Convolutional models for image understanding

principles, challenges and research outlook

Siniša Šegvić
UniZg-FER

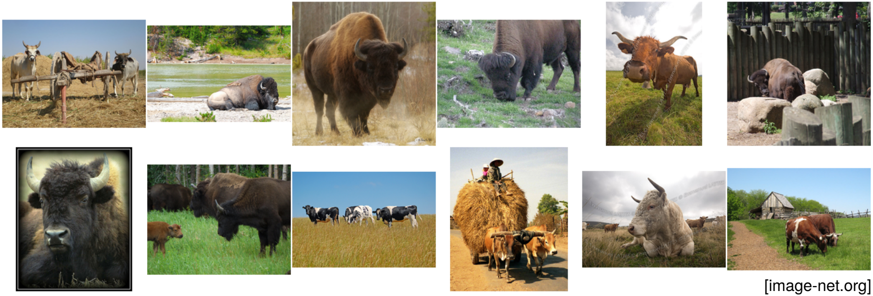
Agenda

- □ Introduction: image understanding
- □ Convolutional models: principles performance
- □ Challenges towards intelligent perception
- □ Overview of our recent research
- □ Conclusions

INTRODUCTION: IMAGE CLASSIFICATION

Story of natural image understanding starts with image classification

Consider the problem of discriminating bison from oxen:



[image-net.org]

Hard because we do not know where is the defining object

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

Introduction: a 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 explainable and neat, but nobody succeeded to make that work!

[image-net.org]

Introduction: learning

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

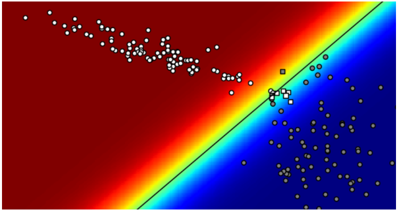
A machine learning approach would roughly follow the following steps:

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 ⇒ the training may not converge

INTRODUCTION: SHALLOW MODELS



Transformation: data → 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

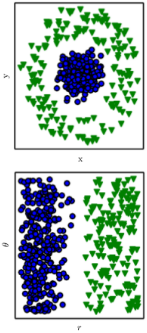
Unfortunately, shallow models are a poor fit for image classification:

- □ the model can only learn a lookup table
- □ applicable only for very simple tasks (eg. 88% on MNIST)
- □ insufficient capacity, tendency to underfit

Introduction: 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 hindered development of science and engineering
  - MCMLXXI + XXIX = ?
  - MXXIV : LXIV = ?

A lot of hard work of smart people went into feature engineering

INTRODUCTION: COMPOSITIONAL PARADIGM

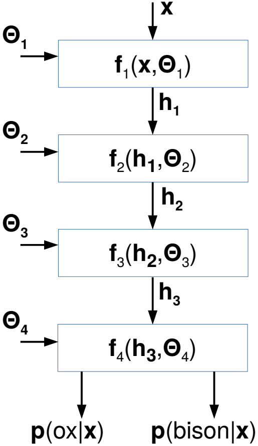
Deep model: a sequence of learned non-linear transformations

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)



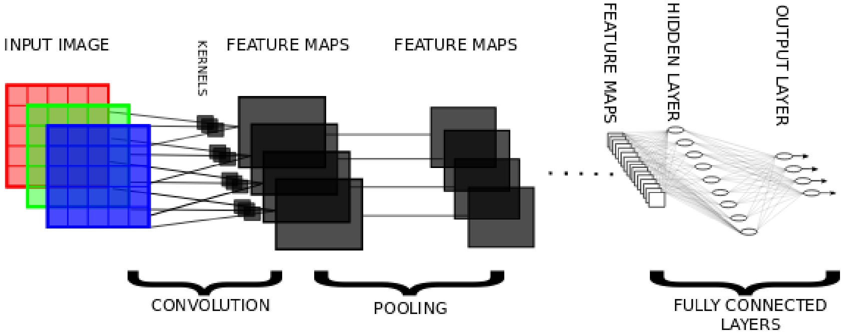
$\mathbf{x}$
Θ_1
$\mathbf{f}_1(\mathbf{x},\Theta_1)$
$\mathbf{h}_1$
Θ_2
$\mathbf{f}_2(\mathbf{h_1},\Theta_2)$
$\mathbf{h}_2$
Θ_3
$\mathbf{f}_3(\mathbf{h_2},\Theta_3)$
$\mathbf{h}_3$
Θ_4
$\mathbf{f}_4(\mathbf{h_3},\Theta_4)$
$\mathbf{p}(\text{ox}|\mathbf{x})$
$\mathbf{p}(\text{bison}|\mathbf{x})$

Suitable for **articulated** data: images, language, speech, bioinformatics...

