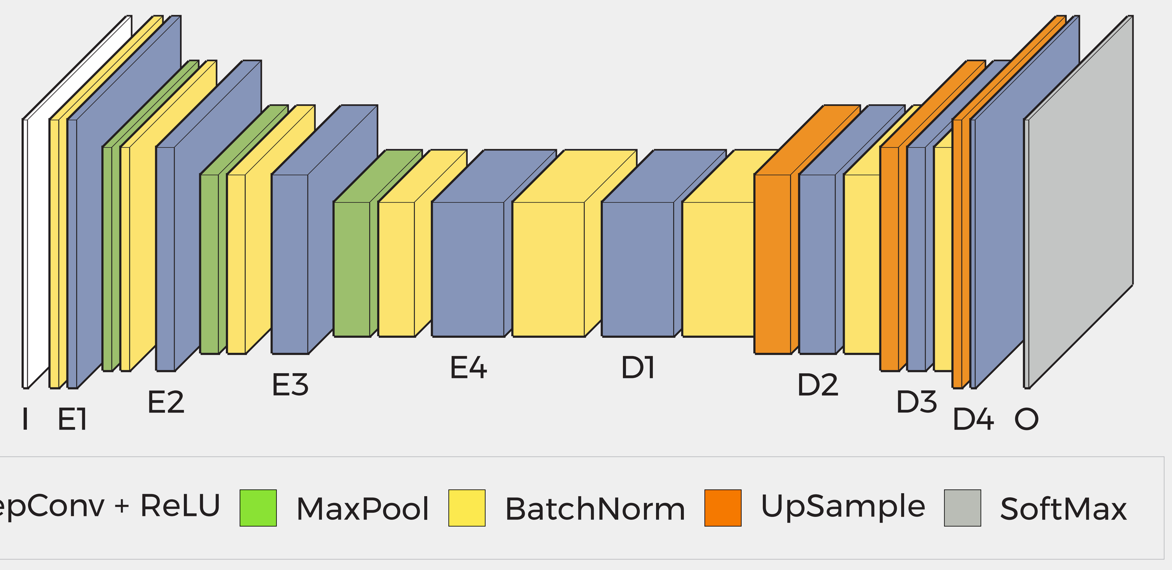
Deep Learning for Semantic Segmentation on Minimal Hardware

Sander G. van Dijk and Marcus M. Scheunemann

University of Hertfordshire, AL10 9AB, UK

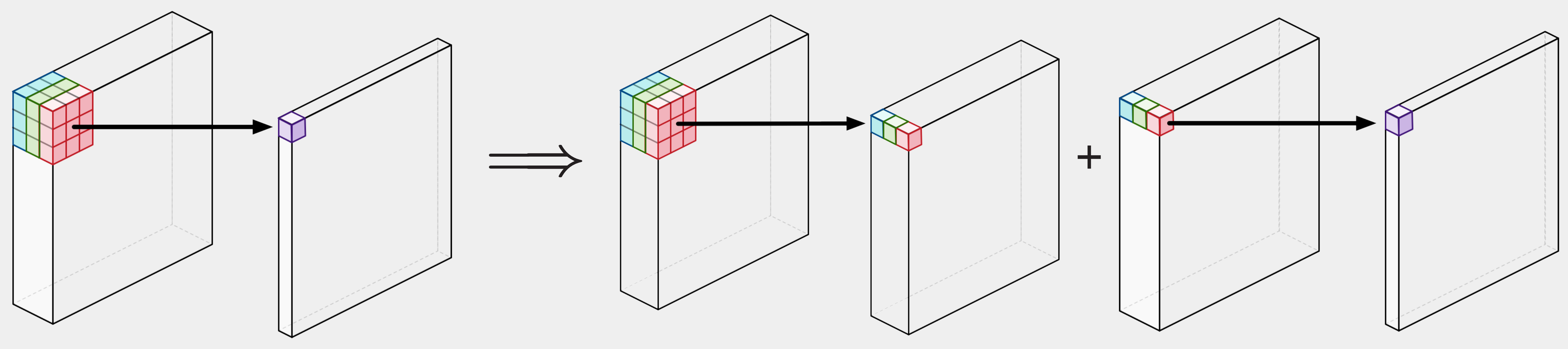
Deep learning has revolutionised many fields, but it is still challenging to transfer its success to small mobile robots with minimal hardware. Specifically, some work has been done to this effect in the RoboCup humanoid football domain, but results that are performant and efficient and still generally applicable outside of this domain are lacking. We propose an approach conceptually different from those taken previously. It is based on semantic segmentation and it is being able to process full VGA images in real-time on a low-power mobile processor, e.g., an Odroid-XU4. It can further handle multiple image dimensions without retraining and it does not require specific domain knowledge to achieve a high frame rate.

