Convolutional Neural Networks for Semantic Segmentation

Advances in Deep Learning with Applications in Text and Image Processing

Giacomo Boracchi

giacomo.boracchi@polimi.it

February 14, 2019

Setting up the stage

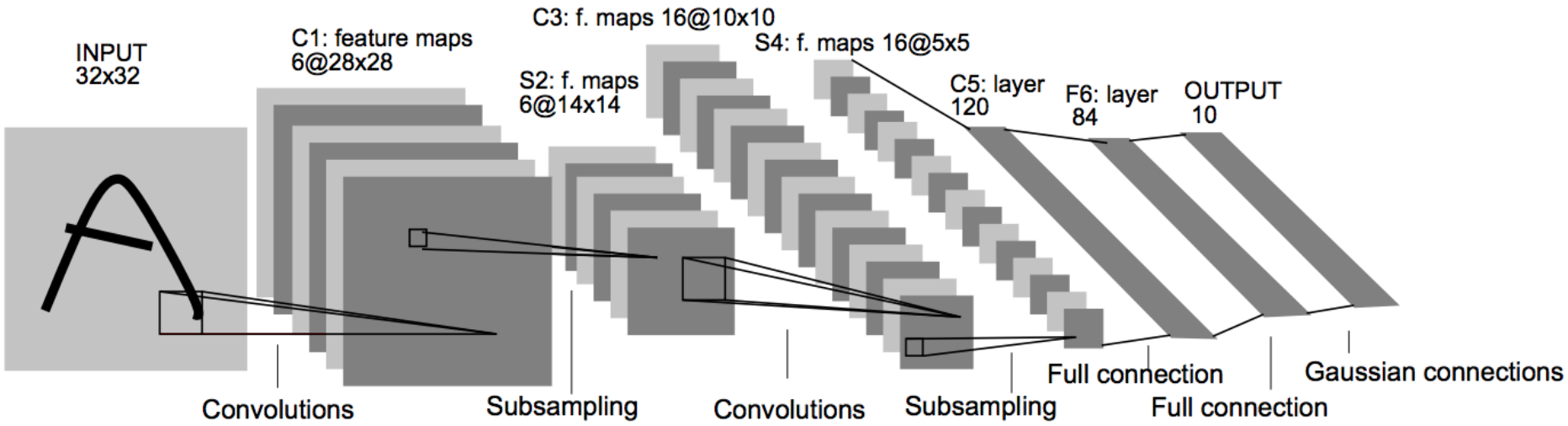
Convolutional neural networks

LeNet-5 (1998)

Yann LeCun's LeNet-5 model was developed in 1998 to identify handwritten digits for zip code recognition in the postal service.

This pioneering work introduced one of the most common CNN architecture, widely used today.

Parameters: 60.000, input size (32 x 32)



LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998)