Convolutional models: architecture

Deep models for image classification typically consist of:

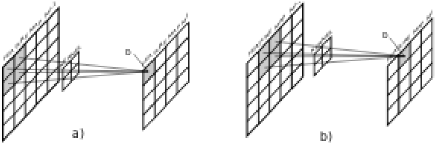
- □ **convolutions** (linear, recognize object parts)
- □ poolings (reduce representation dimensionality)
- □ projections (linear, recognize image as a whole)
- □ elementwise non-linearities, eg. max(0,x)



[vukotic14ccvw]

Convolutional models: convolutions

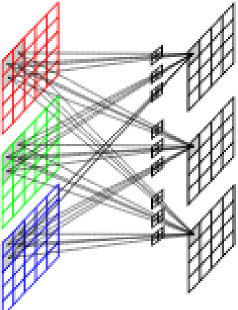
Task: convolve the previous feature map with a kernel

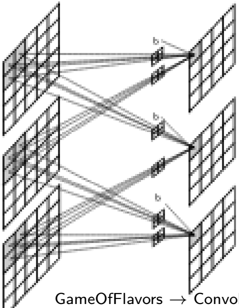


[vukotic14ccvw]

We typically have multiple feature maps on output ⇒ multiple kernels

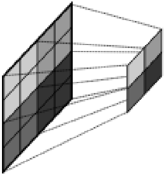
We typically have several feature maps on input ⇒ kernels are 3D





Convolutional models: poolings

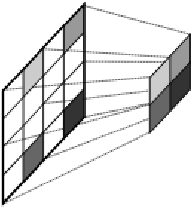
Task: reduce dimensionality to relax memory requirements

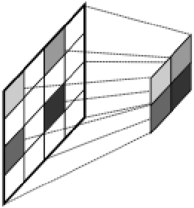


[vukotic14ccvw]

Most often implemented as average pooling or max pooling

It also increases translation invariance:



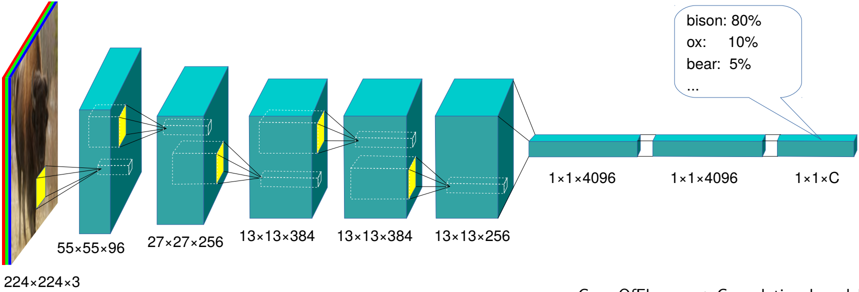


[vukotic14ccvw]

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(10^8) bytes and O(10^9) operations for a 224×224 image!



Convolutional models: Two views

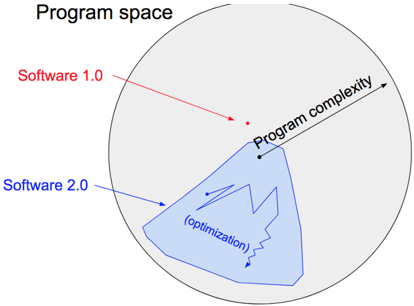
THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

[xkcd 1838]

~~Neural networks~~ →

~~Deep learning~~ →

Differential programming (software 2.0)



[karpathy17medium]