Approach - Fully Convolutional Semantic Segmentation



- Small: no fully connected layers
- Resolution independent: VGA/QVGA/4K without retraining
- No domain knowledge: no problem specific preprocessing
- RoboCup friendly:
  - drop-in replacement of LUT labelling
  - applicable on minimal mobile hardware, e.g., Odroid-XU4

Depthwise Separable Convolution



source: http://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/

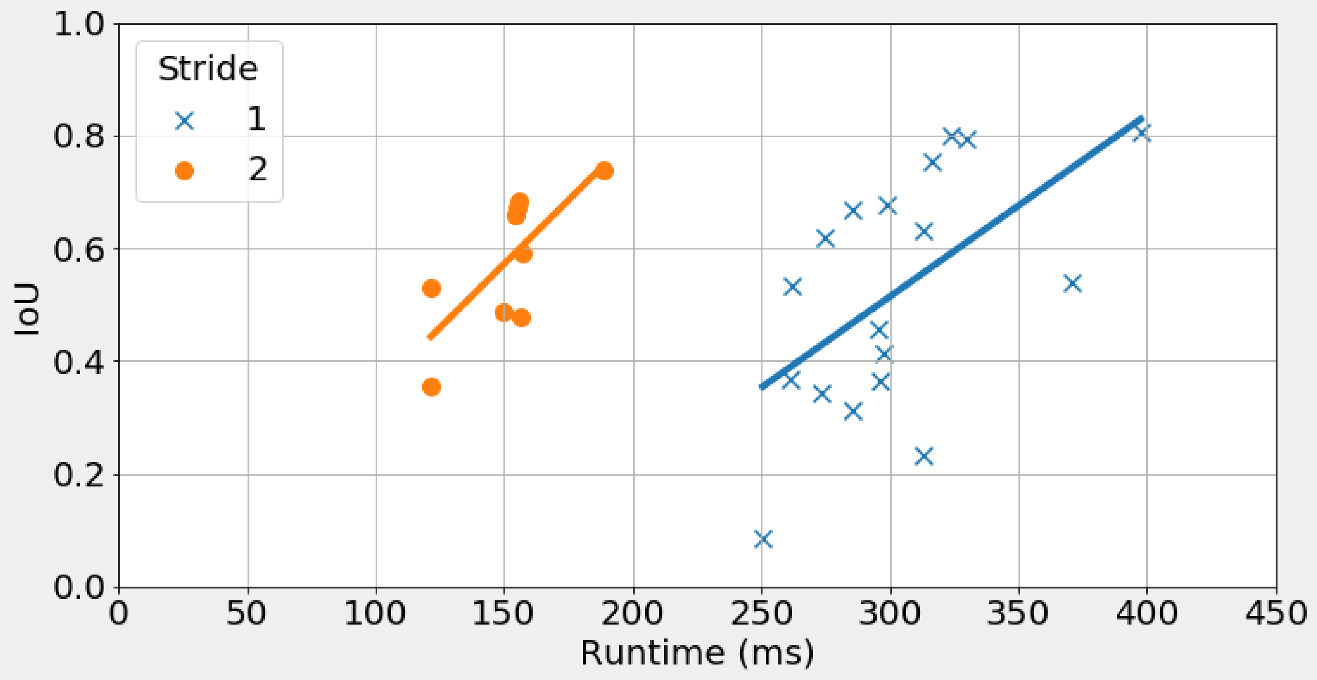
Split regular convolution into filter and combination step:

- Separate 2D kernels applied to each input channel
- 1×1 convolution combines results of filters

