# Lightweight convolutional models for real-time dense prediction and forecasting

when less may be more in machine learning

Siniša Šegvić, Marin Oršić, Josip Šarić, Petra Bevandić, Ivan Grubišić
UniZg-FER

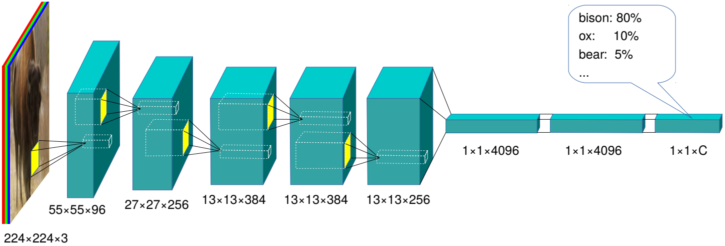
# Agenda

1. Introduction
   - image classification
   - semantic segmentation
2. Convolutional models
   - improved building blocks
   - advanced layers
   - key properties
3. State-of-the-art real-time performance
   - dense semantic prediction
   - dense semantic forecasting
4. Conclusion

# Introduction: image classification

The task: associate input image with one of C predefined classes

State-of-the-art: deep convolutional model, end-to-end training



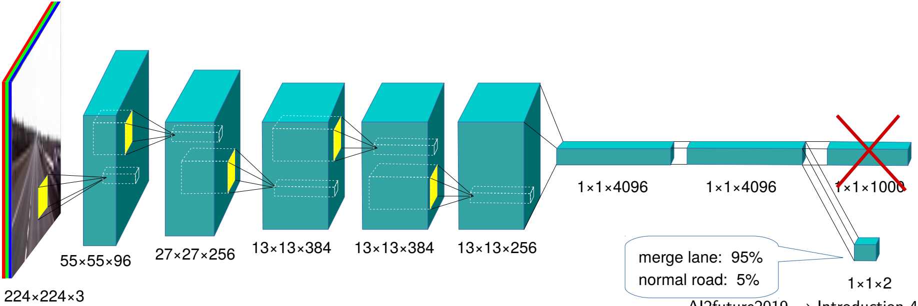

- **input**: image; **output**: distribution over known classes
- **structure**: a succession of convolutions and poolings
  - gradual decrease of resolution and increase of the semantic depth
  - large-scale operation: O($10^3$) classes, O($10^2$) layers, O($10^6$) parameters, O($10^9$) multiplications for a 224x224 image!
- **fitness criterion**: (average) log probability of the correct class

# Introduction: knowledge transfer

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

- train initial model on a large dataset (ImageNet, openimages)
- remove the last few layers (neural surgery?)
- transplant the remaining layers to a back-end for the new task
- fine-tune the resulting model on new images

Fine-tuning may succeed with only a few thousands images



# INTRODUCTION: SEMANTIC SEGMENTATION

Dense visual recognition: associate each pixel with a high-level class

**traffic participants: person, car, truck, bicycle**

**objects: pole, traffic sign, traffic light**

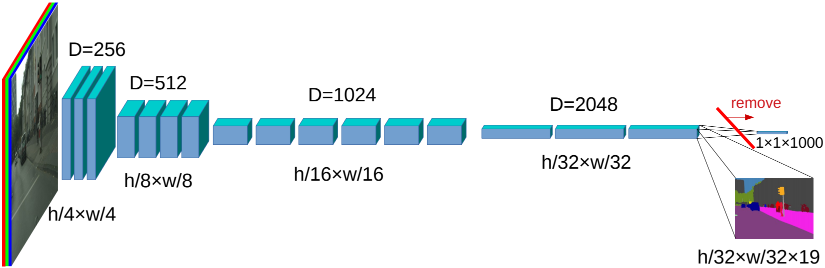
**landscape: road, sidewalk, building, fence, wall, vegetation, terrain, sky**

# Introduction: semantic segmentation (2)

State-of-the-art solutions based on pre-trained classification models

The simplest transition from classification to semantic segmentation:

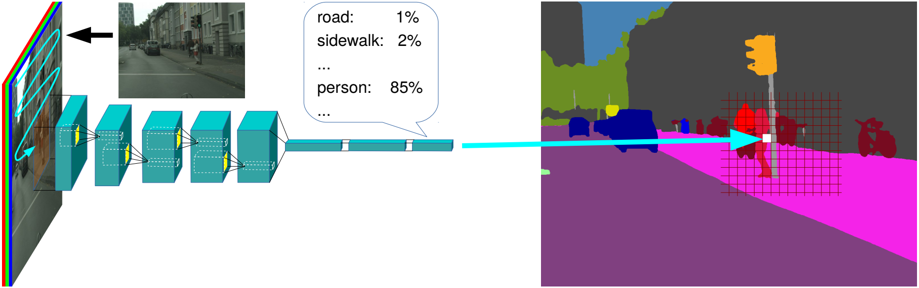
- detach the image-wide classification back-end
- attach a dense prediction layer (eg. 3x3 convolution)
- attach a bilinear interpolation layer to upsample predictions
- attach the pixel-level loss (NLL of the correct class)



# INTRODUCTION: SEMANTIC SEGMENTATION (3)

The described approach equivalent to classifying patches one by one:

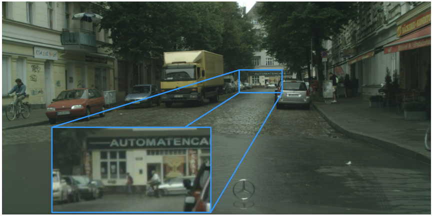
- analyze each image patch in a **sliding window** fashion
- each patch corresponds to one pixel of the semantic map
- each pixel becomes one component of the fitness criterion

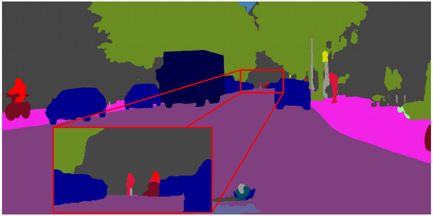


# Introduction: semantic segmentation (4)

Usually we have to deal with large input (and output) resolutions

- we need 1024×2048 images to perceive pedestrians at 200 m

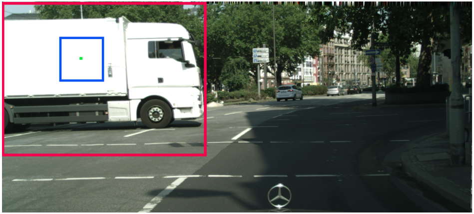




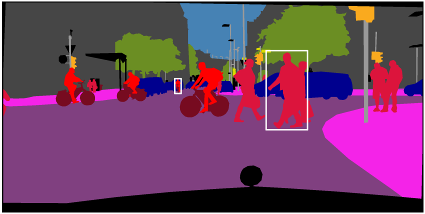
Hence, the following problems are specific to semantic segmentation:

- recognizing smooth surfaces at large objects
- recognizing small objects
- large GPU memory requirements during training
- large computational requirements during evaluation

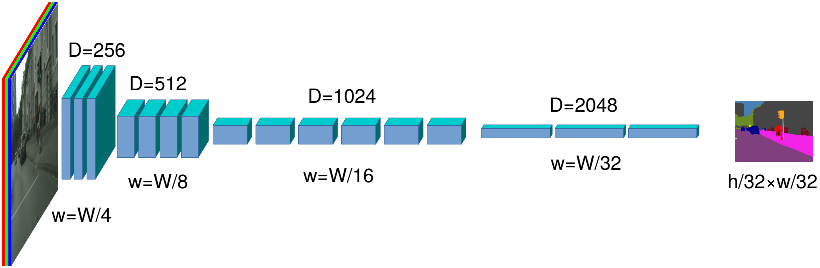
# INTRODUCTION: SEMANTIC SEGMENTATION (5)



smooth surfaces at large objects



small objects




computational requirements: memory and processing time

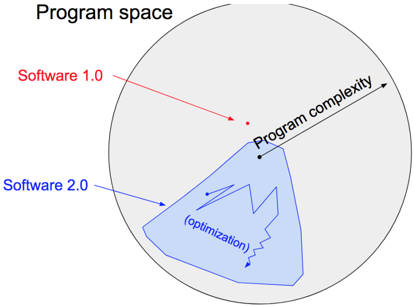
# Convolutional models: two views


[xkcd 1838]

