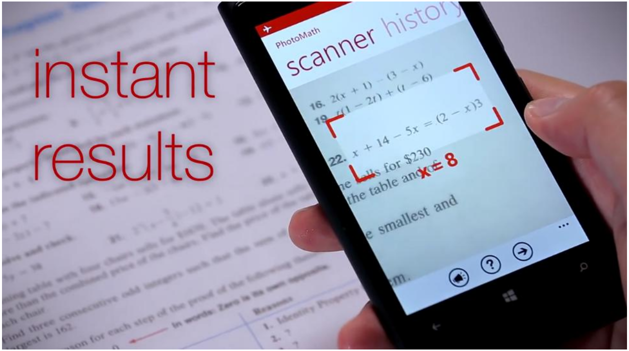
# Applications: automated shop

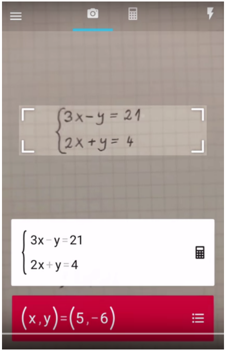



https://www.youtube.com/watch?v=Jg0pQaV44I8

Amazon Go, 2131 7th Ave, Seattle, WA.

# Applications: automated payments



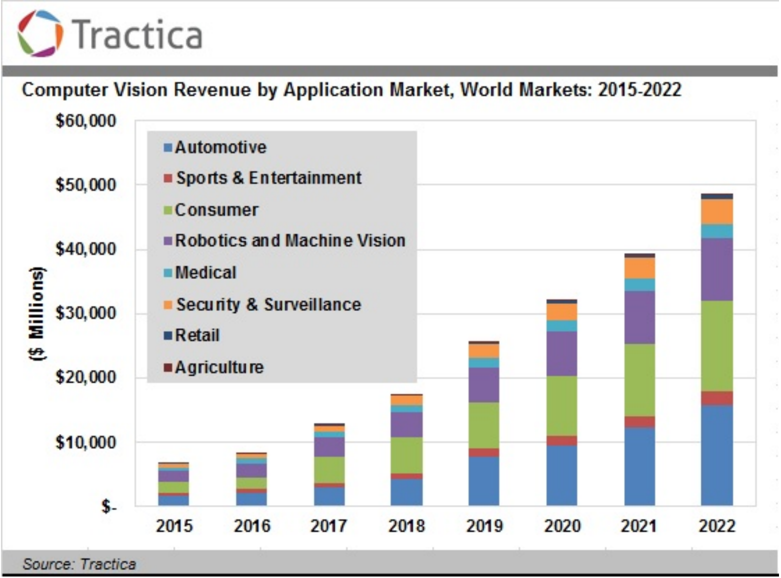




https://www.youtube.com/watch?v=oXn9NuUmhDM

https://www.youtube.com/watch?v=XlbVB50mlh4

...and solutions to mathematical problems (with all steps!)

# Applications: projections and predictions




https://www.tractica.com/

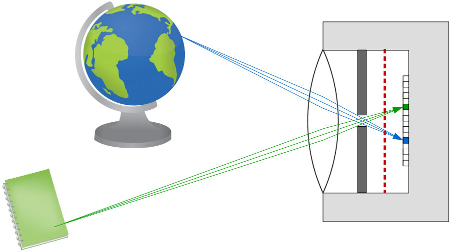
# Applications: agenda

1. Introduction and motivation
   - two basic approaches, applications
2. A look back: classic computer vision
   - on digital images, main milestones
3. The image classification problem
   - approaches, advantages, shortcomings
4. Convolutional models
   - architectures, layers, code
5. Our research
   - overview, datasets, results
6. Hardware support, future trends
7. Conclusion

# Digital images: formation

A digital camera consists of three main elements

1. sensor: a rectangular array of light sensing elements ($\rightarrow$ **pixels**!)
2. aperture: a tiny hole which lets the light arrive onto the sensor
   - it ensures that each sensing element gets excited by the corresponding narrow beam of light.
3. lens: a light collection device (ensures stronger signal)



For color images, we would need a slightly more complex system

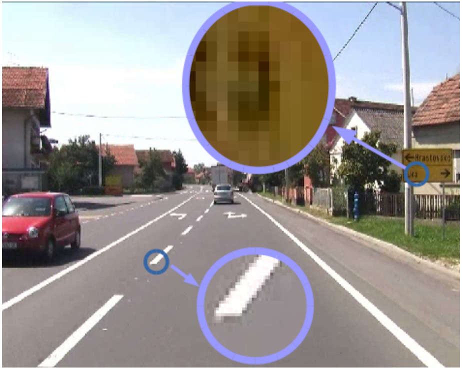
- e.g. three color filters, three sensors and a system of mirrors

# Digital images: pixels

Thus, any digital image is a rectangular matrix of pixels

- dimensions (rows $\times$ columns) depend on the sensor

If we zoom enough into the image, pixels become visible:

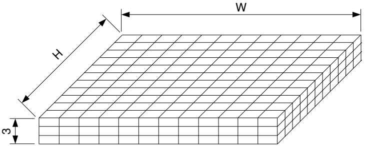


(if they don't - instruct your viewer to refrain from smoothing)

# Digital images: structure

In color images, each pixel is an (R, G, B) triplet.

Such image can be represented with a tensor H$\times$W$\times$3:




A flattened tensor can be fit into memory as an array of H$\cdot$W$\cdot$3 numbers:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $R_{11}$ | $G_{11}$ | $B_{11}$ | $R_{12}$ | $G_{12}$ | $B_{12}$ | ... | ... | ... | $R_{HW}$ | $G_{HW}$ | $B_{HW}$ |

An image of 200$\times$200 pixels can be described with 120000 numbers.