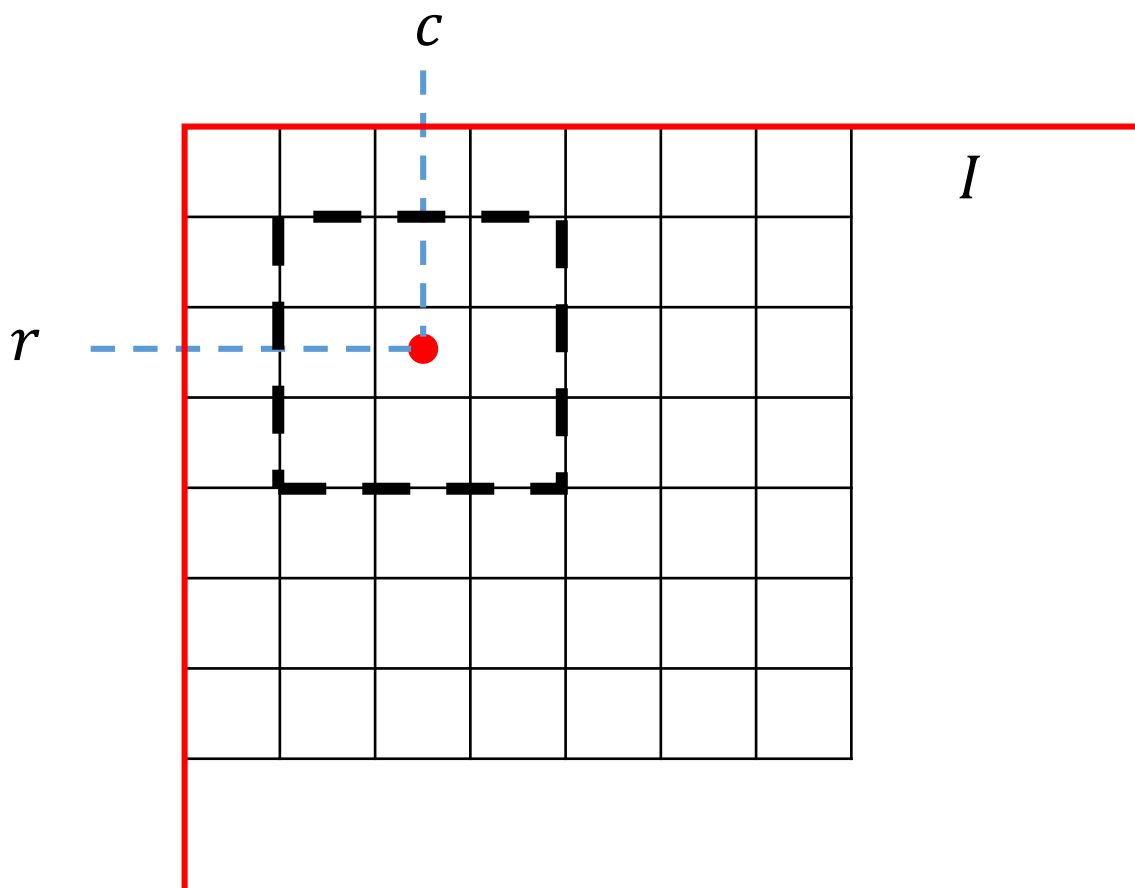
2D Convolution

Linear Transformation: Linearity implies that

$$T[I](r,c) = \sum_{(x,y)\in U} w(x,y) * I(r+x, c+y)$$

U

$w(-1,-1)$	$w(-1,0)$	$w(-1,1)$
$w(0,-1)$	$w(0,0)$	$w(0,1)$
$w(1,-11)$	$w(1,0)$	$w(1,1)$



We can consider weights as **a filter** h

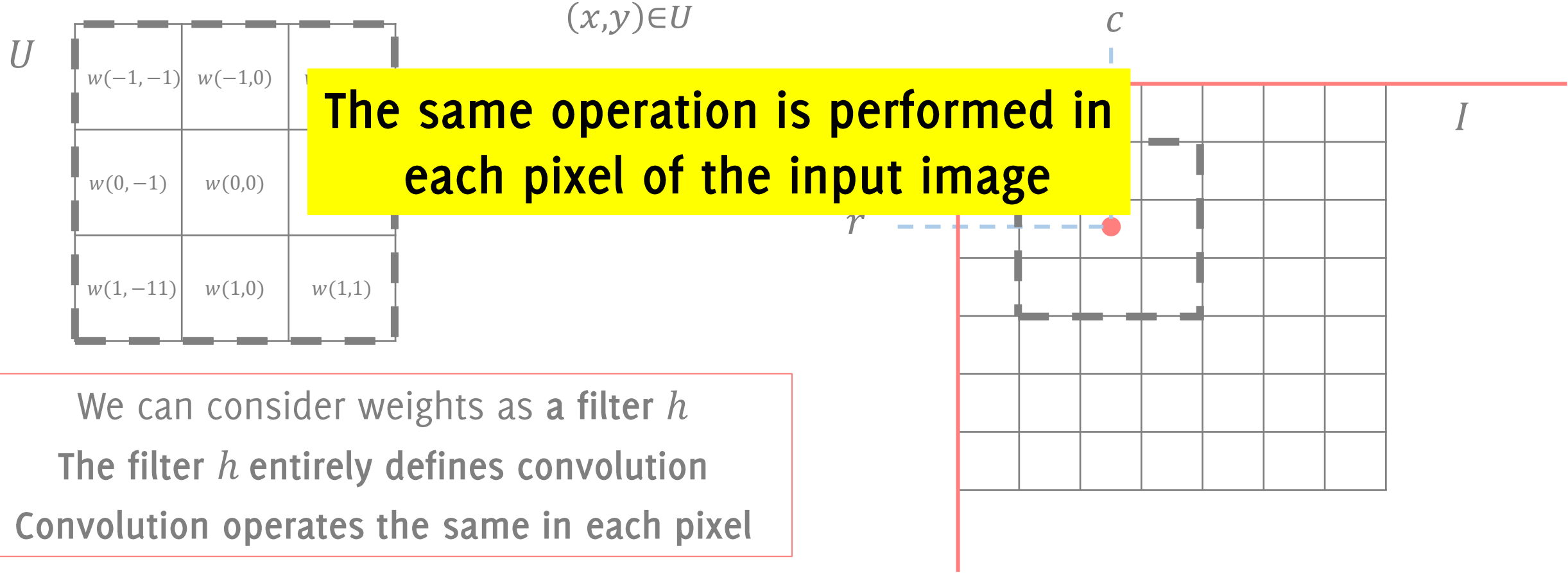
The filter h entirely defines convolution