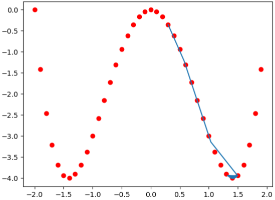
Convolutional models: Diff. programming

```
import torch

step=0.12

x=torch.rand(1, requires_grad=True)
#x=torch.tensor(1.9, requires_grad=True)

for i in range(100):
  y = x**4 - 4*x**2
  y.backward()
  print(x, y, x.grad)
  x.data = x - step*x.grad
  x.grad.zero_()
```

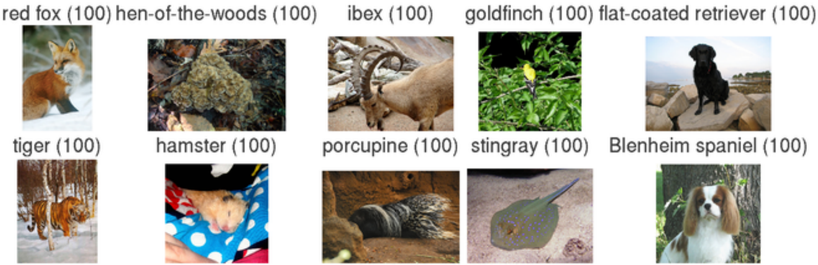


The workhorse algorithm: reverse-mode automatic differentation

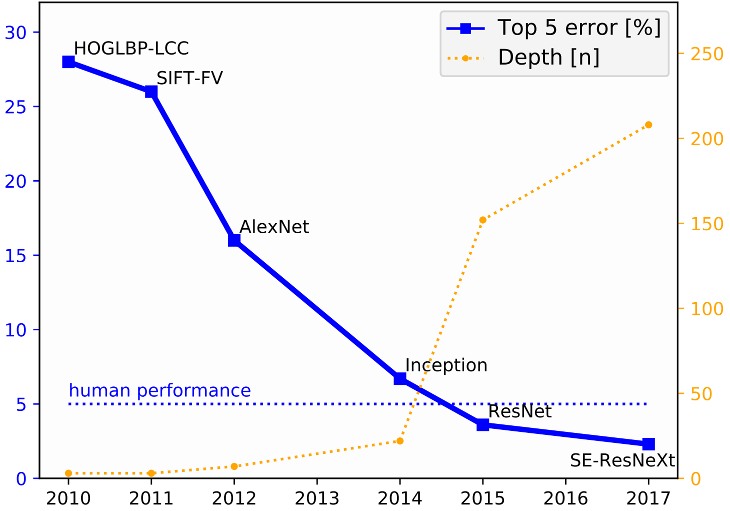
Convolutional models: ImageNet

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



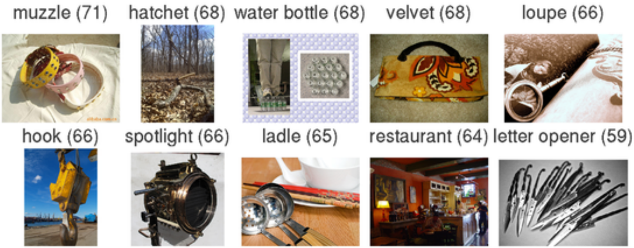
Convolutional models: ImageNet hard examples

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

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

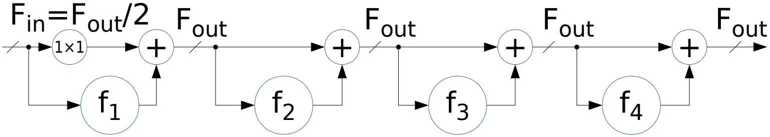
Tasks which are difficult for **GoogleNet** (2014, 6.7%):

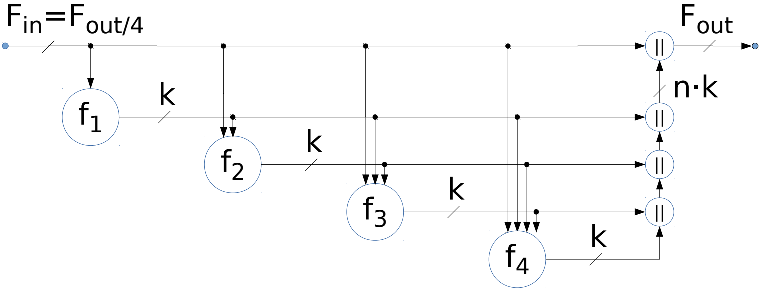
- little and thin objects, filtered and atypical images
- abstraction (a toy hatchet, images with text)
- large intra-class variance, small between-class variance