# A LOOK BACK: INCEPTION (1966)

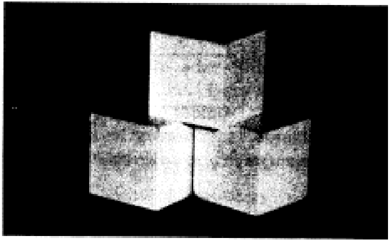
Immediately after digital images appeared, people started to wonder:

- can a computer comprehend the world by analyzing images?

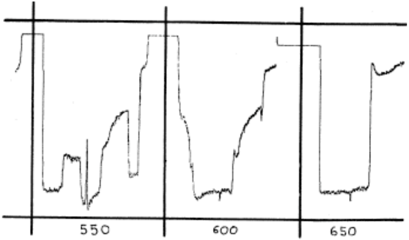
Year 1966: an american professor assigns a student project with a goal to develop a program which prints a description of the input image

The professor was a little in front of the time: the problem had remained unsolved for many years despite diligent work of many people...

Our understanding grew slowly but steadily, main milestones follow



[MIT AI memo 200]

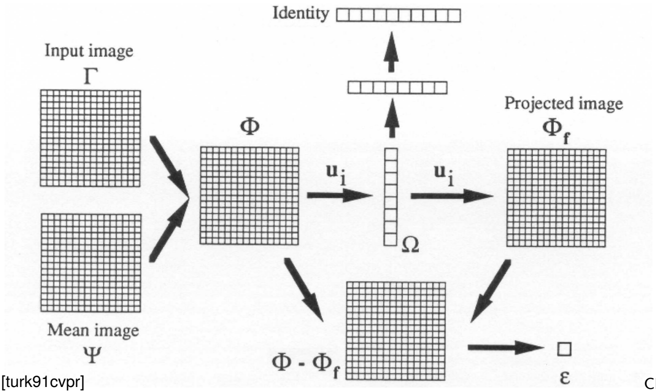



[MIT AI memo 200]

## A look back: object classification (1991)

Approach to classify well-aligned face images [turk91cvpr]:

- treat images as high-dimensional datapoints
- hypothesize that they live on a lower-dimensional manifold
- use training data to learn a projection to the **eigenface** manifold
- perform the recognition task on the manifold representation

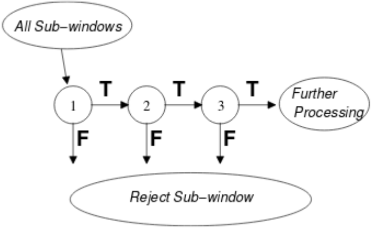


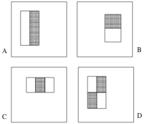

[turk91cvpr]

# A LOOK BACK: OBJECT LOCALIZATION (2001)

Approach to localize objects from a single class [viola01cvpr]:

- treat localization as binary detection in the sliding window
- learn the binary classifier as an attentional cascade
- learn individual levels of the cascade by combining simple features
- could process over 100000 patches at 25 Hz



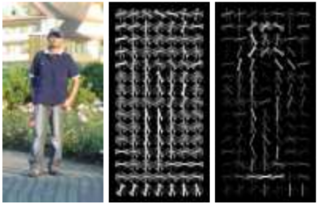




[viola01cvpr]

# A look back: object localization (2005)

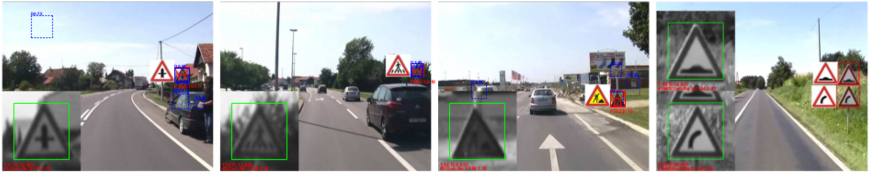
Approach to localize objects from a single class [dalal05cvpr]:

- linear classification in a sliding window
- represent patches with a powerful local descriptor (HOG)
- more accurate than [viola01cvpr] but slower



[dallal05cvpr]

Extension of the last two approaches solved the traffic sign localization:

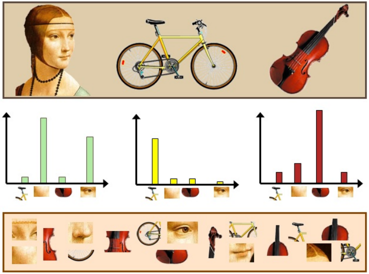


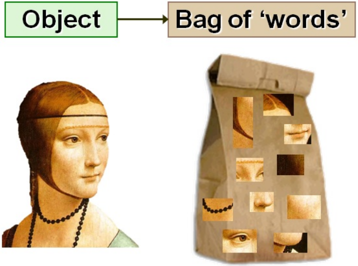
[segvic14mva]

# A look back: image classification (2004)

Approach to classify unaligned images [csurka04eccv]:

- code local patch appearance with hand-crafted features (SIFT)
- learn a visual dictionary of patch descriptors (no supervision)
- represent images as histograms over visual words
- train a shallow model to classify such representations






[fergus09iccv]

# A look back: yesterday

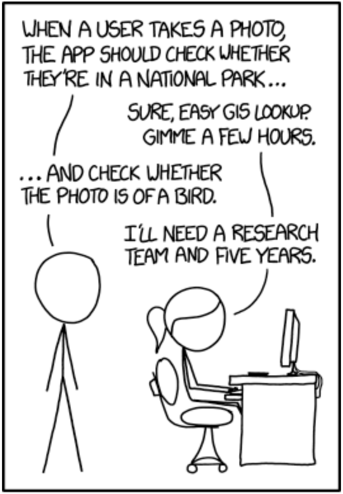
It became wildly known that vision is hard...

2014: popular culture portrays computer vision as a mission impossible...

- finding out whether an image contains a bird?
- well, with a research team and five years - maybe...

However, we came a long way since 1966

- reliable bird detection was feasible before 2010 (with a research team)
- methodology and tools of today allow to solve such problems at home!