Convolution operates the same in each pixel

2D Convolution

Linear Transformation: Linearity implies that

$$T[I](r,c) = \sum_{(x,y)\in U} w(x,y) * I(r+x, c+y)$$



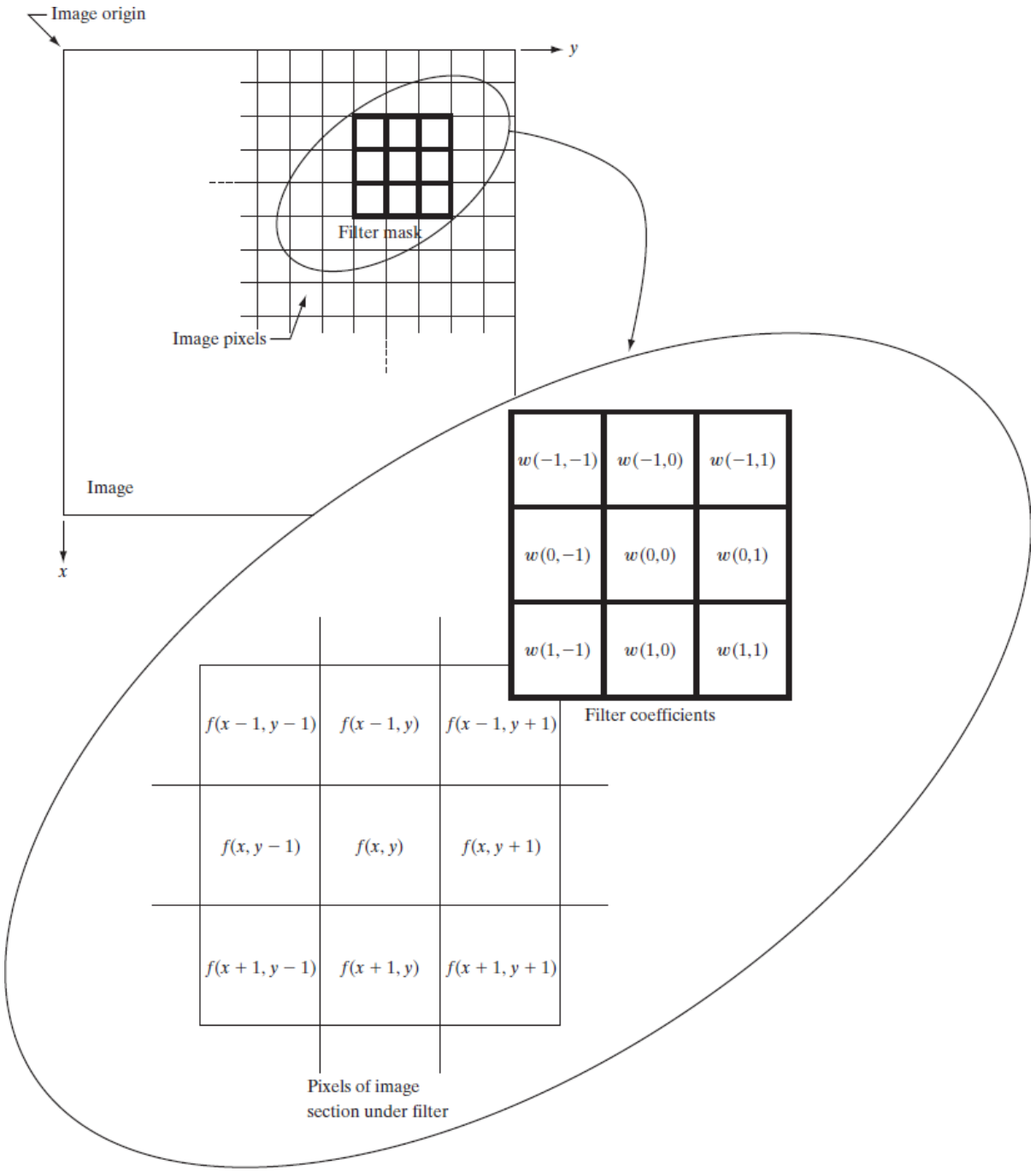
The same operation is performed in each pixel of the input image

