Image Segmentation using Neural Networks

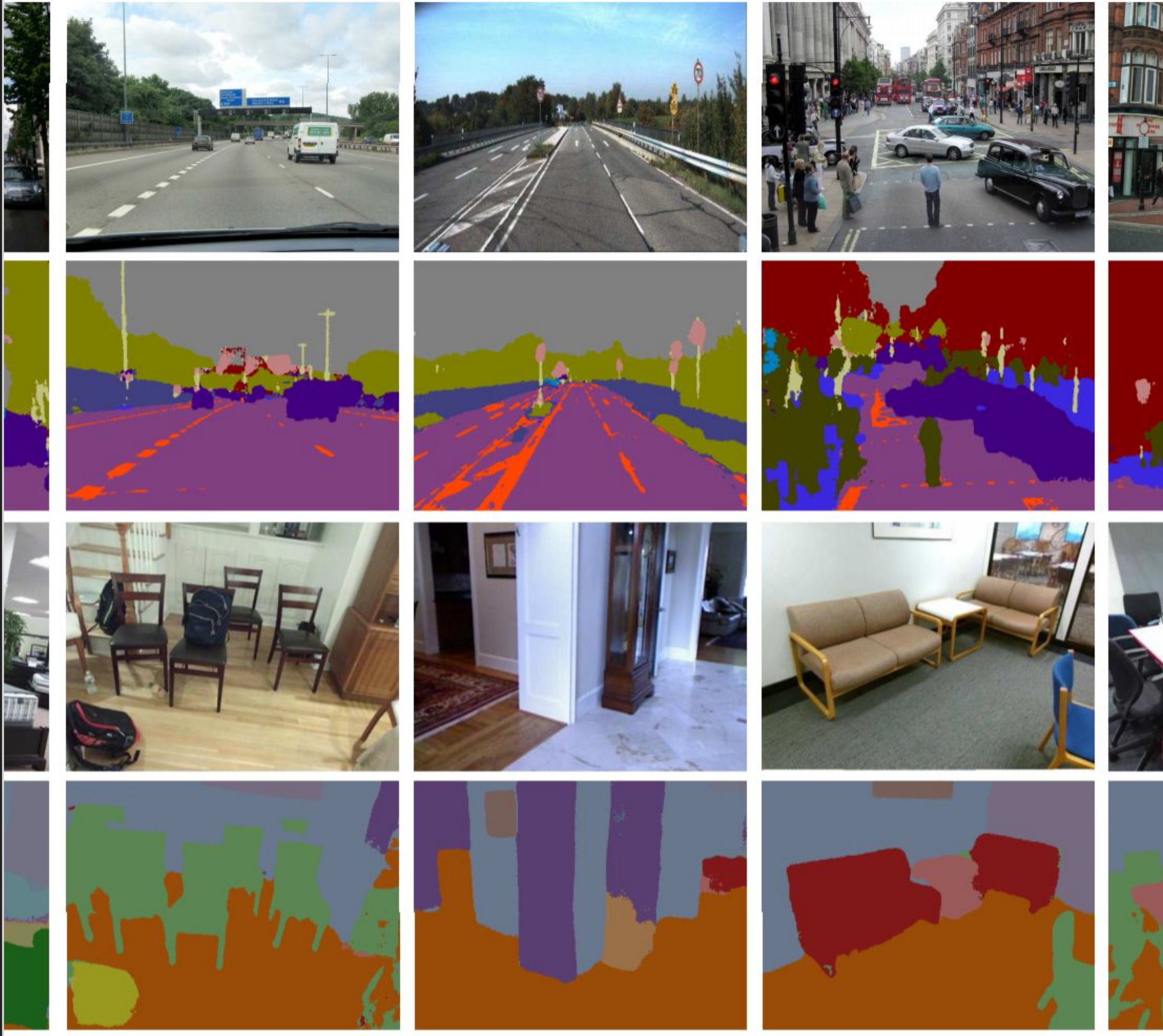
Jakub Náplava, Jan Klůj, Ondřej Švec

What is Segmentation?

- semantic segmentation
- instance segmentation

http://stackoverflow.com/questions/33947823/what-is-semantic-segmentation-compared-to-segmentation-and-scene-labeling

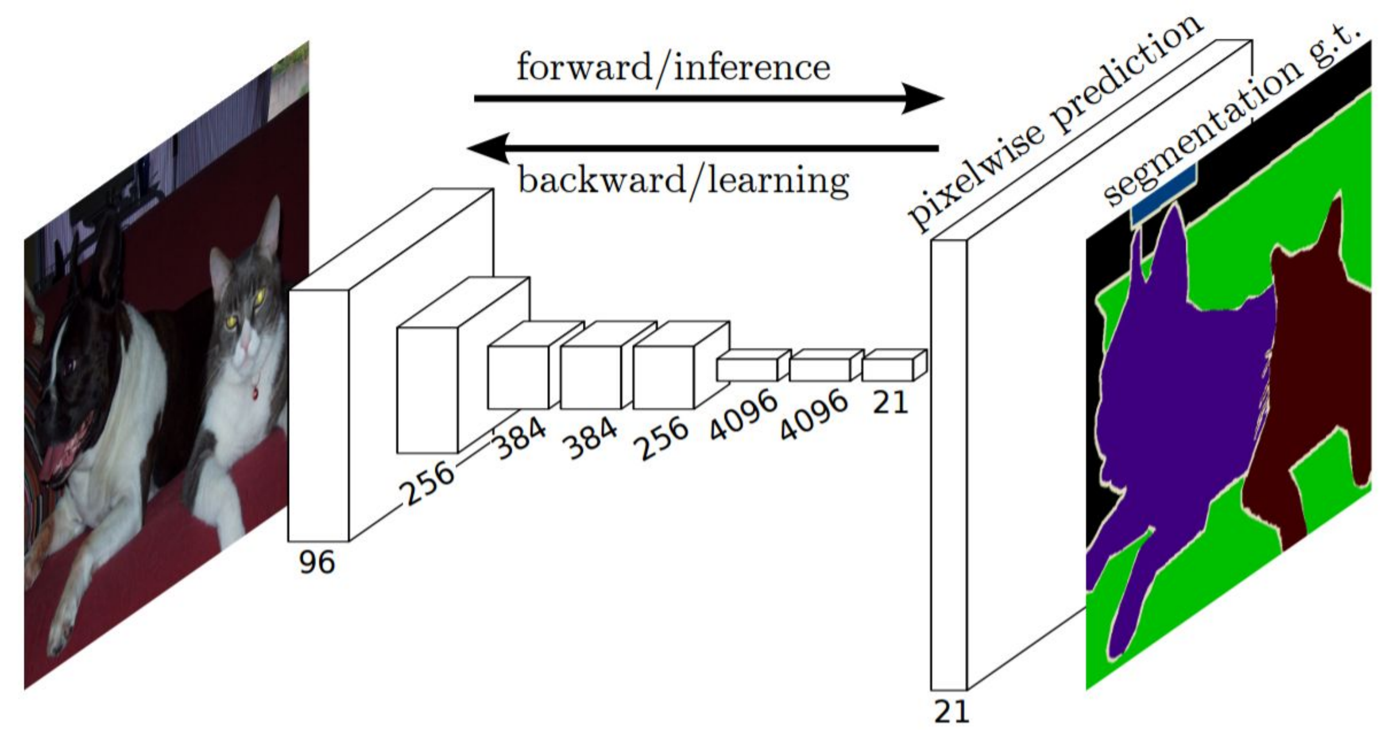
http://www.cs.toronto.edu/~urtasun/courses/CSC2541/08_instance.pdf