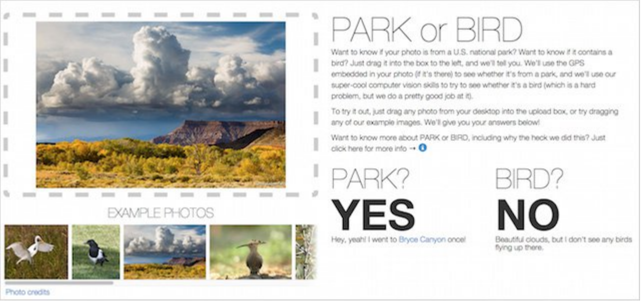

IN CS, IT CAN BE HARD TO EXPLAIN THE DIFFERENCE BETWEEN THE EASY AND THE VIRTUALLY IMPOSSIBLE.

https://xkcd.com/1425 24.09.2014

# A LOOK BACK: TODAY

Thus, a month after XKCD 1425 came out, Flickr proudly reported:

- the "park or bird" problem is solved...
- ...in less than five years :-)




https://code.flickr.net/2014/10/20/introducing-flickr-park-or-bird/

Computer science problems are easy to underestimate and overestimate :-)

# Classification: problem

Image classification is the ultimate recognition problem

Hard becaue we do not know where is the defining object

Especially hard when intra-class variance is large and inter-class variance is small

Consider the problem of discriminating bison from oxen:

[image-net.org]

# CLASSIFICATION: RULE-BASED APPROACH

We could try to solve image classification by a rule-based system:

1. concentrate on image regions projected from four legged animals
2. oxen have longer horns or no horns at all
3. bison are dark brown, etc, etc

It's neat (no learning!), but nobody succeeded to make that work!

[image-net.org]

# Classification: learning

Hence, we look up approaches which are able to learn functionality from the data

A machine learning approach would roughly follow the following steps:

1. express the program with many free parameters
   - the parameters determine a transformation which we call the model
2. fit parameters on the training set
3. evaluate performance on the test set

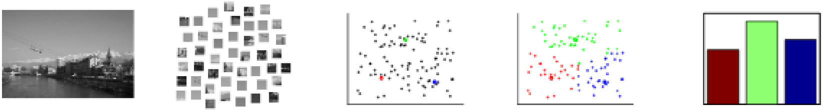
Success depends on the model, training set and processing power

1. model may have insuficient (or excessive) capacity
2. the training set may be too small or not representative enough
3. insufficient processing power $\Rightarrow$ the training may not converge

# Classification: bag of words

Early image classification approaches (BOW) had three layers:

1. describe patches with hand-crafted feature descriptors
2. learn a visual dictionary (a collection of visual words)
3. map patches into visual words, and aggregate them into a histogram
4. classify image descriptors with a shallow model
5. can be viewed as a kernel approach with explicit embedding



Region selection — Region appearance description — Region appearance coding — Image features from region appearance codes

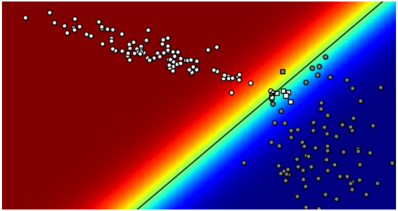
[krapac11phd]

# Classification: shallow classification

Transformation data $\rightarrow$ decision:

1. one linear projection
2. optional squashing non-linearity

An example in 2D: $y_i = \sigma(\mathbf{w}^\top \mathbf{x} + b)$



Advantages of shallow models:

- the best solution is guaranteed and fast
- this is the best approach when classes are linearly separable

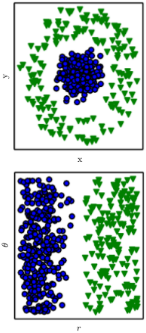
Unfortunately, shallow models are a poor fit for image classification:

- state-of-the-art BOW embeddings are not linearly separable
- insufficient capacity, poor generalization (tendency to overfit)

# Classification: data representation

Classification can profit from good representation:

- it would be easy to discriminate bison from oxen if some magical algorithm converted the input image into a binary vector: [fur?, small horn?, wilderness?, ...]
- most bison would be: [1, 1, 1, ...]
- most oxen would be: [0, 0, 0, ...]




[goodfellow16]

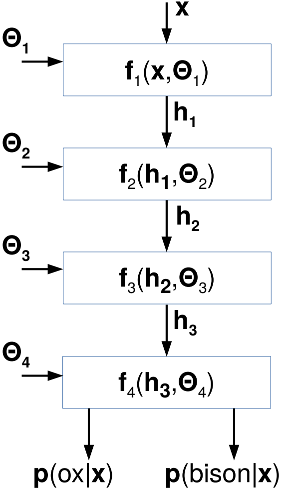
Hand crafting quality representation is hard:

- Greek and Romans did not invent 0 in 1000 years of civilization
- that significantly hindered the development of maths
  - MCMLXXI + XXIX = ?
  - MXXIV : LXIV = ?

Best: learn the representation and the classification simultaneously!

# CLASSIFICATION: COMPOSITIONAL PARADIGM

Deep model: a sequence of learned non-linear transformations




Why deep learning was not successful?

- no guarantee of the learning success
- non-competitive performance

Why deep learning became popular?

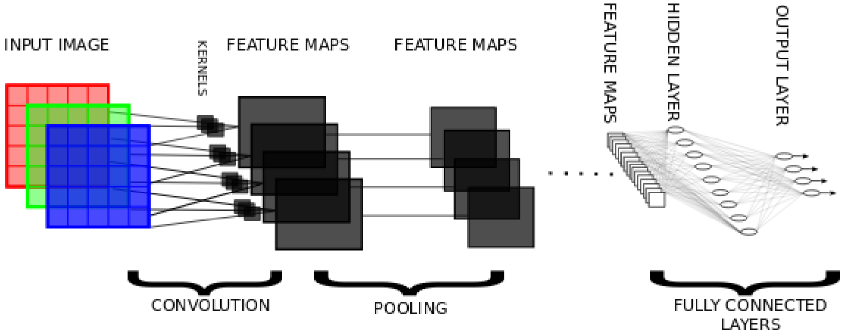
- better modeling and training
- large datasets (n=$10^6$)
- processing power (TFLOPS)

Useful for understanding images, languages, speech, bioinformatics...