~~Neural networks~~ →

~~Deep learning~~ →

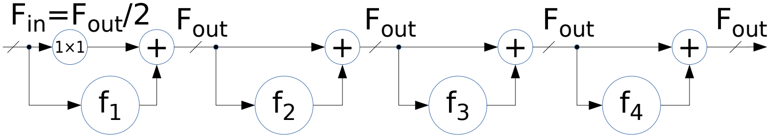
Differential programming (software 2.0)



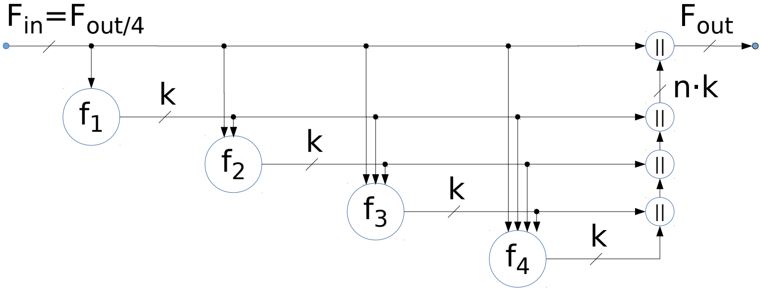

[karpathy17medium]

# Convolutional models: improved design

Advanced connectivity patterns significantly better than simple chaining

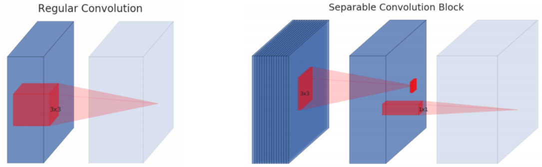



[he16eccv]

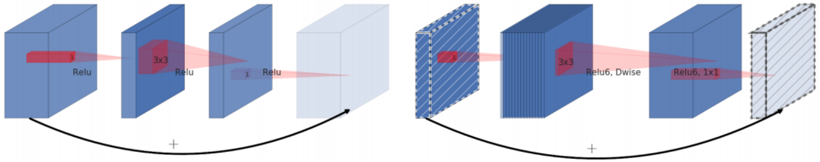



[huang17cvpr]

# Convolutional models: less parameters




idea 1: depthwise separable convolution




regular residual unit

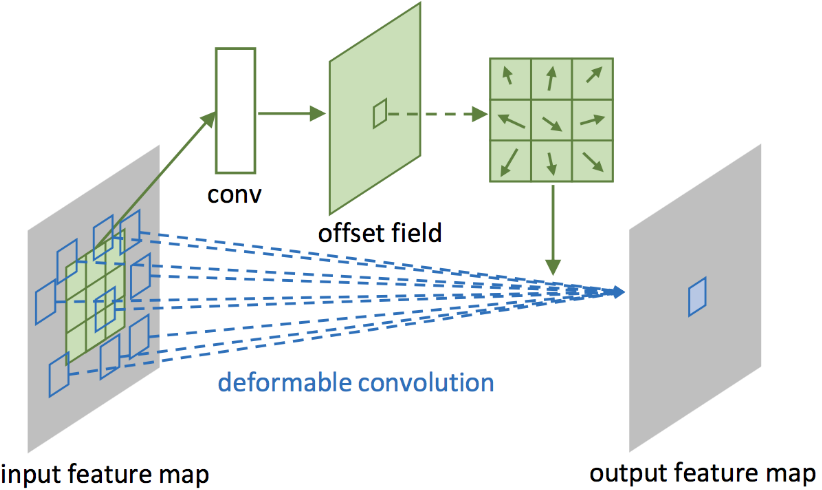
dws separable residual unit

idea 2: depthwise separable convolution on "inflated" representation

[sandler18cvpr]

# Convolutional models: more flexibility

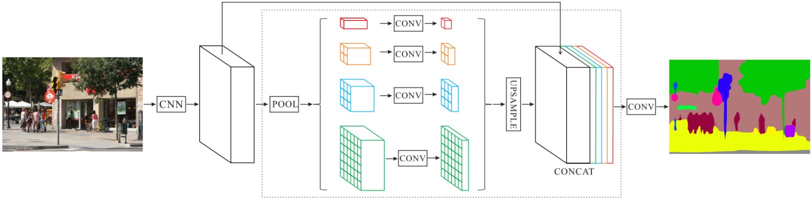
Deformable convolutions:




[dai17iccv]

# Convolutional models: increased receptive field

Spatial pyramid pooling (SPP)




(a) Input Image (b) Feature Map (c) Pyramid Pooling Module (d) Final Prediction

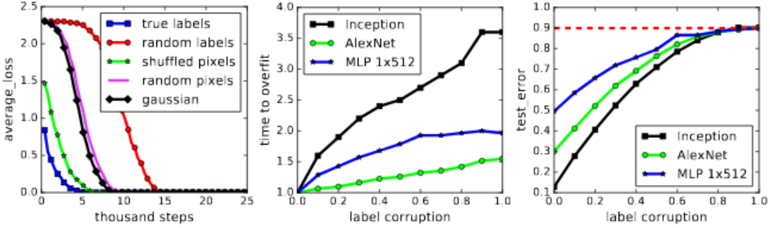
[zhao17cvpr]

Idea: provide wide contextual information to subsequent convolutions

Helps to recognize large objects with a model pre-trained on small images

# Convolutional models: 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, deep models generalize well when trained on correct labels

A theory to explain this behaviour is missing.

# Real-time prediction: approach

Recent work provides solid empirical evidence that convolutional models are extremely resistant to overfitting [zhang17iclr]

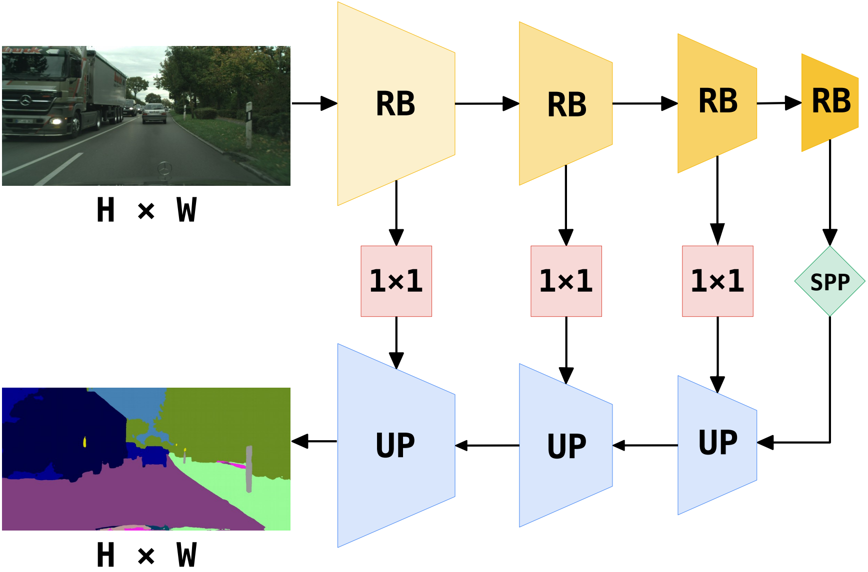
Hence, practitioners tend to overshoot the model capacity

Oversupply of modelling power leads to diminished returns.

On the other hand, it is not sensible to renounce on ImageNet pre-training whenever we deal with natural images

Hence, we base all our real-time models on lightweight ImageNet pre-trained models [orsic19cvpr]

# Real-time prediction: baseline




[orsic19cvpr]

# Real-time prediction: details

Downsampling path (ImageNet backbone): ResNet-18 [he15cvpr] or MobileNetv2 [sandler18cvpr]

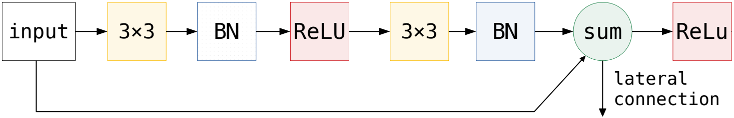
- ensures efficient recognition

Spatial pyramid pooling [zhao17cvpr,kreso19arxiv]