We can consider weights as **a filter** h

The filter h **entirely defines convolution**

Convolution operates the same in each pixel

2D Convolution



Gonzalez and Woods «Digital image Processing», Prentice Hall;, 3° edition

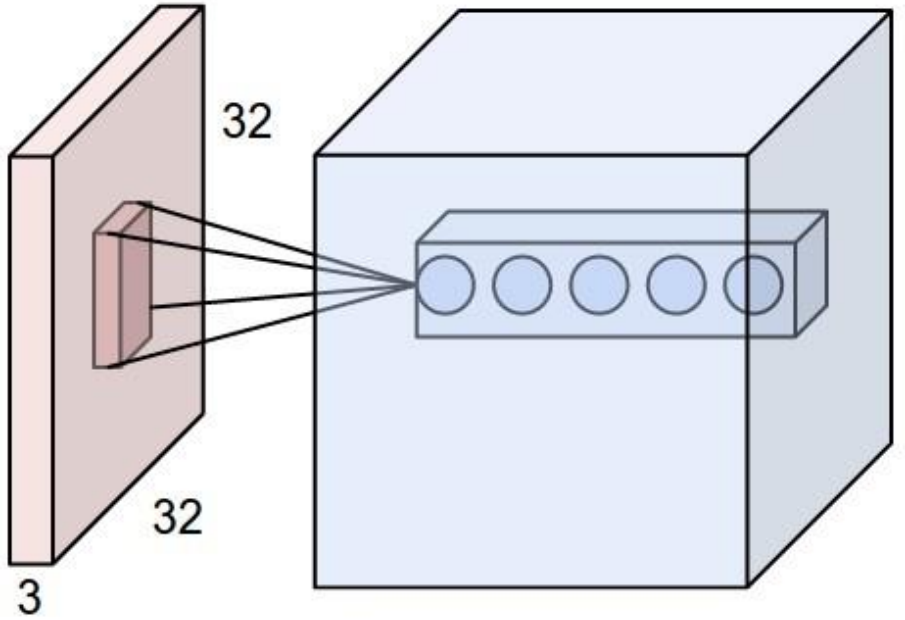
Convolutional Layers

Convolutional layers: compute feature maps as the convolution between a learnable filter and some volume (either the input image or the output of some previous layer).

In CNNs Convolutional filters are 3D and **mix all the components** in the input image (or volume of the previous layer).