# Convolutional models: architecture

Deep models for image classification typically consist of:

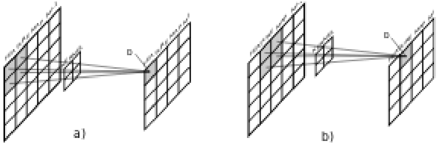
- linear **convolutions** (recognize object parts)
- linear projections (recognize image as a whole)
- poolings (reduce representation dimensionality)
- linear layers followed by an elementwise non-linearity, eg. max(0,x)




[vukotic14ccvw]

# Convolutional models: convolutions

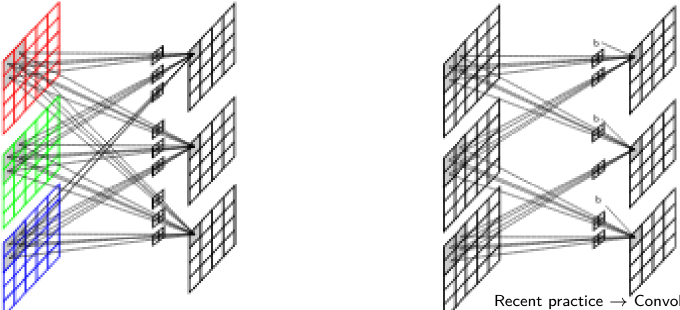
Task: convolve the previous feature map with a kernel




[vukotic14ccvw]

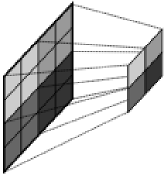
We typically have multiple feature maps on output ⇒ multiple kernels

We typically have several feature maps on input ⇒ kernels are 3D



# Convolutional models: poolings

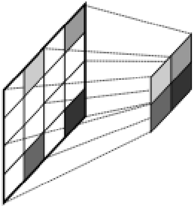
Task: reduce dimensionality to relax memory requirements

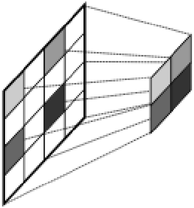


[vukotic14ccvw]

Most often implemented as average pooling or max pooling

It also increases translation invariance:



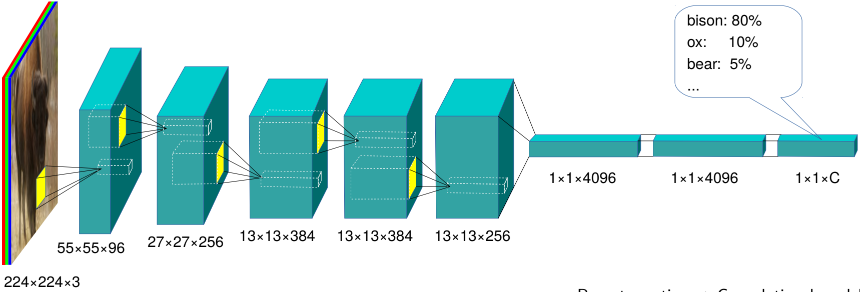


[vukotic14ccvw]

# Convolutional models: large-scale

Deep convolutional model for image classification [krizhevsky12nips]

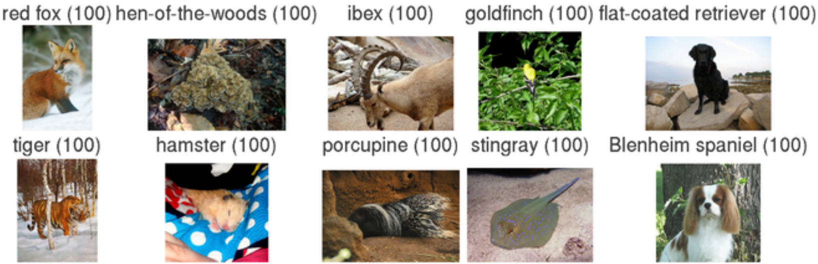
- **input**: image; **output**: distribution over 1000 classes
- **fitness criterion**: average log probability of the correct class
- **structure**: a succession of convolutions and poolings
  - gradual decrease of resolution and increase of the semantic depth
- recent architectures: O($10^2$) layers, O($10^6$) parameters, O($10^9$) multiplications for a 224x224 image!



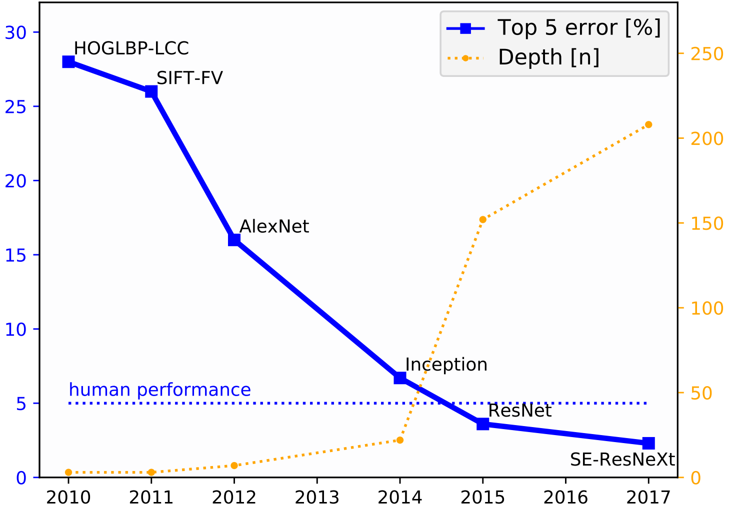
# CONVOLUTIONAL MODELS: IMAGENET

One of the most popular vision datasets [russakovsky15ijcv]