Architectures

- FCN (Nov 14)
  https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf
- SegNet (Nov 15)
  https://arxiv.org/pdf/1511.00561.pdf
- DeconvNet (May 15)
  https://arxiv.org/pdf/1505.04366v1.pdf
- DeepLab-LargeFOV (Dec 14)
  https://arxiv.org/pdf/1412.7062v4.pdf

FCN Fully Convolutional Networks for Semantic Segmentation



https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

FCN Fully Convolutional Networks for Semantic Segmentation

- VGG16/GoogLeNet architectures
- conv layer 1x1, #channels = #classes
- one upsampling layer (transposed deconv)
- skip connections

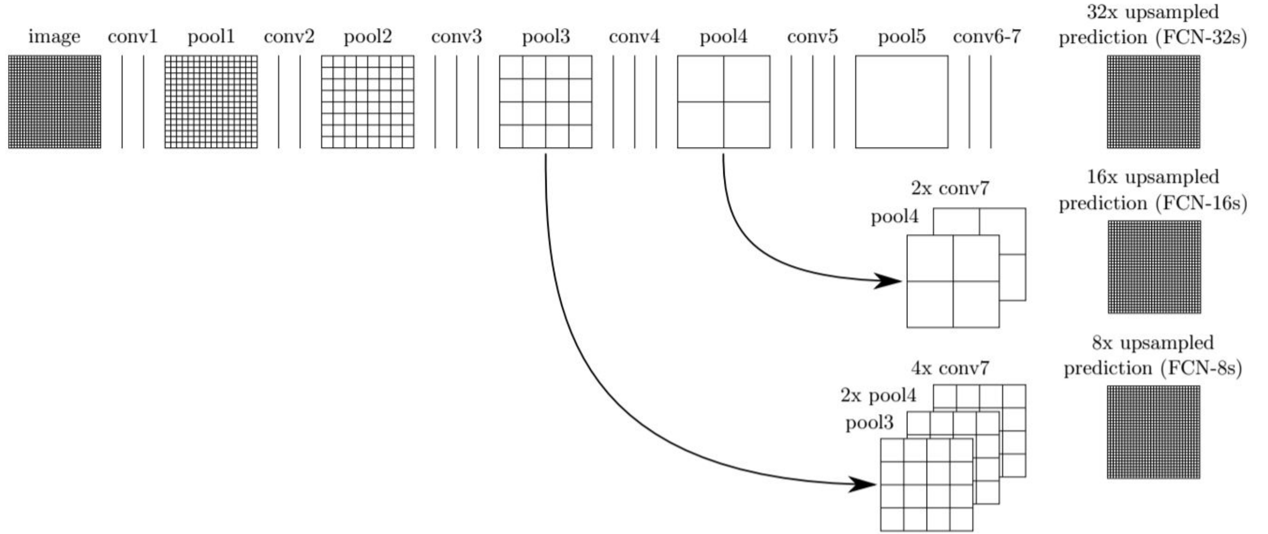
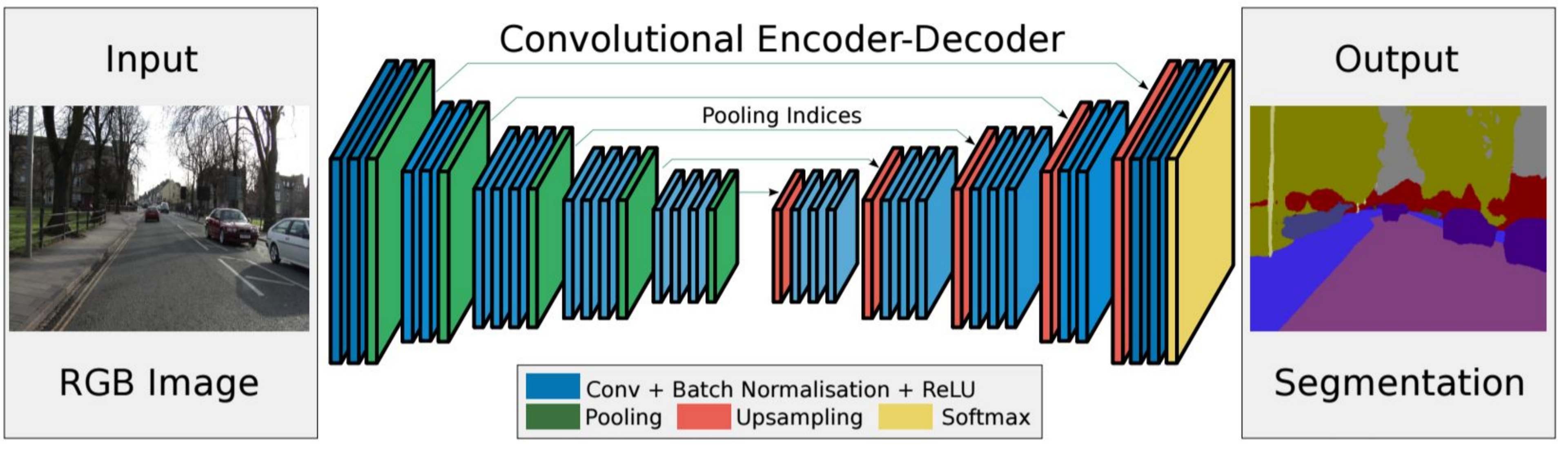