The output is also called volume or activation maps

Each filter yields a different slice of the output volume



By Aphex34 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=45659236

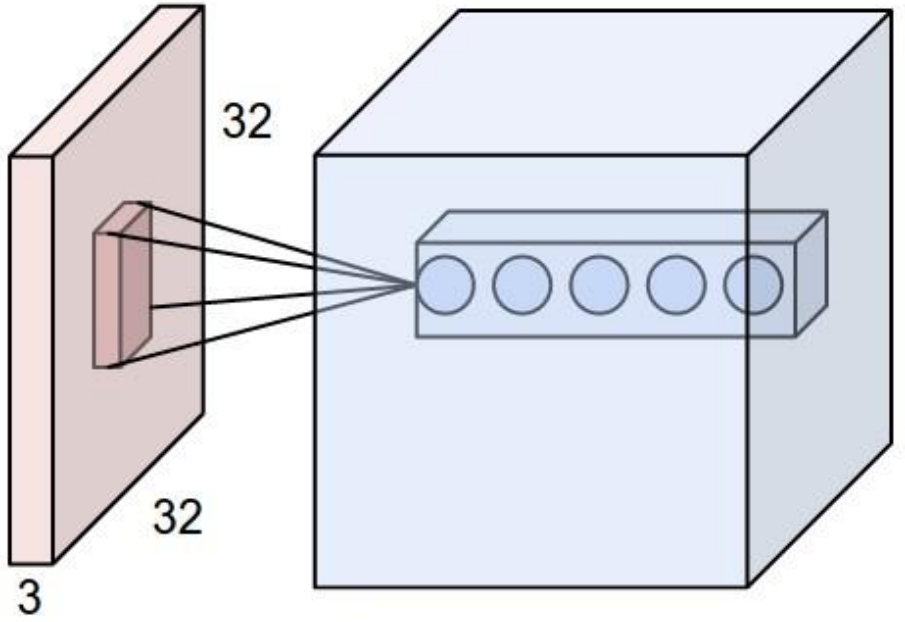
Convolutional Layers

Remarks:

Convolutional Layers have very small spatial extent

Very deep 3D extent, as they cover the whole volume fed to the layer

The output of the convolution against a filter becomes a slice in the volume feed to the next layer



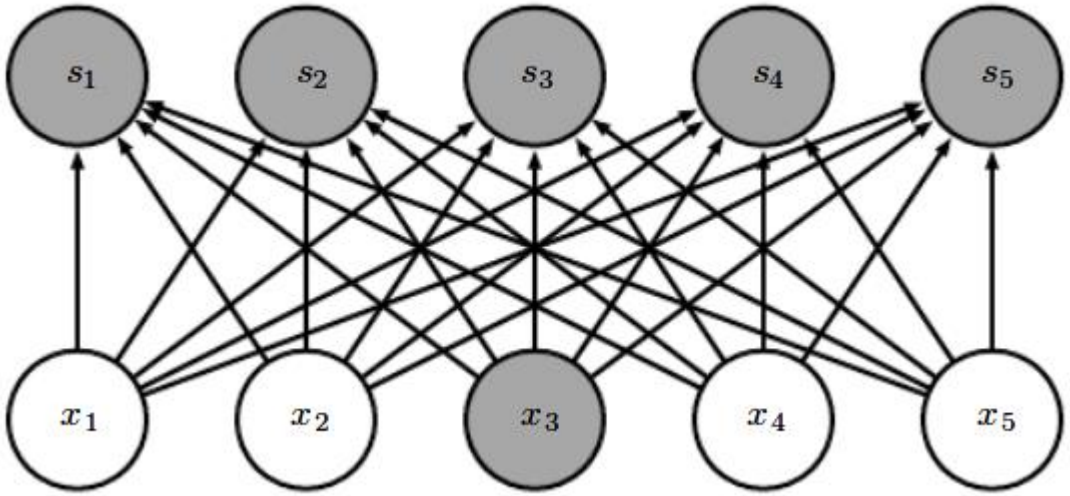
By Aphex34 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=45659236

Convolutions as MLP

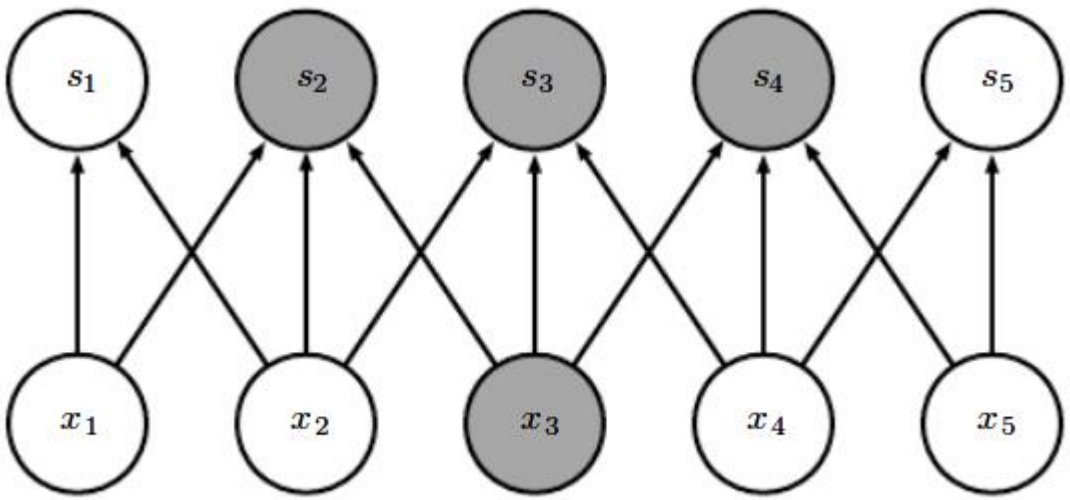
If you unroll the input image to a vector, you can consider convolution weights as the weights of a Multilayer Perceptron Network