- annual challenges: classification, localization, detection in video
- we focus on the classification challenge: $10^6$ images, $10^3$ classes
- fine-grained animals, objects, materials, sports, dishes...
- evaluation metric: top-five prediction error (trained human: 5%)



# Convolutional models: ImageNet performance



# Convolutional models: ImageNet hard examples

Tasks which are difficult for **humans** [russakovsky15ijcv]:

- fine-grained classification (e.g. 120 breeds of dogs!)
- exotic classes (pulley, spotlight, maypole)

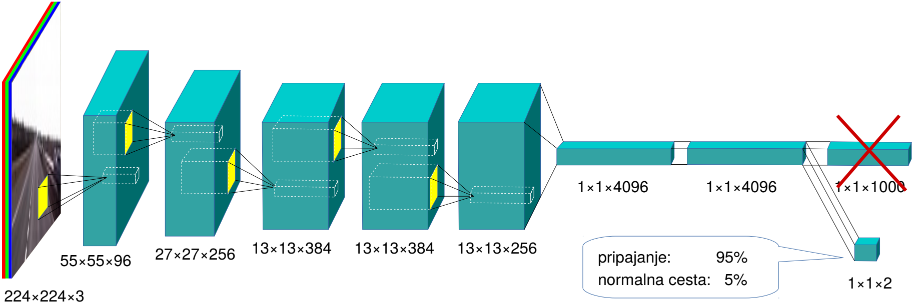
Tasks which are difficult for **GoogleNet** (2014, 6.7%):

- little and thin objects, filtered and atypical images
- abstraction (a toy hatchet, images with text)
- large intra-class variance, small between-class variance

# Convolutional models: knowledge transfer

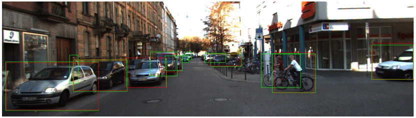
A deep classification model can be fine-tuned for another (easier) task:

- cut-off the last few layers
- connect the remaining layers with the back end for the new task
- train the resulting model on new images
- inherited layers are well-trained so we can train with less data (few thousands images)



# Convolutional models: tasks

- Object localization:
  - detect objects...
  - ...and indicate their location



- Semantic segmentation:
  - label all image pixels...
  - ...with semantic classes

- Stereoscopic reconstruction:
  - label all image pixels...
  - ... with metric distances

- Recognition in video, object tracking, styling and generating images, ...

# Convolutional models: tensorflow

```
class TFLogreg:
  def __init__(self, d, m, param_delta=0.5, param_lambda=1e-3):
    self.X    = tf.placeholder(tf.float32, [None, d])
    self.Yoh_ = tf.placeholder(tf.float32, [None, m])
    self.W    = tf.Variable(tf.zeros([d, m], tf.float32))
    self.b    = tf.Variable(tf.zeros([m], tf.float32))
    self.probs = tf.nn.softmax(tf.matmul(self.X, self.W) + self.b)

    self.truelogprobs = tf.reduce_sum(
        self.Yoh_ * tf.log(self.probs), reduction_indices=1)
    self.cross_entropy = tf.reduce_mean(- self.truelogprobs)

    self.loss = self.cross_entropy
    self.trainer = tf.train.GradientDescentOptimizer(param_delta)
    self.train_step = self.trainer.minimize(self.loss)
    self.sess = tf.Session()

  def train(self, X,Yoh_, param_niter=100):
    self.sess.run(tf.global_variables_initializer())
    for i in range(param_niter):
      loss,_ = self.sess.run([self.loss, self.train_step],
        feed_dict={self.X: X, self.Yoh_: Yoh_})
      if i % 10 == 0:
        print(i, loss)

  def eval(self, X):
    probs = self.sess.run(self.probs, feed_dict={self.X: X})
    return probs
```

# Convolutional models: pytorch

```python
class LogisticRegression(nn.Module):
    def __init__(self, n_input, n_classes):
        super(LogisticRegression, self).__init__()
        self.W = nn.Parameter(torch.rand((n_input, n_classes)))
        self.b = nn.Parameter(torch.rand((1, n_classes)))

    def forward(self, x):
        return torch.matmul(x, self.W) + self.b

data_x = Variable(torch.from_numpy(np.float32(X)).cuda())
data_y = Variable(torch.from_numpy(np.int64(Y)).cuda())

model = LogisticRegression(D, C)
optimizer = optim.SGD(model.parameters(), lr=learning_rate)
model.cuda()

for e in range(epochs):
    output = model.forward(data_x)

    loss = nn.CrossEntropyLoss().forward(output, data_y)
    loss.backward()

    optimizer.step()
    optimizer.zero_grad()
```

# Hardware: CPU vs GPU

CPU core (AVX) can dispatch up to 2 FMA instructions / cycle

- 2·2·8 operations @3GHz $\rightarrow$ 100 GFLOPS
- if a CPU has 10 cores - that's 1 TFLOPS

Modern GPUs achieve 10+ TFLOPS in matrix multiplication

- in practice, the advantage is $\times$50 due to better memory bandwidth.