Figure 3. Our DAG nets learn to combine coarse, high layer information with fine, low layer information. Pooling and prediction layers are shown as grids that reveal relative spatial coarseness, while intermediate layers are shown as vertical lines. First row (FCN-32s): Our single-stream net, described in Section 4.1, upsamples stride 32 predictions back to pixels in a single step. Second row (FCN-16s): Combining predictions from both the final layer and the pool4 layer, at stride 16, lets our net predict finer details, while retaining high-level semantic information. Third row (FCN-8s): Additional predictions from pool3, at stride 8, provide further precision.

https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

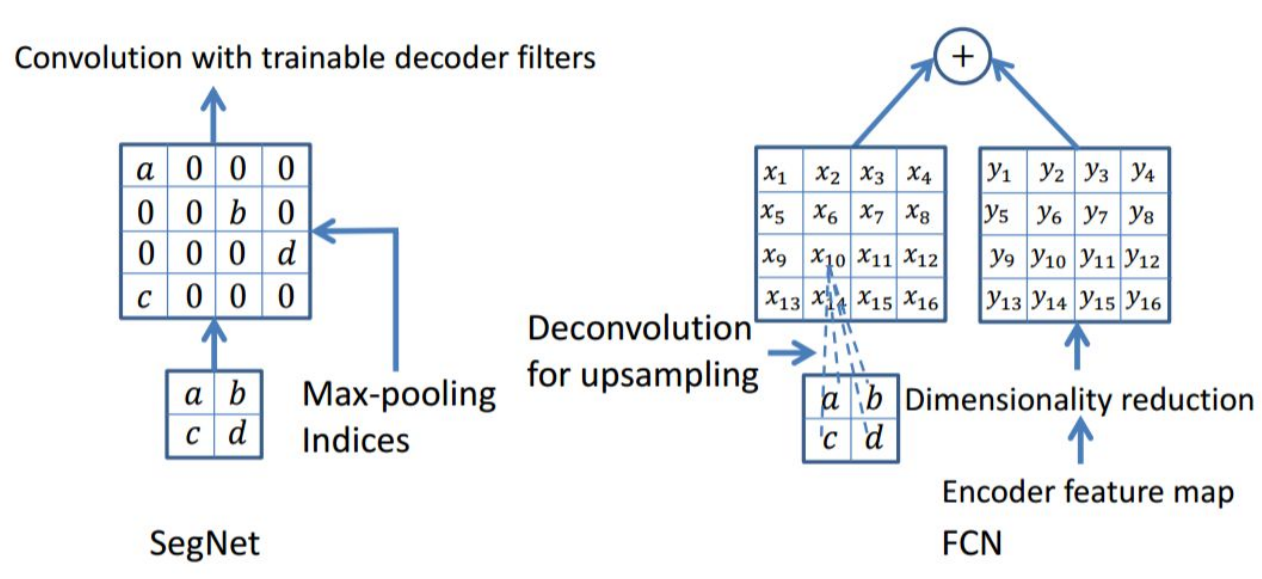
SegNet A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation



https://arxiv.org/pdf/1511.00561.pdf

SegNet A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

- VGG16 architecture
- special demaxpool
- conv layers after demaxpool



https://arxiv.org/pdf/1511.00561.pdf

DeconvNet Learning Deconvolution Network for Semantic Segmentation

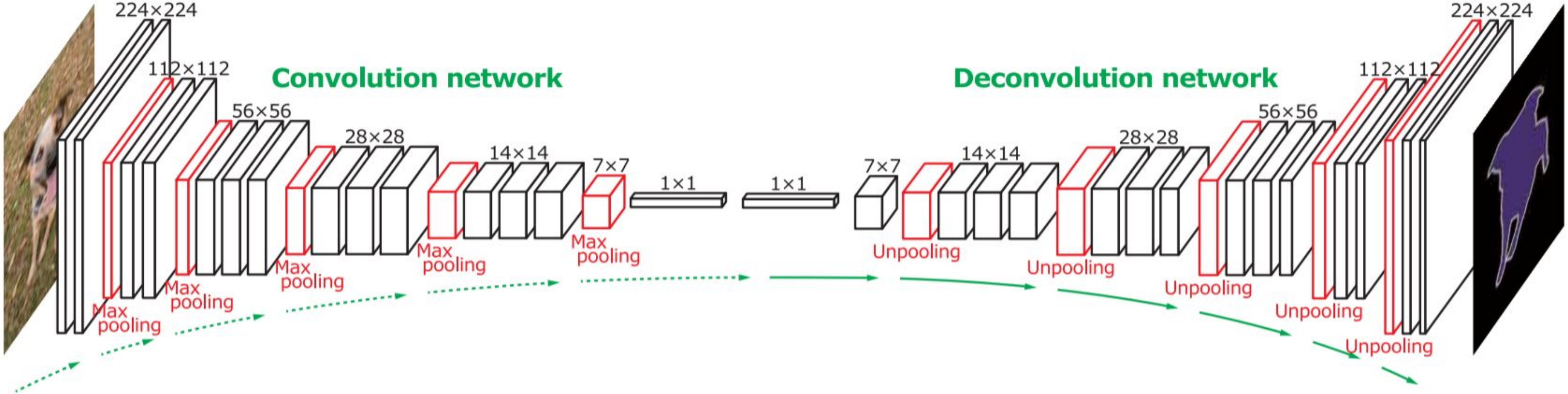


Figure 2. Overall architecture of the proposed network. On top of the convolution network based on VGG 16-layer net, we put a multi-layer deconvolution network to generate the accurate segmentation map of an input proposal. Given a feature representation obtained from the convolution network, dense pixel-wise class prediction map is constructed through multiple series of unpooling, deconvolution and rectification operations.