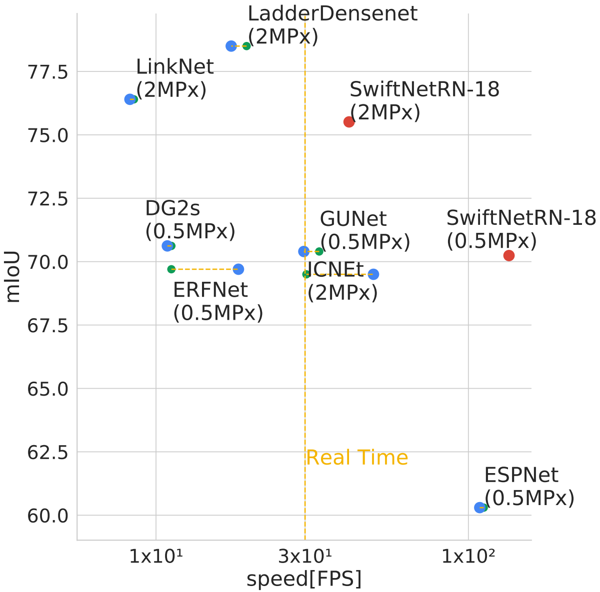
- ensures large receptive field (for large objects)

Ladder-style upsampling [lin17cvpr,kreso17cvrsuad]

- recovers details (for small objects)
- skip-connection taken before ReLU [orsic19cvpr]



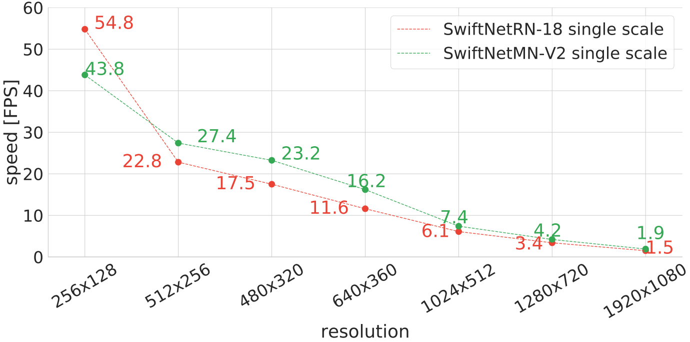
# Real-time prediction: results




[orsic19cvpr]

Best accuracy/latency among all previously published models

# Real-time prediction: embedded

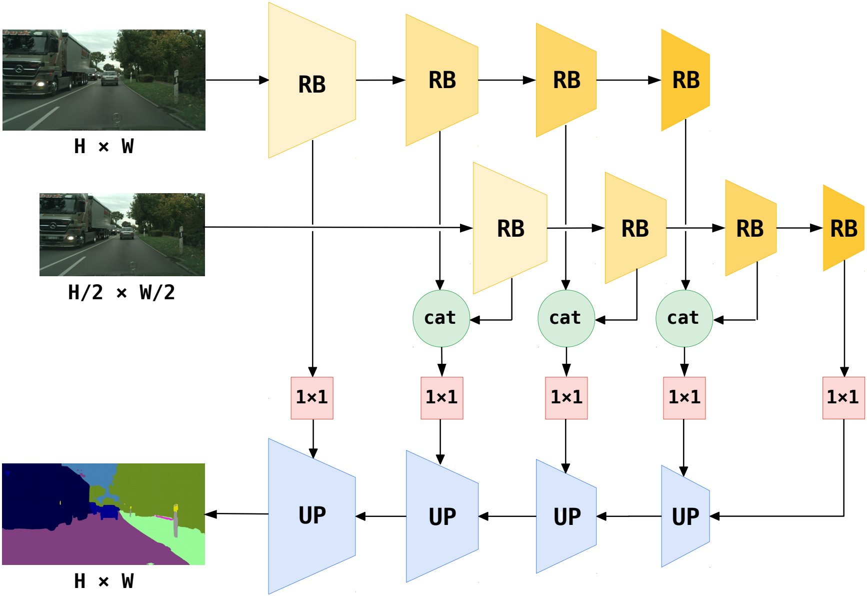



[orsic19cvpr]