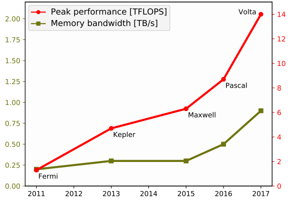
Price of training a simple ImageNet model:

| GPU | TFLOPS | price | time |
|---|---:|---:|---:|
| GTX 1070 | 6.5 | 4000 kn | 52 h |
| GTX 1080 Ti | 11.3 | 7000 kn | 31 h |
| P100 FP16 | 25.0 | 47000 kn | 13 h |
| DGX-1 FP16 | 170.0 | $129000 | 2 h |

# Hardware: GPU

Best performance/price is obtained on gaming GPUs:

- NVIDIA Titan X: 250W, 12GB, 11 TFLOPS, 3000 GPU cores
- Radeon Vega FE: 300W, 16GB, 13 TFLOPS, 4000 GPU cores
- NVIDIA Titan V: 250W, 16GB, 110 TFLOPS (?), 12nm, 8cm$^2$, 21Gt


Actual non-representative measurement:

- GTX1070 (3500 kn) 60$\times$ faster than E3-1220 v3 (1700kn)

Additional challenge: deliver such performance with low power

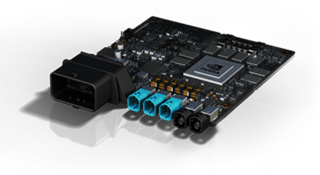
# Hardware: embedded

Novel hardware concept: processing units for artificial intelligence

- fast matrix multiplication (10 TFLOPS)
- low power, low precision (8-32 bit)

Main players:

- NVIDIA TX2: 1.5 TOPS, 15W, 8GB RAM
- NVIDIA Xavier: 20 TOPS, 20W, automotive certificate (ISO 26262)
- Google TPU2: 45 TFLOPS
- Microsoft + Intel DPU (Stratix-10): 40 TFLOPS



# Our research: semantic segmentation

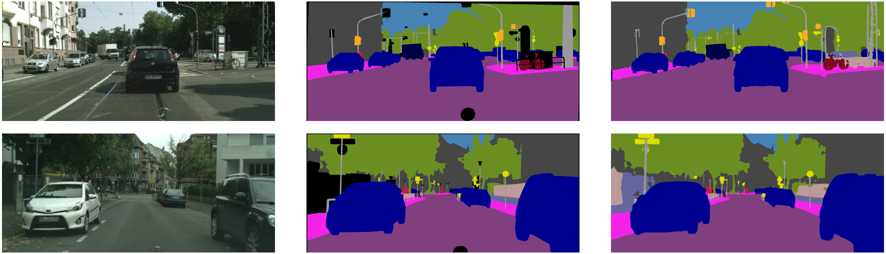
Pixel-level image understanding (or semantic segmentation):

- pixel-level: associate each pixel with a class
- image understanding: classes have a high-level meaning

  **traffic participants**: person (red), car (blue), bicycle (dark red)

  **objects**: pole (light grey), traffic sign (yellow), traffic light (orange)

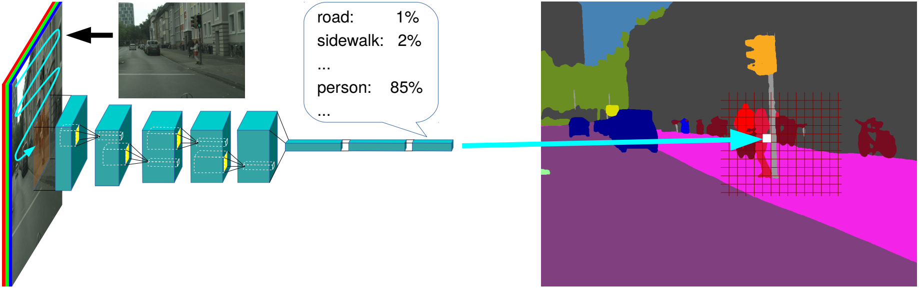
  **landscape**: road (purple), sidewalk (pink), building (dark grey), vegetation (dark green), terrain (light green), sky (light blue)



# Our research: return of the sliding window

A classification model can be applied to the segmentation task:

- analyze the image in the **sliding window** fashion
  - each patch produces one pixel of the semantic map
- segmentation groundtruth allows end-to end training
- each pixel becomes one component of the fitness criterion
- optimized implementation required in practice
  - $10^6$ pixels $\times$ $10^9$ multiplications?

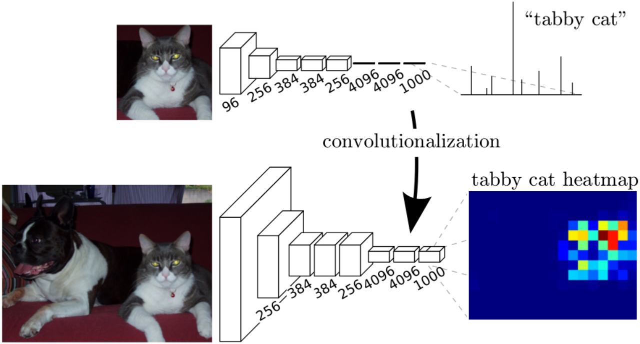


# Our research: going fully convolutional

Luckily, the processing of neighbouring patches involves calculating many common latent activations

**More efficient**: perform the classification layer-wise [long15cvpr]:

- the resulting semantic map is subsampled due to pooling
- this can be relaxed to some extent with dilated filtering [yu16iclr]

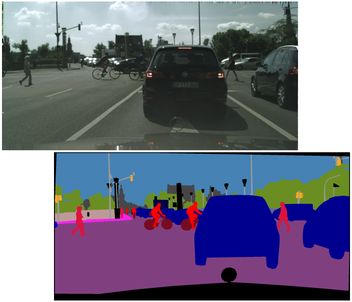



[long15cvpr]

# Our research: datasets

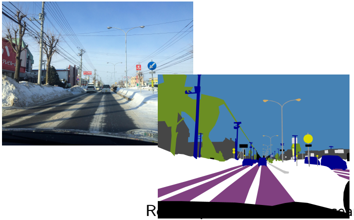
Cityscapes [cordts16cvpr]

- driver's perspective, 19 classes
- 5000 stereo images, 2MPixel
- 20000 coarsely annotated images
- instance level annotations
- 50 cities, spring to autumn



Vistas [neuhold17iccv]

- driver's perspective, 100 classes
- 25000 images, 2-8 MPixel
- instance level annotations
- worldwide, various weather



# Our research: performance characterization

Widely used performance metric: intersection over union (IoU)