https://arxiv.org/pdf/1505.04366v1.pdf

Upsampling methods

- bilinear interpolation
- transposed convolution
- naive demaxpool
- learnable demaxpool
- fixed demaxpool

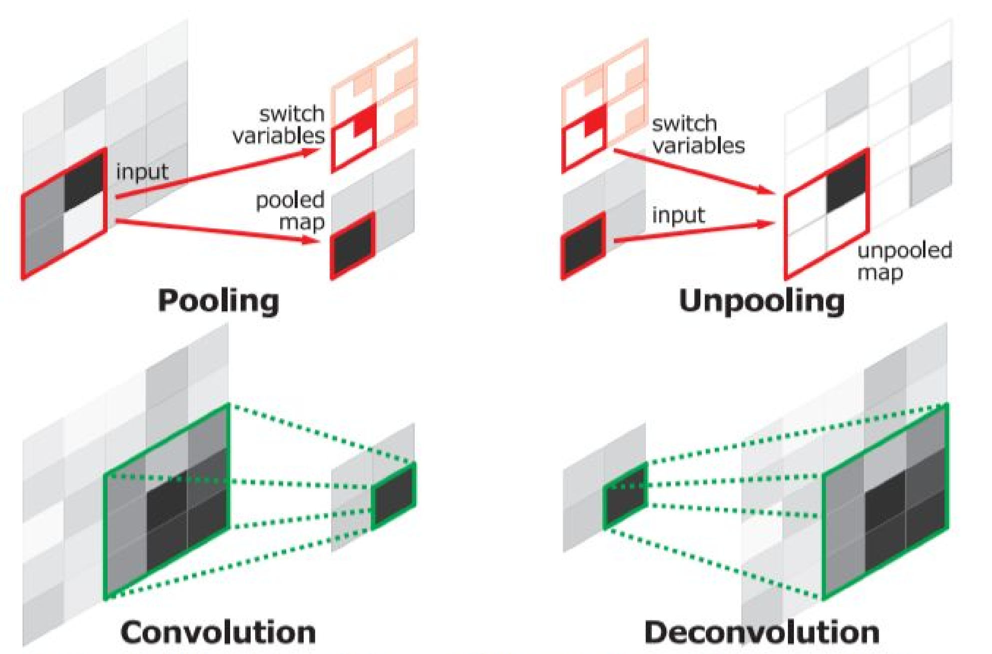
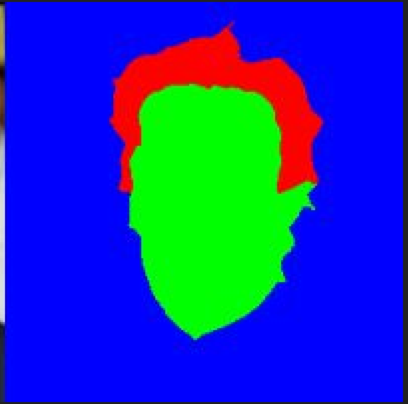
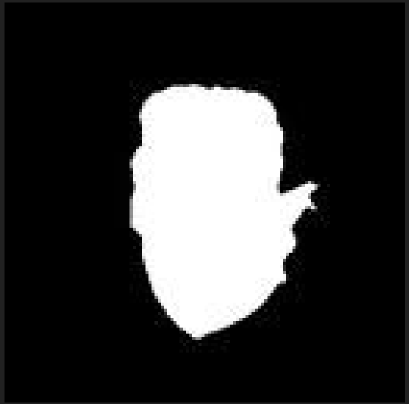


Figure 3. Illustration of deconvolution and unpooling operations.

Dataset

- http://vis-www.cs.umass.edu/lfw/part_labels/
- part of ‘Labeled Faces in the Wild’ dataset
- 2927 samples