Sparse connectivity

Fully connected



3x1 convolutional

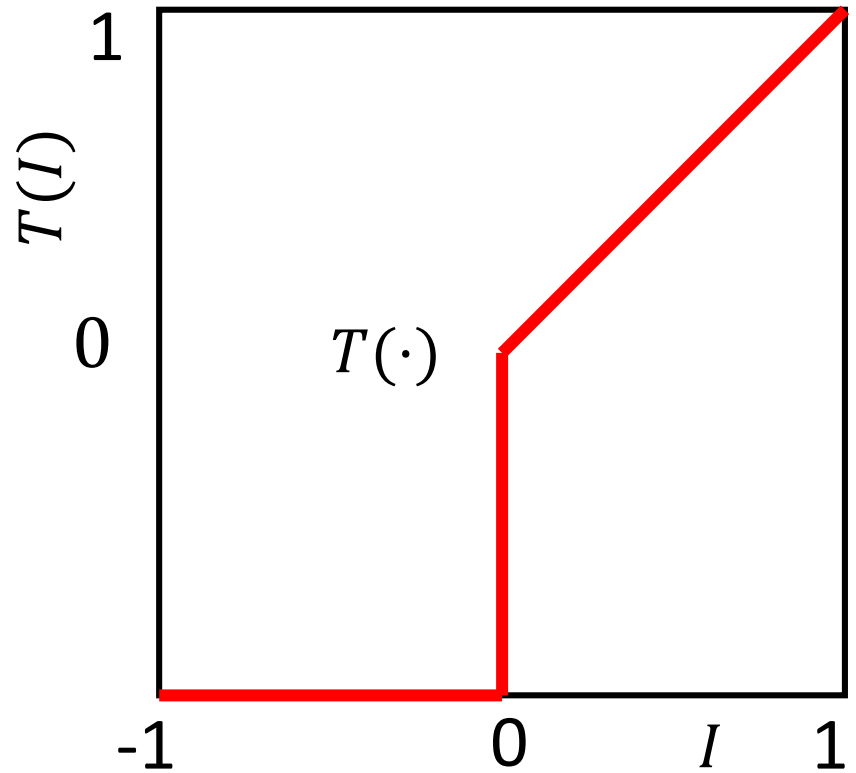


Activation Layers

Introduce nonlinearities in the network, otherwise the CNN might be equivalent to a linear classifier…

RELU (Rectifier Linear Units): normalize the feature maps e.g. by $\max(0,\cdot)$ operator.

$$T\big(I(r,c)\big) = \begin{cases} T\big(I(r,c)\big), & \text{if } I(r,c) \geq I(r,c) \\ 0, & \text{if } I(r,c) < 0 \end{cases}$$