- set A: groundtruth pixels of class c
- set B: predicted pixels of class c
- $\text{IoU}_X = |A \cap B| \;/\; |A \cup B|$


Typically, the performance is expressed as mean IoU over all classes

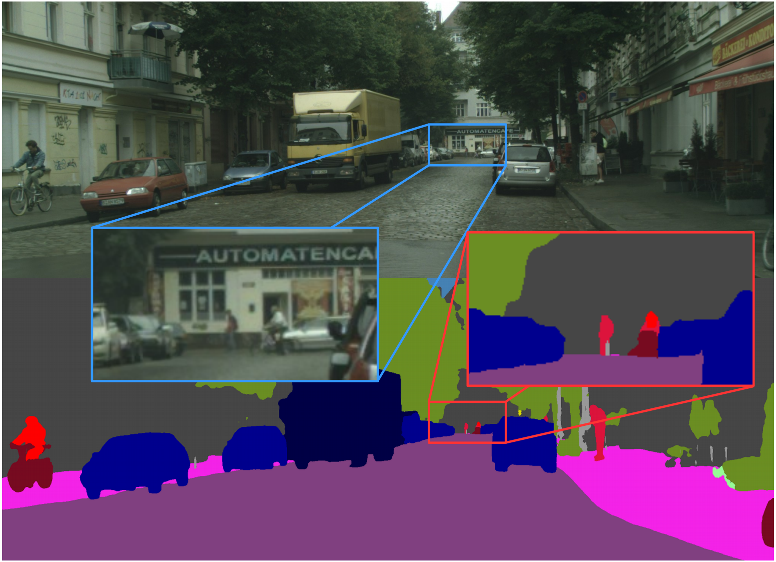
- $\text{mIoU} = \frac{\sum_c \text{IoU}_c}{C}$
- this increases the influence of rare classes with few training pixels
- examples: wall, fence, pole, bottle, potted plant

To ensure integrity, labels of test subsets are withheld from public datasets

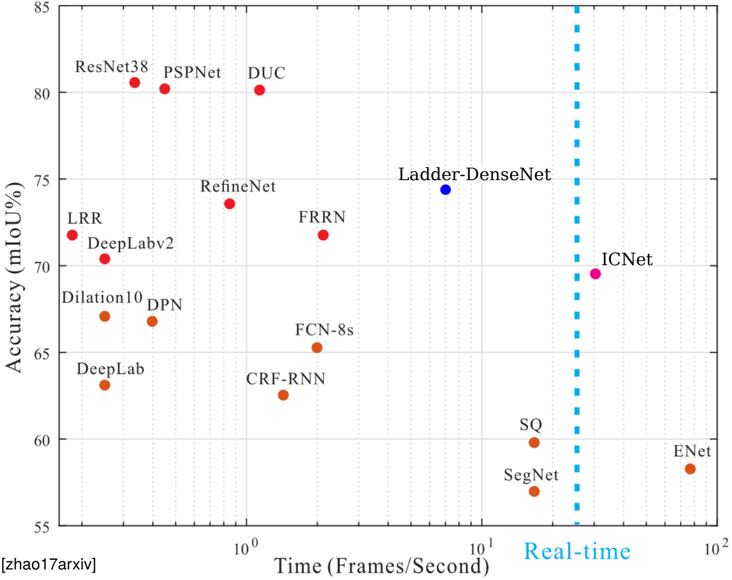
The performance on the test set is determined by submitting results to the evaluation server

# Our research: case study

A pedestrian and a cyclist correctly recognized by 6 out of 7 models:



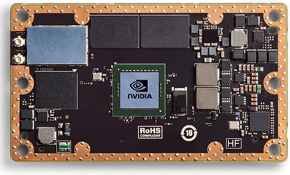
# Our research: speed vs accuracy



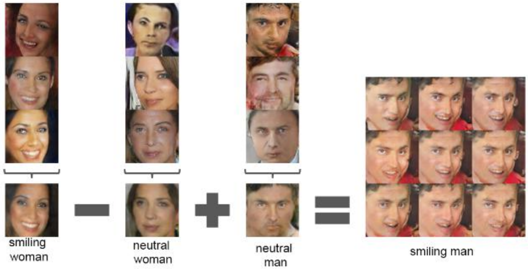

[zhao17arxiv]

# Trends

## Embedded applications (TX2, Xavier)



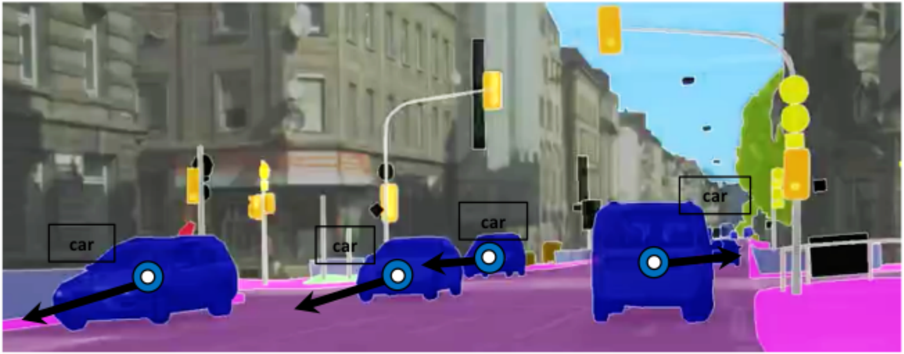

## Unsupervised learning and image generation




[radford16iclr]

# Trends

## Future prediction




[luc17iccv]

# Trends

## Image to text translation

A close up of a person brushing his teeth.

A woman laying on a bed in a bedroom.

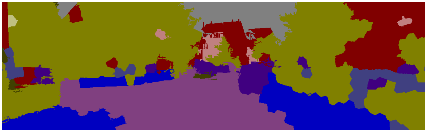
A black and white cat is sitting on a chair.