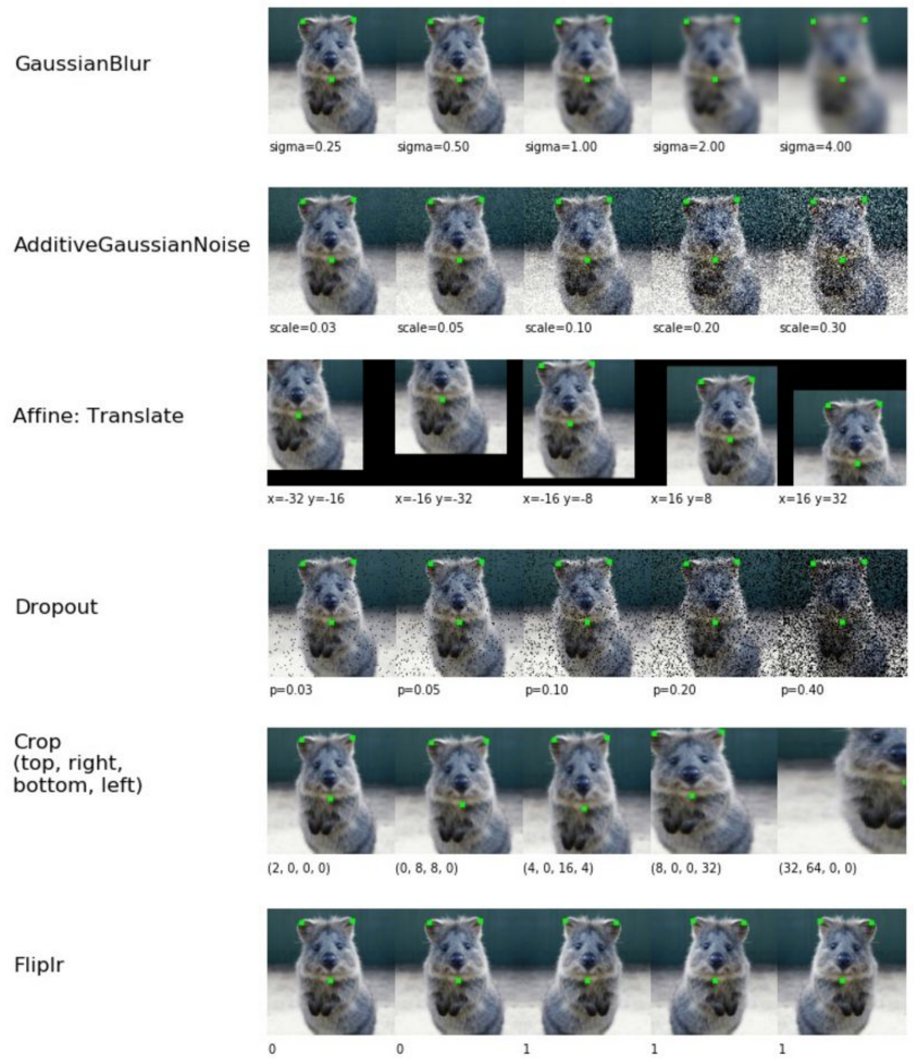




Data augmentation

- crop
- flip
- gaussian blur
- dropout
- additive gaussian noise
- affine

https://github.com/aleju/imgaug



Common accuracy metrics

- per pixel accuracy
- mean accuracy
- mean intersection over union (IU)
- frequency weighted IU

$$\sum_i n_{ii} / \sum_i t_i$$

$$(1/n_{\mathrm{cl}}) \sum_i n_{ii} / t_i$$

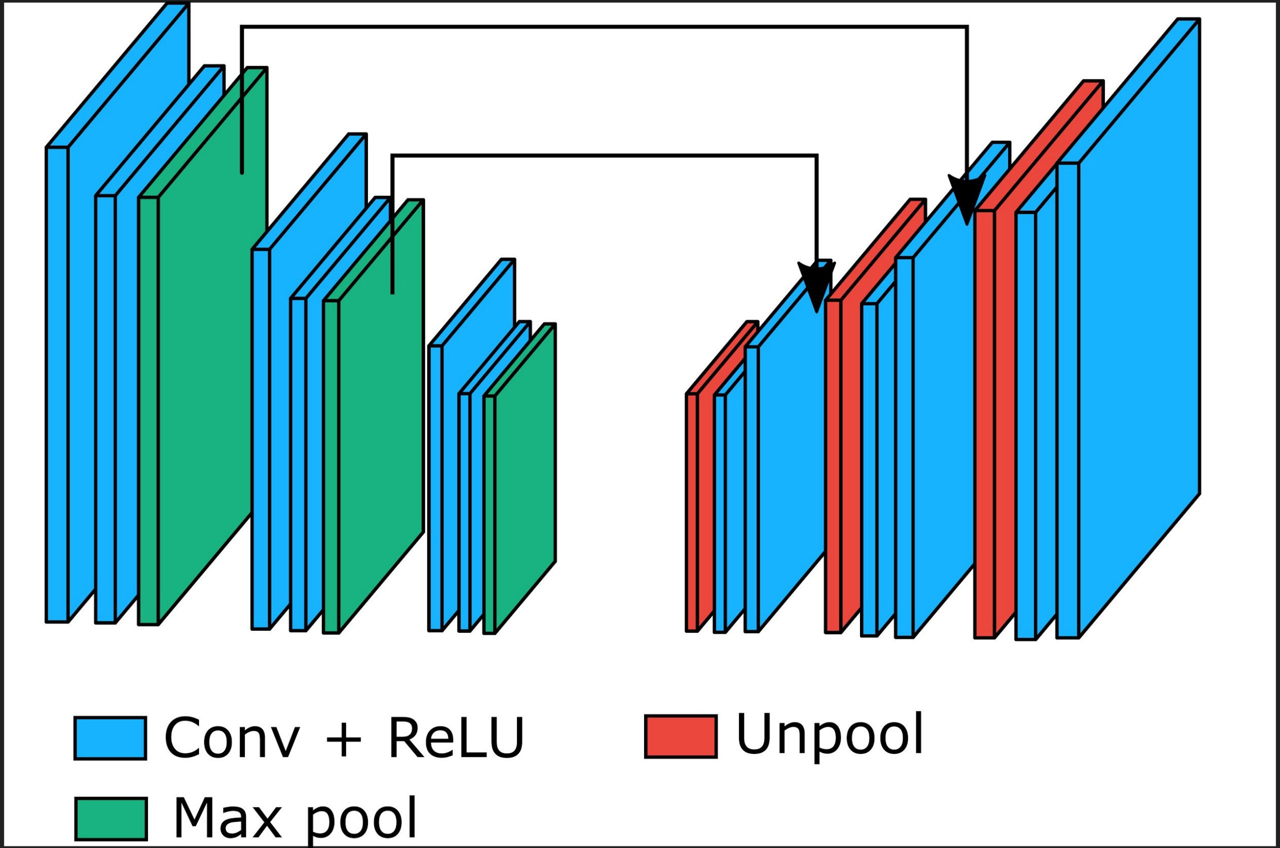
$$(1/n_{\mathrm{cl}}) \sum_i n_{ii} / \left(t_i + \sum_j n_{ji} - n_{ii} \right)$$

$$\left(\sum_k t_k\right)^{-1} \sum_i t_i n_{ii} / \left(t_i + \sum_j n_{ji} - n_{ii} \right)$$

nij be the number of pixels of class i predicted to belong to class j, where there are ncl different classes, and let ti = P j nij be the total number of pixels of class i.