Convolutional models: further improvement

Important idea: enhance the gradient flow towards early layers

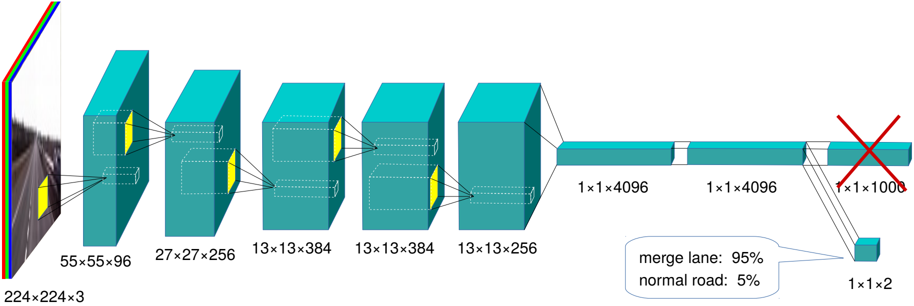




Convolutional models: knowledge transfer

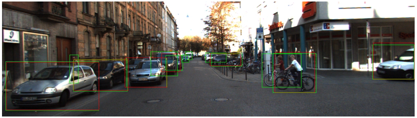
A deep classification model can be fine-tuned for another (easier) task:

- □ 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)

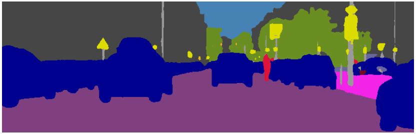


Convolutional models: tasks

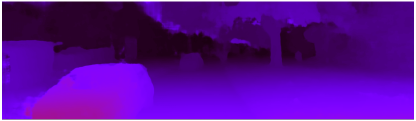
- □ Object localization:
  - □ detect objects...
  - □ ...and indicate their location



- □ 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, ...

Hardware: CPU vs GPU

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

- 2·2·8 operations @3GHz → 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 ×50 due to better memory bandwidth.

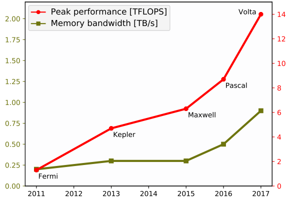
Price of training a simple ImageNet model:

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× faster than E3-1220 v3 (1700kn)

Additional challenge: deliver such performance with low power

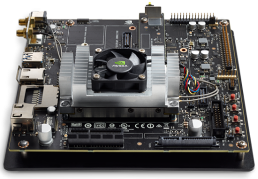
Hardware: embedded

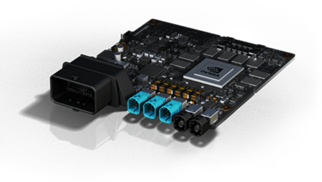
Novel hardware concept: processing units for artificial intelligence

- 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



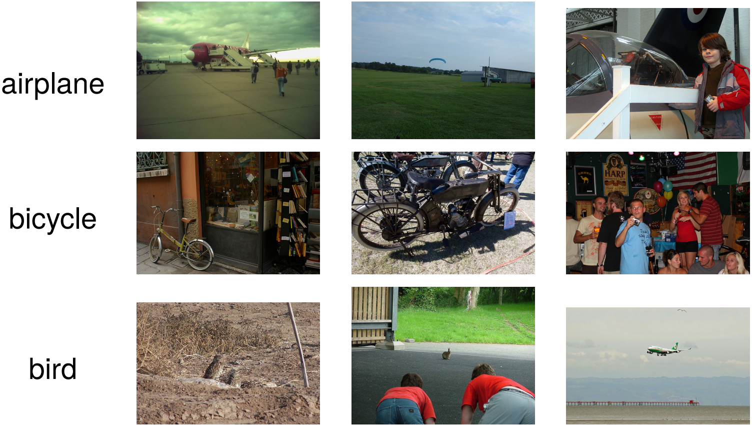


Challenges: false positives due to context

Current models tend to produce false positive detections due to context

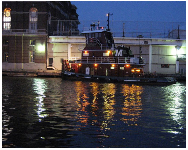
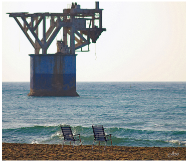
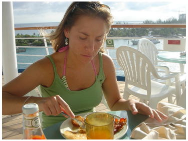
- good performance likely due to recognition of **easy context**