[donahue17pami]

## Important new work: visual genome dataset

# CONCLUSION: STATE OF THE ART

Dramatic improvement of prediction accuracy in the last few years:

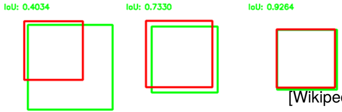


2015 [ros15wacv]


2017 [kreso17cvrsuad]

Cityscapes categories: 91% vs 89.7%

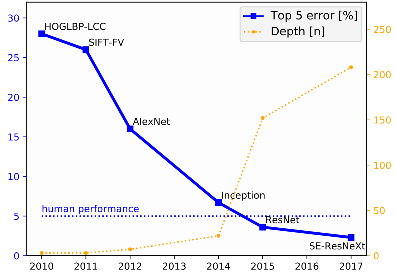
Cityscapes classes: 81.4% vs 78.4 %

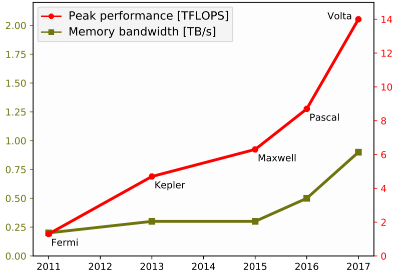



[Wikipedia]

# Conclusion: outlook

Performance of computer vision systems will continue to grow



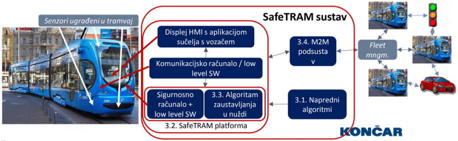

Important research directions:

- relaxing supervision (unsupervised, semi, weakly)
- estimating prediction uncertainties (adversarial examples)
- learning architectures for visual recognition
- forward pass on low power devices (quantization, distillation)

# Conclusion: discussion

## Thank you for your attention!

## questions?




SAFETRAM

Our results have been achieved on projects MultiCLoD (HRZZ I-2433-2014) [1] and SafeTram (Končar i ERDF, 2017-2020) [2]:

[1] `http://multiclod.zemris.fer.hr`

[2] `https://www.koncar-institut.hr/en/content-center/projects/safetram`

Ivan Krešo will defend his thesis subject on January 31 at 15h in D306.

- please let me know if you plan to come so I can arrange a larger room

We will be hiring a PhD student in July-September 2018

- feel free to let me know if you are interested!

# Conclusion: bibliography


[turk91cvpr] Matthew A. Turk, Alex Pentland: Face recognition using eigenfaces. CVPR 1991: 586-591.

[viola01cvpr] Paul A. Viola, Michael J. Jones: Rapid Object Detection using a Boosted Cascade of Simple Features. CVPR (1) 2001: 511-518.

[dalal05cvpr] Navneet Dalal, Bill Triggs: Histograms of Oriented Gradients for Human Detection. CVPR (1) 2005: 886-893.

[segvic14mva] Sinisa Segvic, Karla Brkic, Zoran Kalafatic, Axel Pinz: Exploiting temporal and spatial constraints in traffic sign detection from a moving vehicle. Mach. Vis. Appl. 25(3): 649-665 (2014).

[csurka04eccv] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, Cédric Bray: Visual Categorization with Bags of Keypoints. ECCVW 2004.

[fergus09icml] Rob Fergus. Visual Object Recognition and Retrieval. ICML 2018 tutorial.

[krapac11phd] Josip Krapac. Image Representations for Ranking and Classification. PhD thesis. Universite de Caen.

[goodfellow16] Ian Goodfellow, Yoshua Bengio and Aaron Courville. Deep Learning. MIT press. 2016.

[vukotic14ccvw] Vedran Vukotić, Josip Krapac, Siniša Šegvic. Convolutional neural networks for Croatian traffic signs recognition. CCVW 2014.

[krizhevsky12nips] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton: ImageNet Classification with Deep Convolutional Neural Networks. NIPS 2012: 1106-1114.

russakovsky15ijcv] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, Fei-Fei Li: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision 115(3): 211-252 (2015).

[long15cvpr] Jonathan Long, Evan Shelhamer, Trevor Darrell: Fully convolutional networks for semantic segmentation. CVPR 2015: 3431-3440.

[yi16iclr] Fisher Yu, Vladlen Koltun: Multi-Scale Context Aggregation by Dilated Convolutions. ICLR 2016.

# Conclusion: bibliography (2)


[cordts16cvpr] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, Bernt Schiele: The Cityscapes Dataset for Semantic Urban Scene Understanding. CVPR 2016: 3213-3223.

[neuhold17iccv] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, Peter Kontschieder: The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. ICCV 2017: 5000-5009.

[zhao17arxiv] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, Jiaya Jia: ICNet for Real-Time Semantic Segmentation on High-Resolution Images. CoRR abs/1704.08545 (2017).

[radford16iclr] Alec Radford, Luke Metz, Soumith Chintala: Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ICLR 2016.

[luc17iccv] Pauline Luc, Natalia Neverova, Camille Couprie, Jakob Verbeek, Yann LeCun: Predicting Deeper into the Future of Semantic Segmentation. ICCV 2017: 648-657.

[donahue17pami] Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, Trevor Darrell: Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. IEEE Trans. Pattern Anal. Mach. Intell. 39(4): 677-691 (2017).

[ros15wacv] Germán Ros, Sebastian Ramos, Manuel Granados, Amir Bakhtiary, David Vázquez, Antonio Manuel López Peña: Vision-Based Offline-Online Perception Paradigm for Autonomous Driving. WACV 2015: 231-238.

[kreso17cvrsuad] Ivan Kreso, Josip Krapac, Sinisa Segvic: Ladder-Style DenseNets for Semantic Segmentation of Large Natural Images. ICCV Workshops 2017: 238-245

[gatys16cvpr] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge: Image Style Transfer Using Convolutional Neural Networks. CVPR 2016: 2414-2423.