$$t_i = \sum_j n_{ij}$$

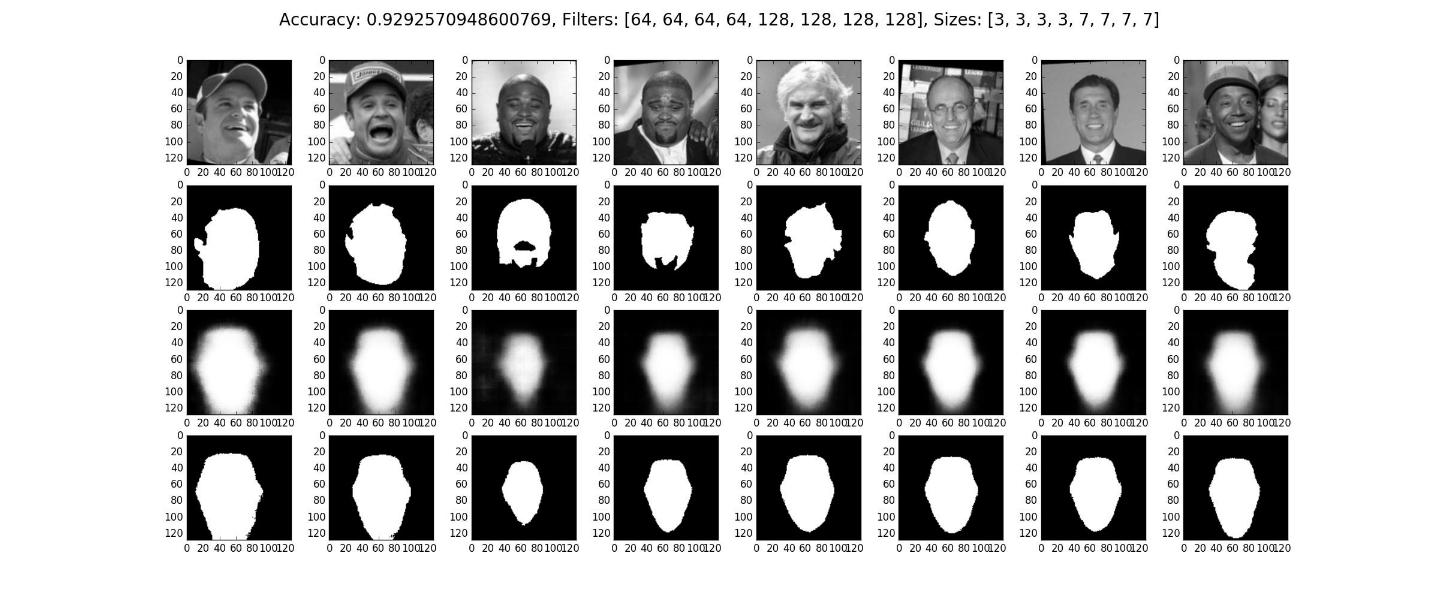
Our architecture



Experiments and results

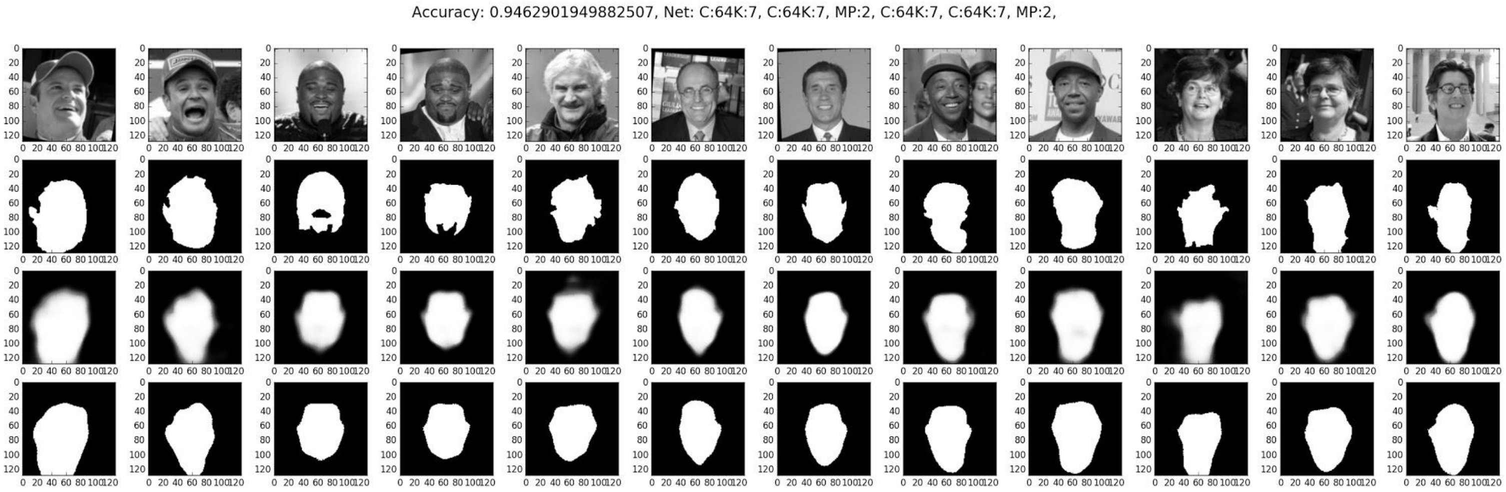
- Architecture: no max pools, no shared weights, no skip connections

  ⇒ 92.92% (per pixel accuracy)