Real-time performance on an embeded SoC

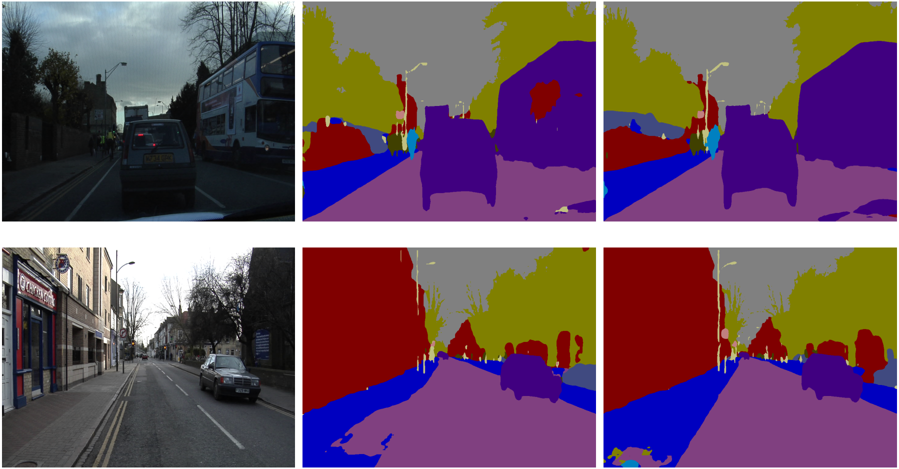
- 256×512 pixels (RGB)
- 27.4 Hz at Jetson TX2 (15W)

# Real-time prediction: pyramid fusion




[orsic19cvpr]

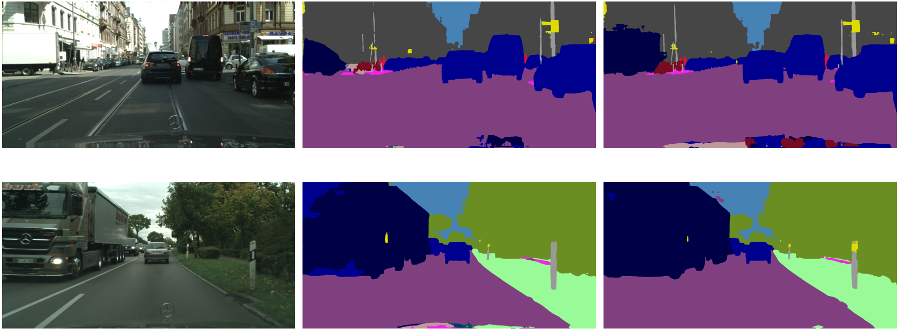
# Real-time prediction: results



[orsic19cvpr]

In both cases, the pyramid fusion improves recognition of surfaces at large structures (bus, sidewalk)

# Real-time prediction: results



[orsic19cvpr]

Again, the pyramid fusion improves recognition of large objects

Other experiments show that pyramid fusion enlarges the effective receptive field of the predictions.

# Real-time prediction: conclusions

ImageNet pre-training leads to 5pp mIoU improvement

Classifier capacity can be compensated by careful design:

- spatial pyramid pooling [zhao17cvpr]
- pyramidal fusion [orsic19cvpr]
- ladder-style upsampling [lin17cvpr,kreso17cvrsuad]

Pyramidal fusion vs spatial pyramid pooling:

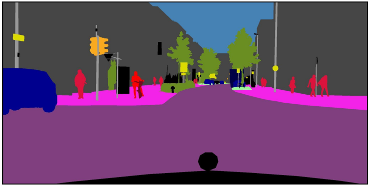
- improved accuracy for 1pp
- improved effective receptive field
- 10% increase in the FLOP count

# Semantic forecasting: the task

Anticipate events by forecasting semantic segmentation of an unobserved future frame ($\Delta t$ = 180 or 540 ms)

current frame (observed)

future frame (unobserved)



groundtruth (used for evaluation)

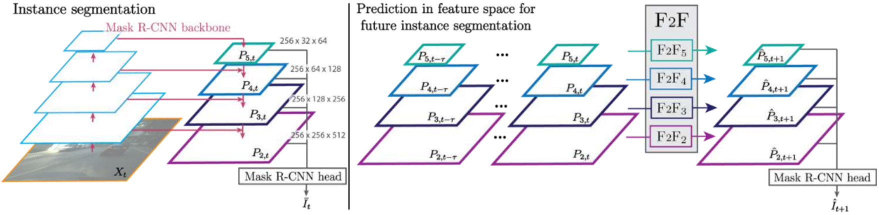
semantic forecast (our result)