Results on Pascal VOC 2012 (confident TP, high FP, low FN):

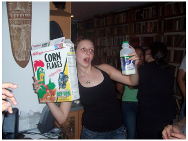


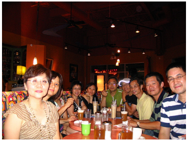
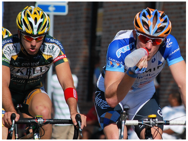
Challenges: false positives due to context (2)

boat

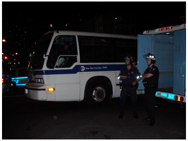
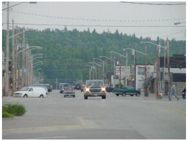
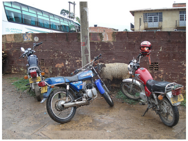




bottle

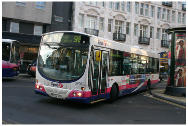
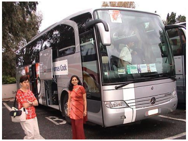
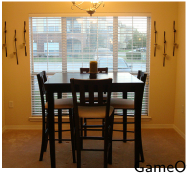


bus

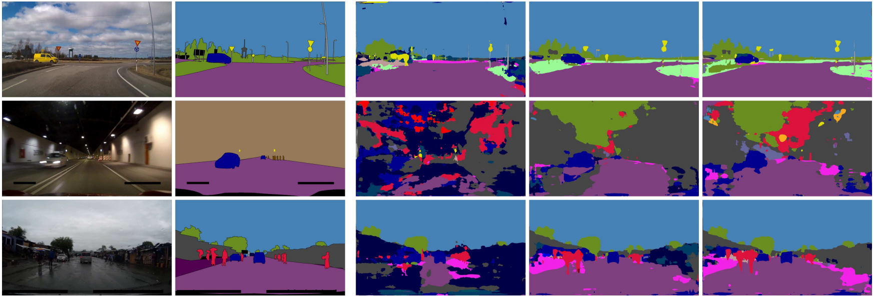
car

Challenges: cross-dataset generalization

Often our models generalize well only within dataset

For instance, images from the novel WildDash dataset (left) fool most models trained on popular datasets such as Cityscapes or Vistas (right)



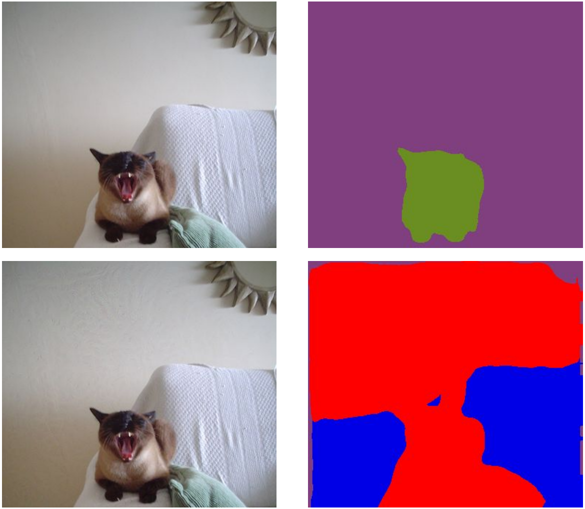
[zendel18eccv]

Conclusion: models tend to overfit to **dataset specifics**

□ camera, weather, environment, climate, ...