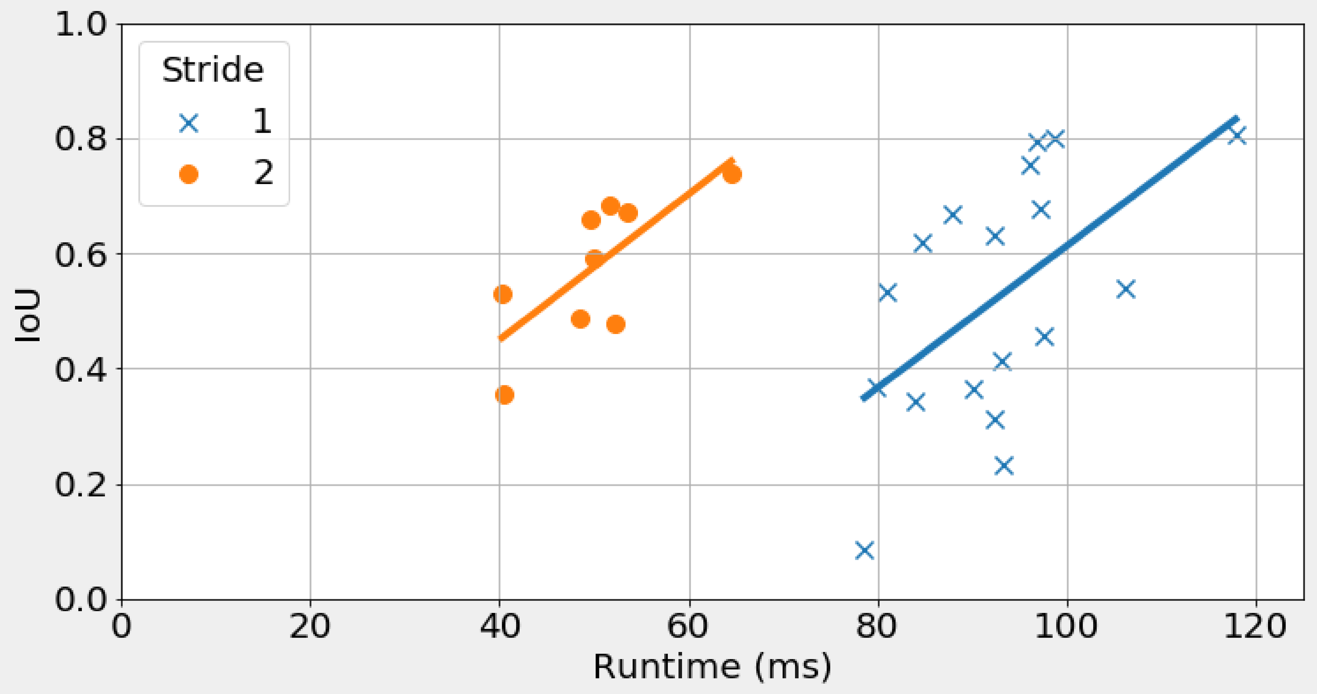
Reduction computational cost:

$$\frac{1}{\text{nr output features}} + \frac{1}{\text{kernelsize}^2}$$

Performance vs Runtime



Full VGA: 2.5 to 8 fps



QVGA: 8 to 25 fps

Hardware: Odroid-XU4, Samsung Exynos 5422 Cortex-A15 2 GHz and Cortex-A7 Octa core