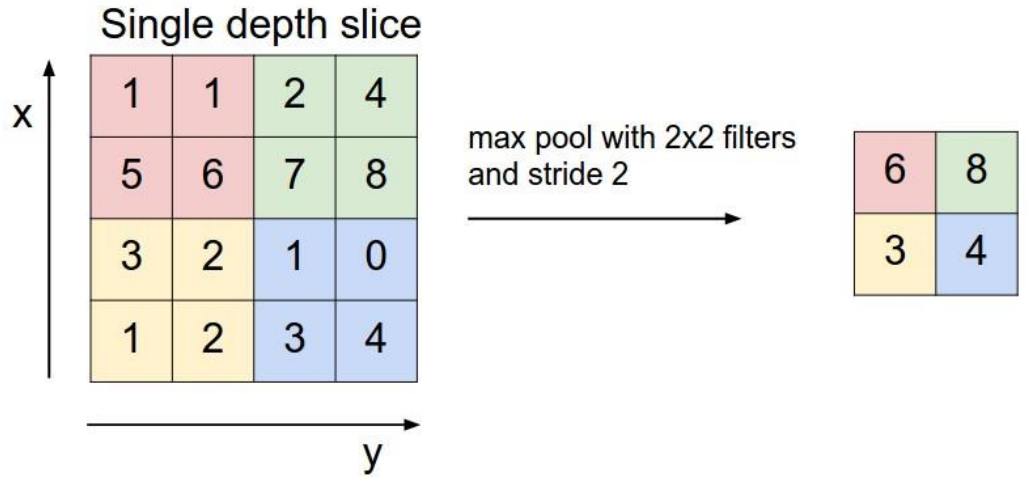


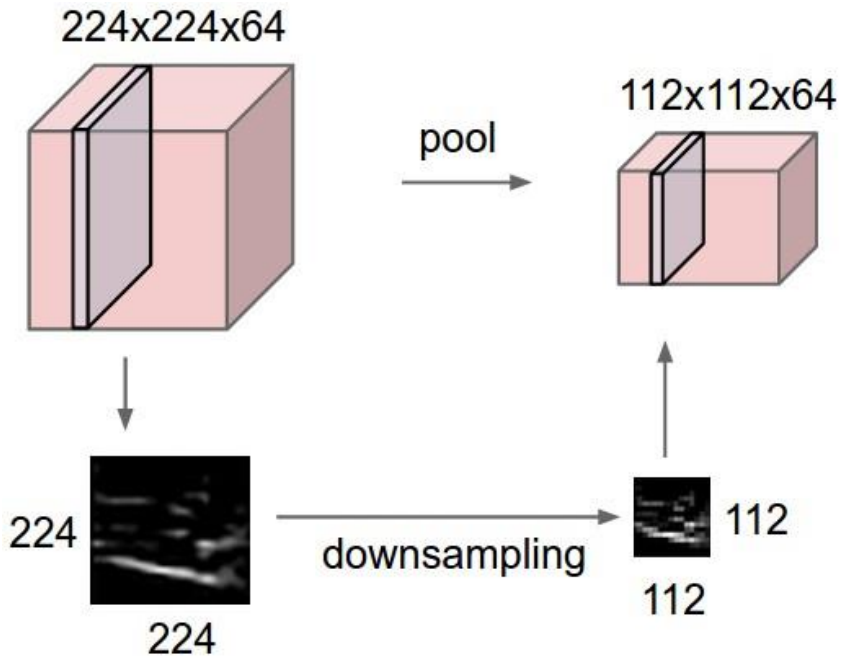
Pooling Layers

Pooling Layers are downsampling layers used to reduce the **spatial** size of the volume.

The Pooling Layer operates **independently on every depth slice** of the input and **resizes it spatially, often using the MAX operation.**