# Semantic forecasting: previous work

Recent work suggests that forecasting abstract features is easier than forecasting pixels [luc18eccv]

- abstract features are more informative than pixels
- especially interesting since it can be trained with **no labels**

However, they propose a heavyweight model which requires training a separate mapping at different levels of abstraction [luc18eccv]



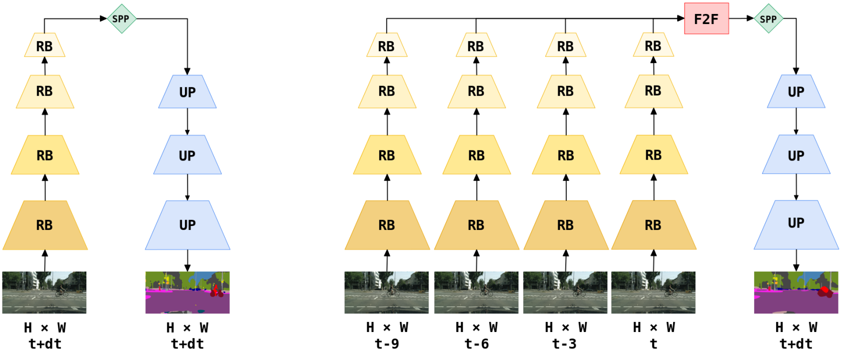
# Semantic forecasting: approach

We again hypothesize that model capacity can be compensated by smart design:

- use a lightweight ImageNet pre-trained recognition backbone (ResNet-18) [orsic19cvpr]
- forecast only the most abstract features
  - use the single-frame model **without** ladder-style upsampling
- use a simple F2F model with **deformable convolutions**
- due to simplicity, we can finetune F2F with supervised loss

A forecasting model with more capacity could not fit into GPU RAM and would require more data to train

# Semantic forecasting: the proposed solution




single-frame model

forecasting model

[saric19gcpr]

# Semantic forecasting: results

| | Short-term | | Mid-term | |
|---|---|---|---|---|
| | mIoU | mIoU-MO | mIoU | mIoU-MO |
| Oracle | 72.5 | 71.5 | 72.5 | 71.5 |
| Copy last segmentation | 52.2 | 48.3 | 38.6 | 29.6 |
| Luc Dil10-S2S [luc17iccv] | 59.4 | 55.3 | 47.8 | 40.8 |
| Luc Mask-S2S [luc18eccv] | / | 55.3 | / | 42.4 |
| Luc Mask-F2F [luc18eccv] | / | 61.2 | / | 41.2 |
| Nabavi [nabavi18bmvc] | 60.0 | / | / | / |
| Bhattacharyya [bhattacharyya19iclr] | 65.1 | / | 51.2 | / |
| Terwilliger [terwilliger19wacv] | **67.1** | **65.1** | 51.5 | 46.3 |
| Luc F2F (our implementation) | 59.8 | 56.7 | 45.6 | 39.0 |
| DeformF2F-8 | 64.4 | 62.2 | 52.0 | 48.0 |
| DeformF2F-8-FT | 64.8 | 62.5 | **52.4** | **48.3** |
| DeformF2F-8-FT (2 samples per seq.) | 65.5 | 63.8 | **53.6** | **49.9** |

# Semantic forecasting: mid-term example


most recent input

future frame (unobserved)

ground truth

mid-term forecast

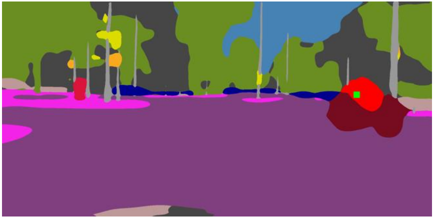
# Semantic forecasting: explaining results