Challenges: Adversarial examples

Imperceptible perturbations may invalidate prediction [szegedy14iclr]:



[kreso18ep]

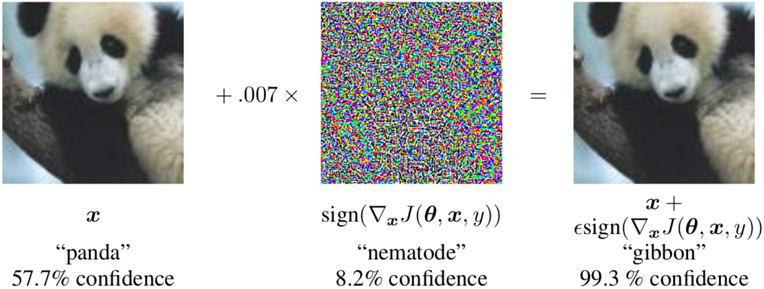
Challenges: Adversarial examples (2)

Adversarial perturbation:

$$\delta = \arg\min_{\delta} p(Y = y_i | \mathbf{x}_i + \delta, \Theta)$$

Adversarial example: $\mathbf{x}_i + \delta$

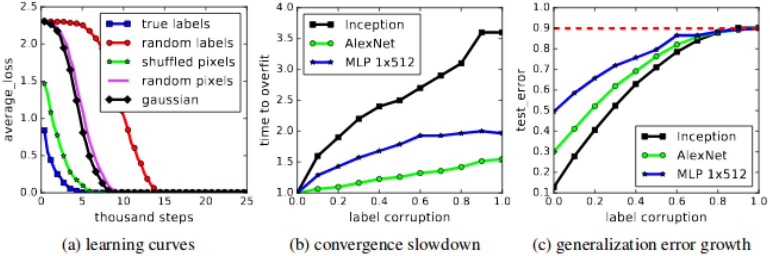
Existence of adversarial examples suggests that current vision systems are free-riding on **easy features** while ignoring the gist of the scene



$+ .007 \times$
$=$
$\boldsymbol{x}$
"panda"
57.7% confidence
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

Challenges: Theory

Effective capacity of deep models is large enough to shatter popular image classification datasets:



(a) learning curves (b) convergence slowdown (c) generalization error growth

[zhang17iclr]

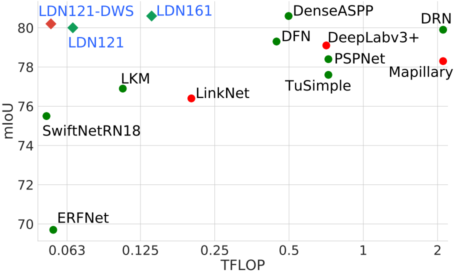
In simple words, the model is able to memorize the entire training data

Yet, the models generalize well when trained on well sorted data

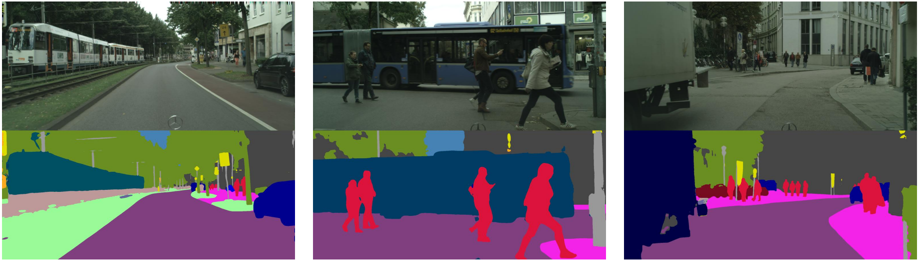
A theory to explain this behaviour is missing.

Results: Efficient semantic segmentation

- □ Goal: pixel-level classification
- □ DenseNet-121 backbone
- □ Lean upsampling path
- □ Memory-efficient training
- □ SotA results with very efficient model

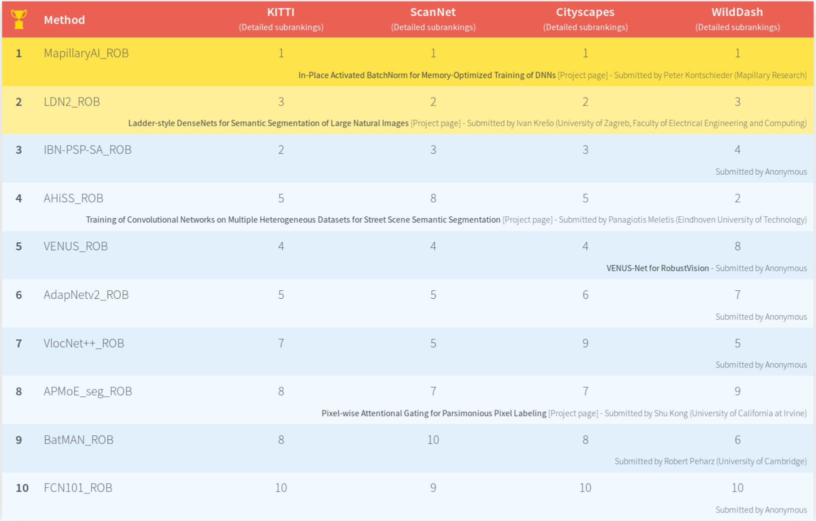


[kreso19review]