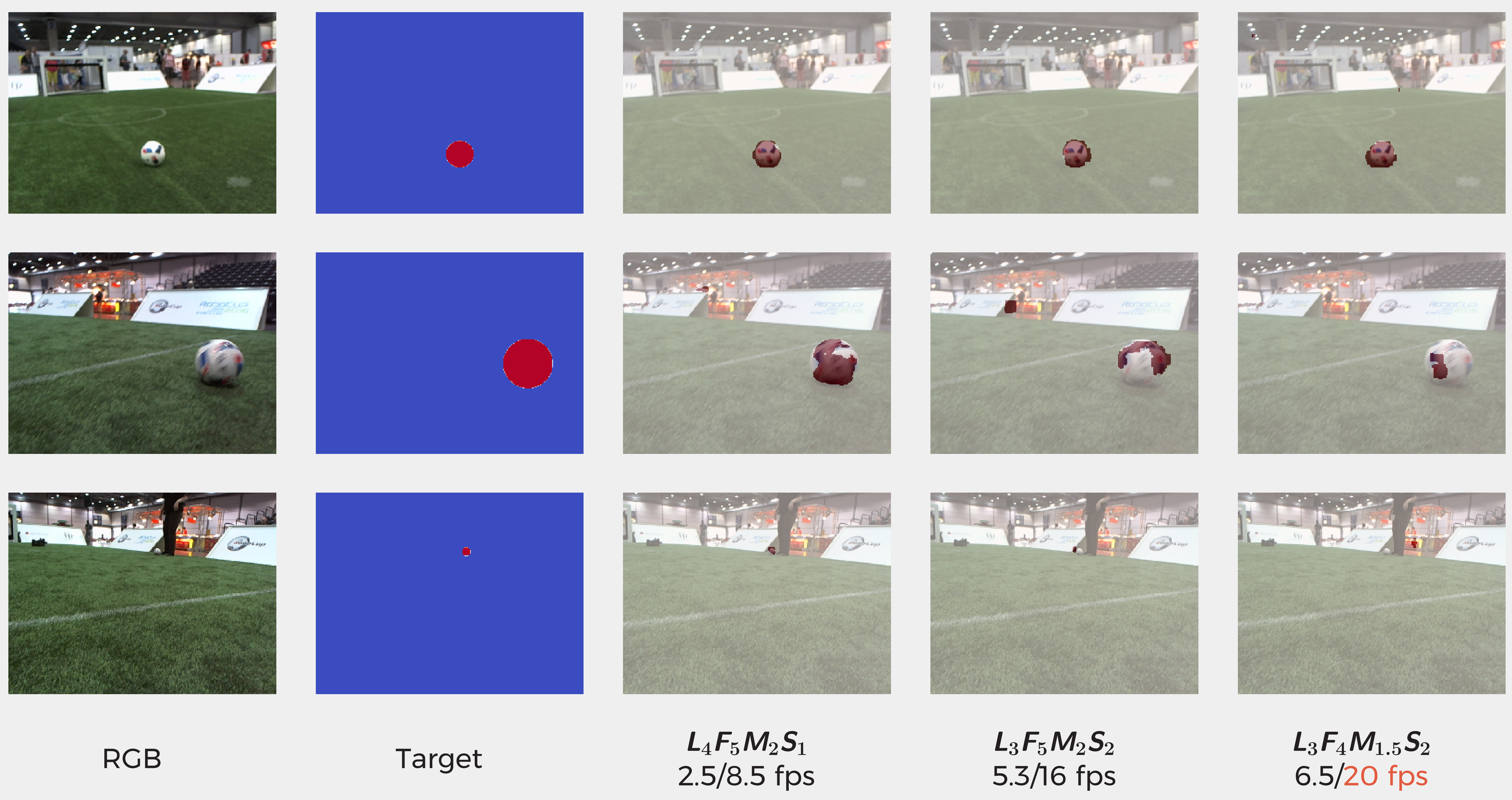
Network Architecture

- Number of Layers (L) - $L \in \{3, 4\}$, number of encoding and decoding layers
- Number of Filters (F) - $F \in \{3, 4, 5\}$, number of filters in first encoding layer.
- Filter Multiplier (M) - $M \in \{1.25, 1.5, 2\}$, increase factor of filters for each subsequent encoding layer
- Convolution Stride (S) - $S \in \{1, 2\}$, stride used in each convolution layer

Networks denoted as $L_xF_yM_zS_w$, e.g.:

- $L_3F_3M_{1.25}S_2$ - smallest network; 352 weights
- $L_4F_5M_2S_1$ - largest network; 5,307 weights