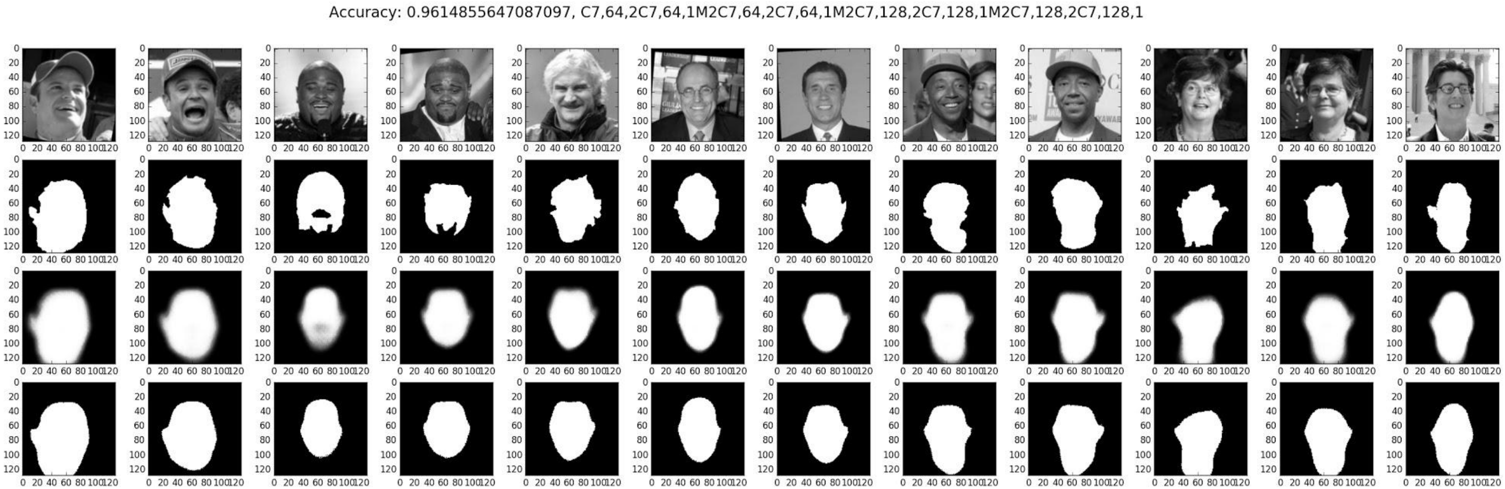
Experiments and results

- Architecture: No shared weights, no skip connections ⇒ 94.62%
    - 4 conv + 4 deconv



Experiments and results

- Architecture: more layers… 8 conv + 8 deconv ⇒ 96.14%

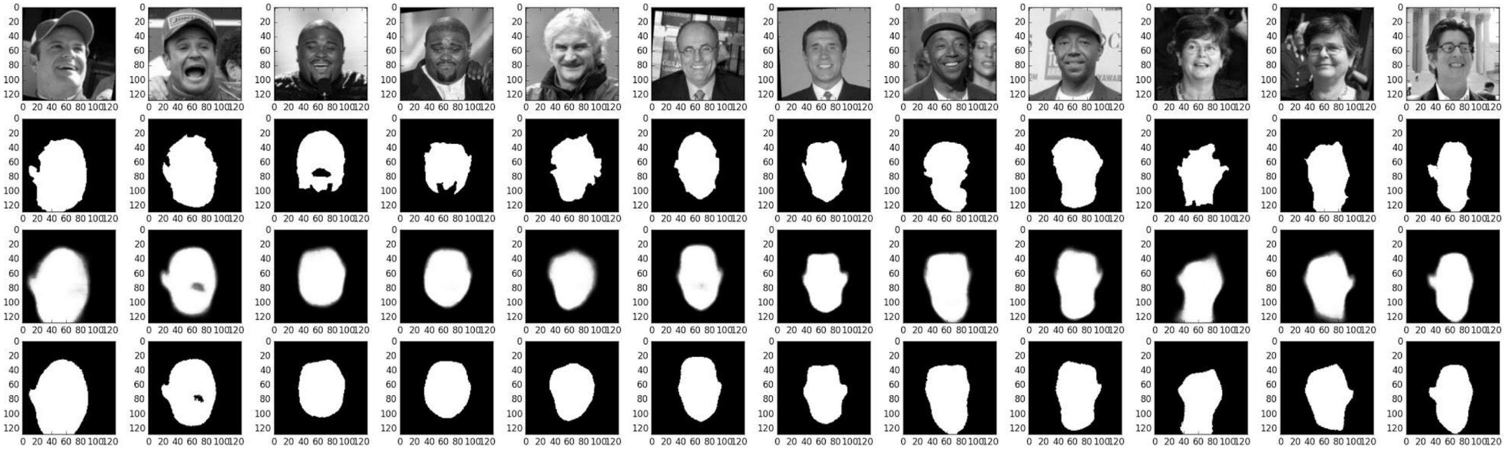


Experiments and results

- Architecture:
  - kernels - 7x7
  - channels - 64
  - shared weights between conv & deconv

96.64 %

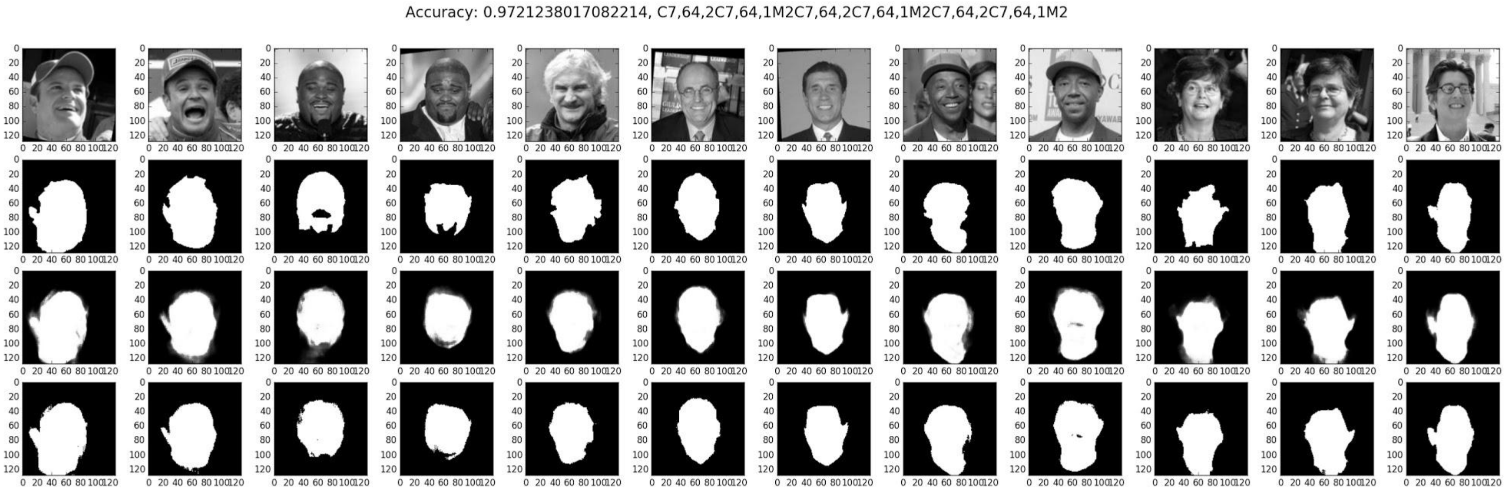
Accuracy: 0.9664298295974731, C7,64,2C7,64,1M2C7,64,2C7,64,1M2C7,64,2C7,64,1M2