Results: Robust Vision Challenge 2018

□ 2nd place, 4 datasets, mutiple domains: driving and indoor



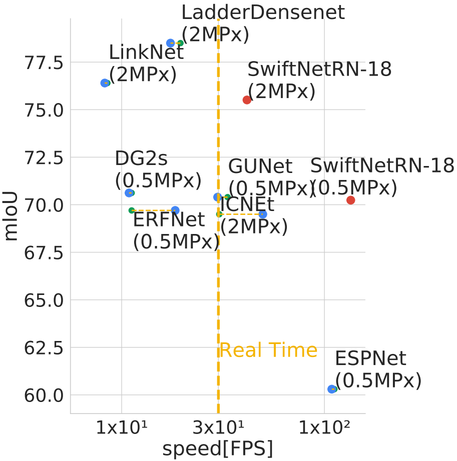
🏆	Method	KITTI (Detailed subrankings)	ScanNet (Detailed subrankings)	Cityscapes (Detailed subrankings)	WildDash (Detailed subrankings)
1	MapillaryAI_ROB **In-Place Activated BatchNorm for Memory-Optimized Training of DNNs** [Project page] - Submitted by Peter Kontschieder (Mapillary Research)	1	1	1	1
2	LDN2_ROB **Ladder-style DenseNets for Semantic Segmentation of Large Natural Images** [Project page] - Submitted by Ivan Krešo (University of Zagreb, Faculty of Electrical Engineering and Computing)	3	2	2	3
3	IBN-PSP-SA_ROB Submitted by Anonymous	2	3	3	4
4	AHiSS_ROB **Training of Convolutional Networks on Multiple Heterogeneous Datasets for Street Scene Semantic Segmentation** [Project page] - Submitted by Panagiotis Meletis (Eindhoven University of Technology)	5	8	5	2
5	VENUS_ROB **VENUS-Net for RobustVision** - Submitted by Anonymous	4	4	4	8
6	AdapNetv2_ROB Submitted by Anonymous	5	5	6	7
7	VlocNet++_ROB Submitted by Anonymous	7	5	9	5
8	APMoE_seg_ROB **Pixel-wise Attentional Gating for Parsimonious Pixel Labeling** [Project page] - Submitted by Shu Kong (University of California at Irvine)	8	7	7	9
9	BatMAN_ROB Submitted by Robert Peharz (University of Cambridge)	8	10	8	6
10	FCN101_ROB Submitted by Anonymous	10	9	10	10

www.robustvision.net/leaderboard.php?benchmark=semantic

[kreso18cvprw]

Results: Real-time semantic segmentation

- □ Increased receptive field can compensate for model capacity
- □ Novel approach for increasing receptive field: pyramid fusion
- □ SotA real-time segmentation: lightweight ImageNet-pretrained models with increased receptive field

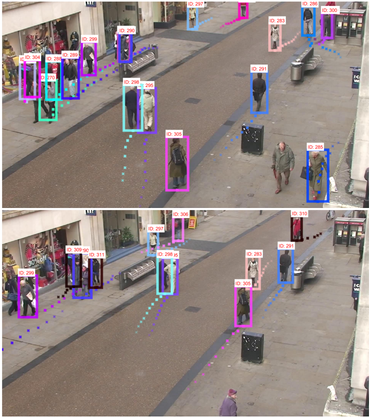


[orsic19cvpr]

[orsic19cvpr]