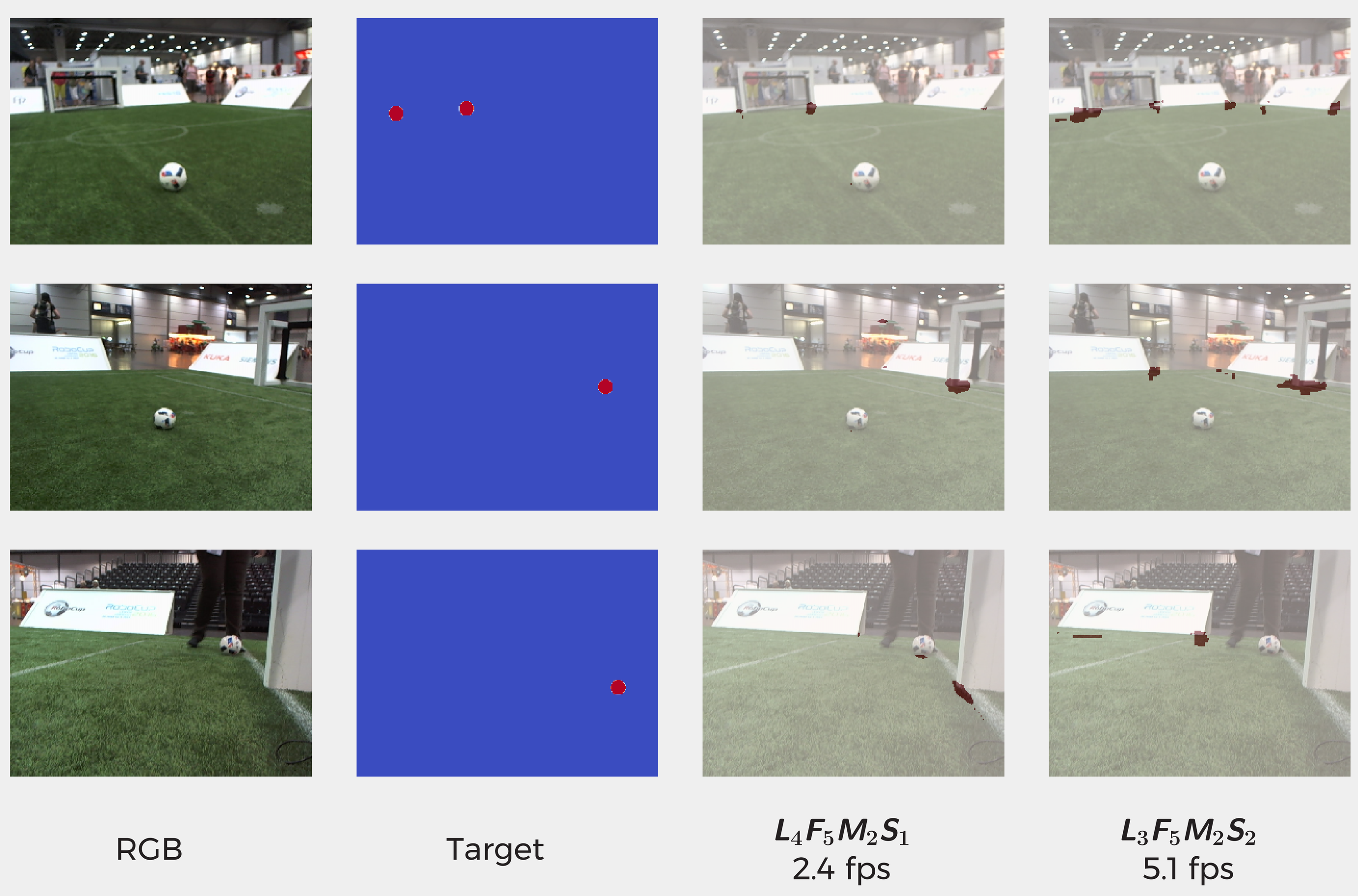
Binary Segmentation



dataset: https://imagetagger.bit-bots.de/images/imageset/12/

- 2 classes: 'ball', 'not ball'
- FPS for full VGA/QVGA resolutions

Multiclass Segmentation



dataset: https://imagetagger.bit-bots.de/images/imageset/233/

- 3 classes: 'ball', 'goal post', 'other'
- FPS for full VGA resolution
- IoU drop ball: 4 to 7%
- IoU goalposts: 0.273 ($L_4F_5M_2S_1$), 0.102 ($L_3F_5M_2S_2$)

sgvandijk@gmail.com