Experiments and results

- Architecture:
    - kernels - 7x7
    - channels - 64
    - shared weights between conv & deconv
    - added skip connections

97.21 %

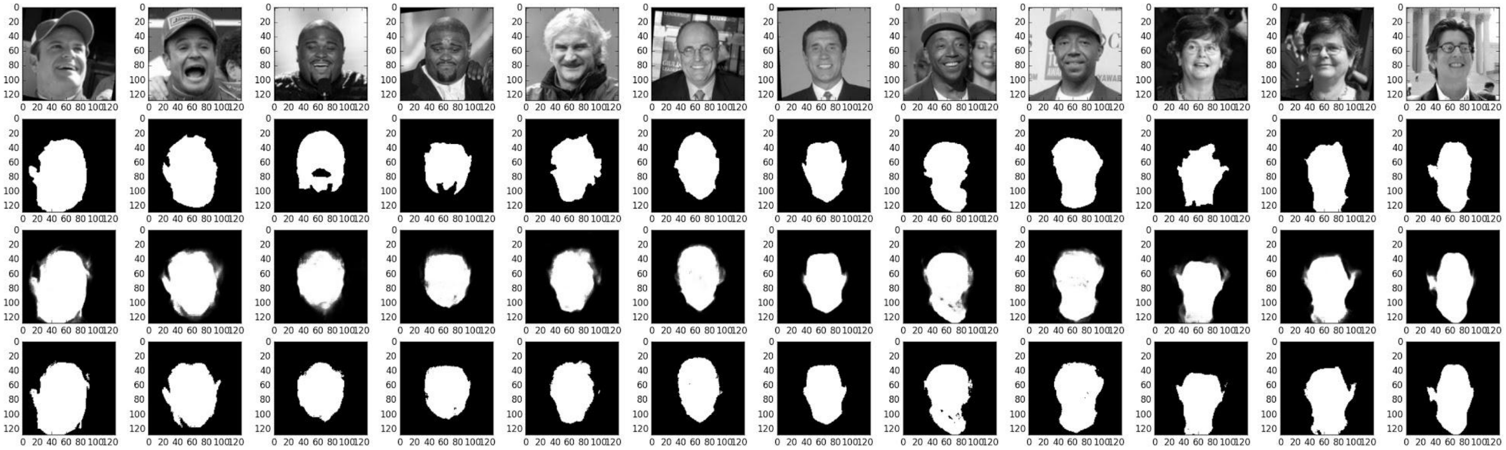


Experiments and results

- Architecture:
  - kernels - 7x7
  - channels - 64
  - shared weights between conv & deconv
  - added skip connections + image standardization

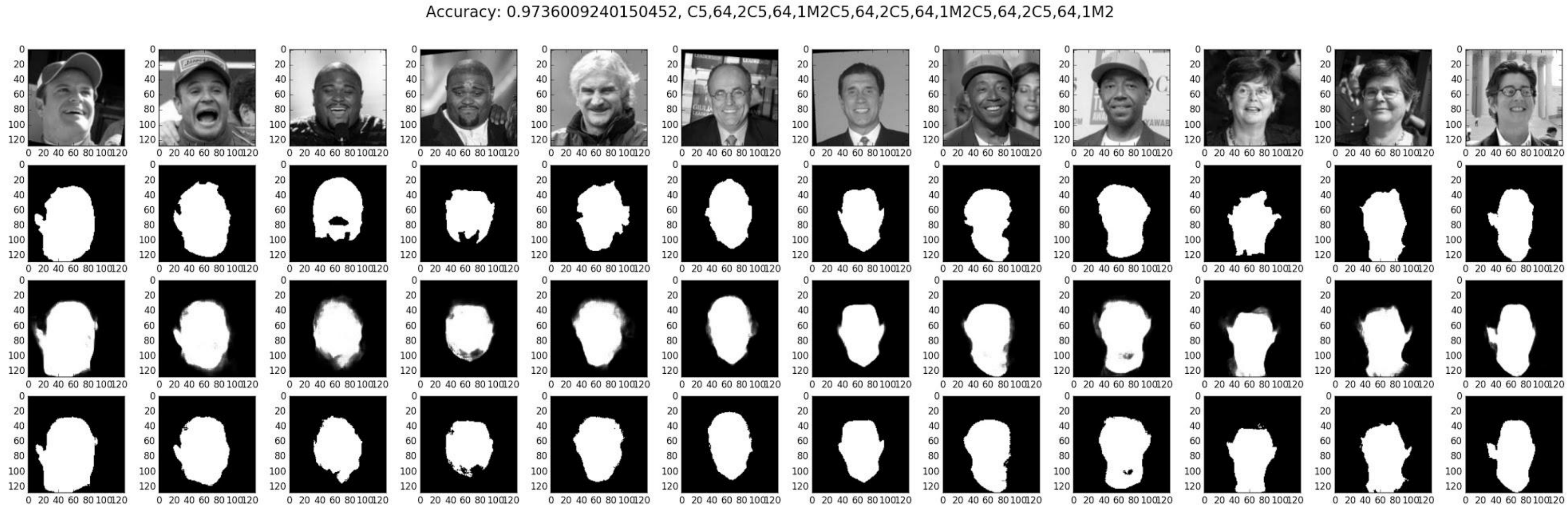
97.22 %

Accuracy: 0.9722825884819031, C7,64,2C7,64,1M2C7,64,2C7,64,1M2C7,64,2C7,64,1M2



Experiments and results

- The best… 97.36 %
  - kernels - 5x5



What did not help...

- CPU
- All strides = 1 (in conv layers)
- Batch-norm

Thank you!

Questions?