Results: Pedestrian tracking

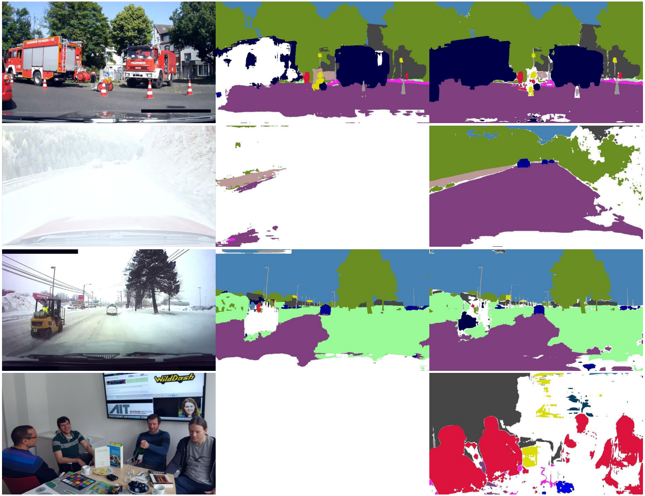
- □ Deep learning for pedestrian detection and appearance representation
- □ Detection: Mask R-CNN ResNet-50 pre-trained on COCO and Cityscapes
- □ Appearance: correspondence embedding based on ResNet-18 trained using angular loss
- □ Tracking: fusion of probabilistic data association and appearance descriptors
- □ Best result on MOT2015-3D



[bicanic19review]

Results: segmentation and outlier detection

- □ Outlier detection = locating unknown concepts in images
- □ Fast evaluation, concurrent with semantic segmentation



[bevandic19review]

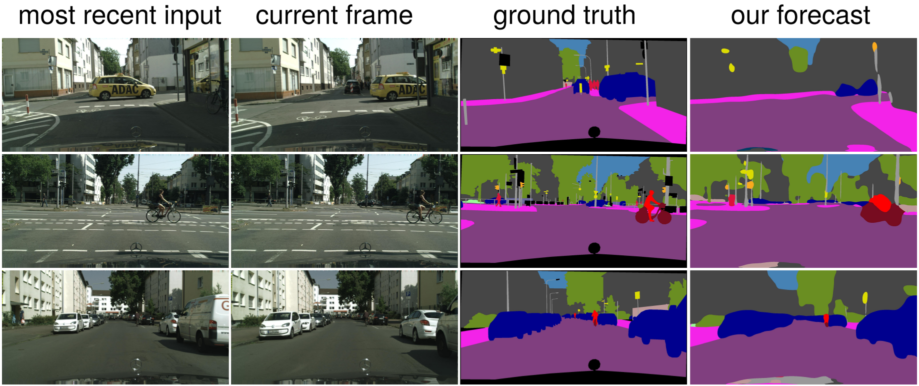
Results: Semantic segmentation and outlier detection

- First place on WildDash benchmark (very difficult images)

Model	Meta Avg	Classic				Negative
	mIoU	mIoU	iIoU	mIoU	iIoU	mIoU
	cla	cla	cla	cat	cat	cla
APMoE_seg_ROB [kong18]	22.2	22.5	12.6	48.1	35.2	22.8
DRN_MPC [yu17]	28.3	29.1	13.9	49.2	29.2	15.9
DeepLabv3+_CS [chen18]	30.6	34.2	24.6	49.0	38.6	15.7
LDN2_ROB [kreso18]	32.1	34.4	30.7	56.6	47.6	29.9
MapillaryAI_ROB [bulo17]	38.9	41.3	**38.0**	60.5	**57.6**	25.0
AHiSS_ROB [meletis18]	39.0	41.0	32.2	53.9	39.3	43.6
LDN169, two heads (ours)	41.8	**43.8**	37.3	58.6	53.3	**54.3**
LDN169, single OE head (ours)	**42.7**	43.3	31.9	**60.7**	50.3	52.8

[bevandic19review]

Results: Semantic forecasting



most recent input

current frame

ground truth

our forecast

[saric19review]

Results: Semantic forecasting

	mIoU	mIoU-MO
Oracle	72.5	71.5
Copy last segmentation	38.6	29.6
Luc Dil10-S2S	47.8	40.8
Luc Mask-S2S	/	42.4
Luc Mask-F2F	/	41.2
Terwilliger	51.5	46.3
Bhattacharyya	51.2	/
Luc F2F (our implementation)	45.6	39.0
our	**52.4**	**48.3**
our (more data)	**53.6**	**49.9**

[saric19review]

Thank you for your attention!

Questions?