We show pixels with the strongest log-max-softmax gradient (red) in a hand-picked pixel (green)

t-3

t

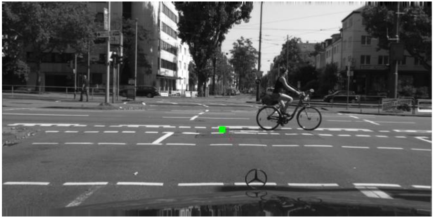
t + 9



forecast

# Semantic forecasting: explaining results (2)

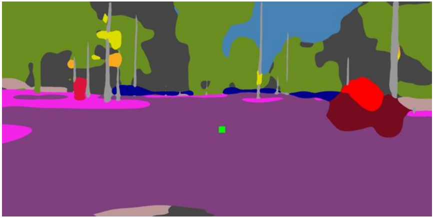
We show pixels with the strongest log-max-softmax gradient (red) in a hand-picked pixel (green)



t-3

t

t + 9



forecast

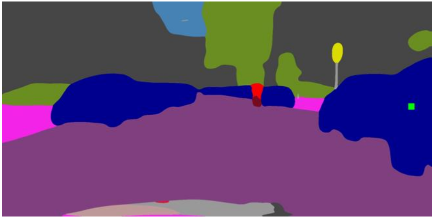
# Semantic forecasting: explaining results (3)

We show pixels with the strongest log-max-softmax gradient (red) in a hand-picked pixel (green)

t-3

t

t + 9



forecast

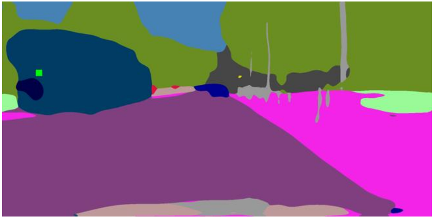
# Semantic forecasting: explaining results (4)

We show pixels with the strongest log-max-softmax gradient (red) in a hand-picked pixel (green)

t-3

t

t + 9

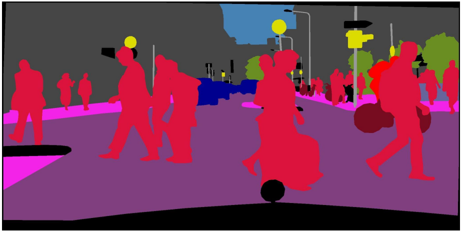


forecast

# Semantic forecasting: pedestrians (short-term)


most recent input


future frame (unobserved)



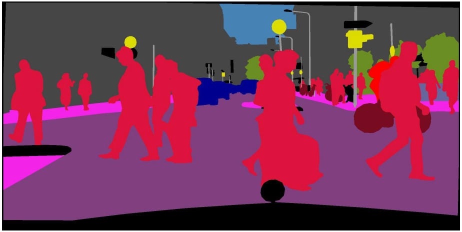
ground truth

short-term forecast

# Semantic forecasting: pedestrians (mid-term)


most recent input


future frame (unobserved)



ground truth

mid-term forecast

# Semantic forecasting: conclusion

- novel method for anticipating semantic segmentation in driving scenarios based on feature-to-feature forecasting
- we forecast only the most abstract features because of coarse resolution and high semantic content
- we favor deformable convolutions in order to account for geometric nature of F2F forecasting
- due to simplicity our F2F module allows joint fine-tuning with the upsampling path and achieves real-time performance
- state-of-the-art results on Cityscapes mid-term forecast

# Conclusion: lightweight models rule

We improve upon the state-of-the-art real-time semantic prediction and forecasting by trading in model capacity

Model capacity can be compensated by careful design:

- residual connections [he15cvpr]
- dws and deformable convolutions [sandler18cvpr,dai17iccv]
- spatial pyramid pooling [zhao17cvpr]]
- pyramidal fusion [orsic19cvpr]
- ladder-style upsampling [kreso17cvrsuad,lin17cvpr]

Future work:

- custom lightweight architectures for efficient recognition
- efficient architectures for video analysis
- include the remaining ingredients (uncertainty, robustness, ...)

# Conclusion: discussion

Thank you for your attention!

Questions?

Presented results have been developed on projects SafeTram (Končar, ERDF) [1], Semantic Forecasting (Rimac Automobili), Datacross (ERDF) [2], and MultiClod (HRZZ) [3].

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

[2] `https://across-datascience.zci.hr/datacross`

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