Typically it discards 75% of samples in a volume

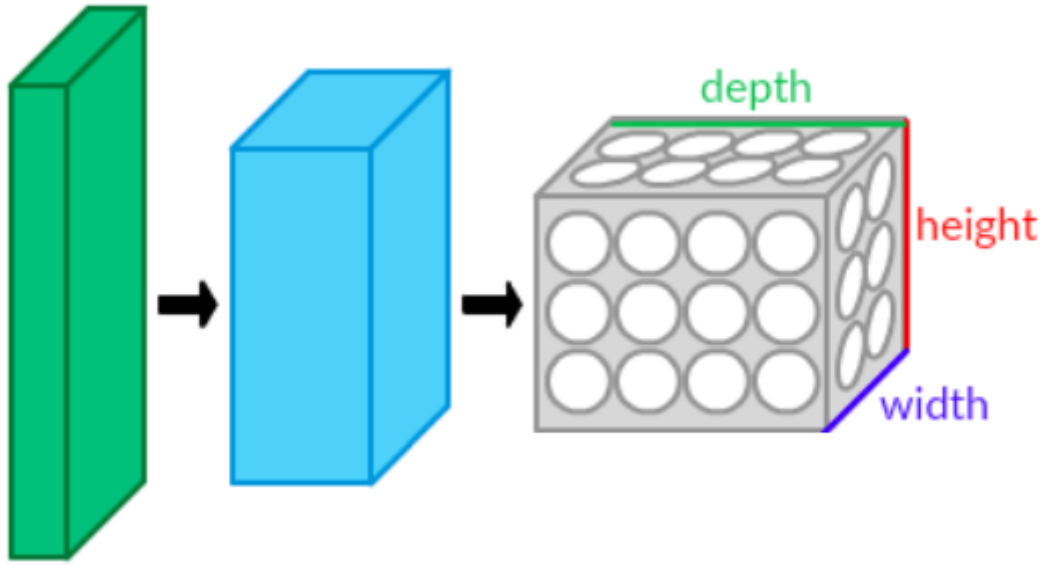




CS231n: Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/

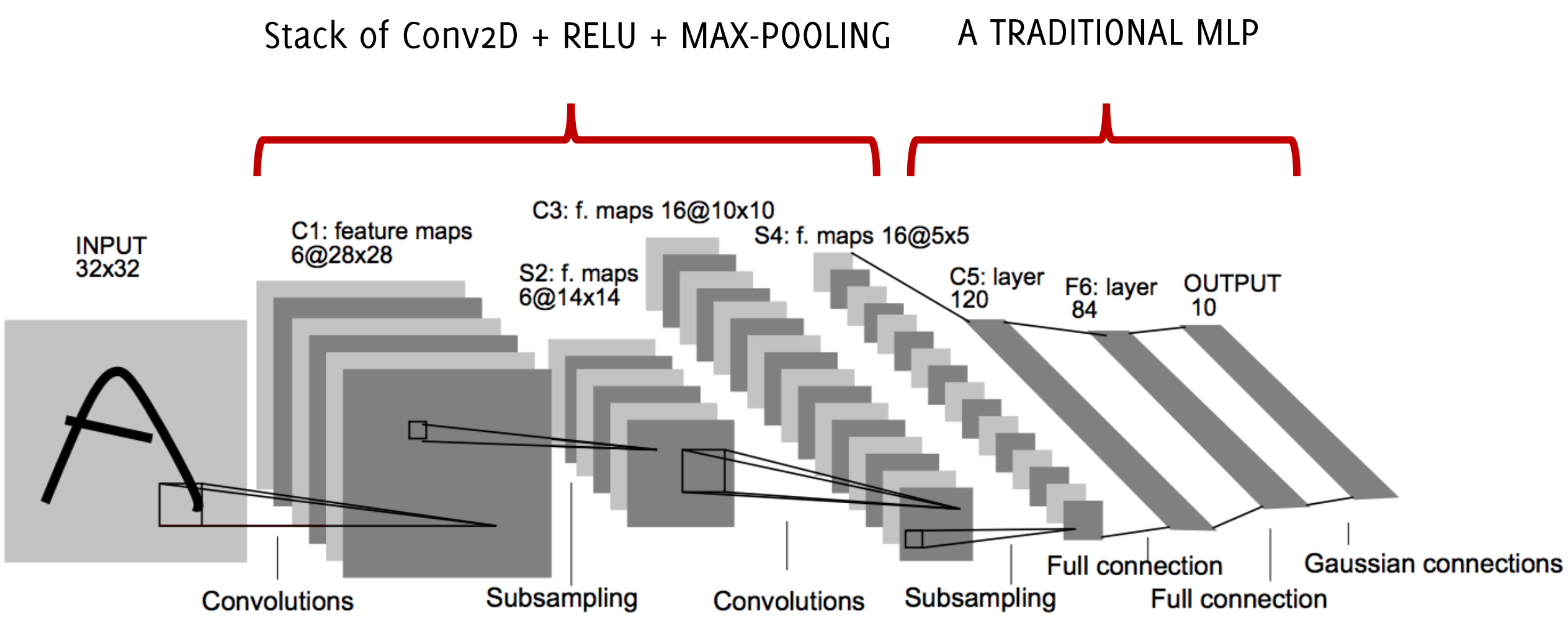
Convolutional Neural Networks (CNN)

The input/output of each layer is a volume, typically having decreasing height and width and increasing depth.



By Aphex34 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=45661858

LeNet-5 (1998)

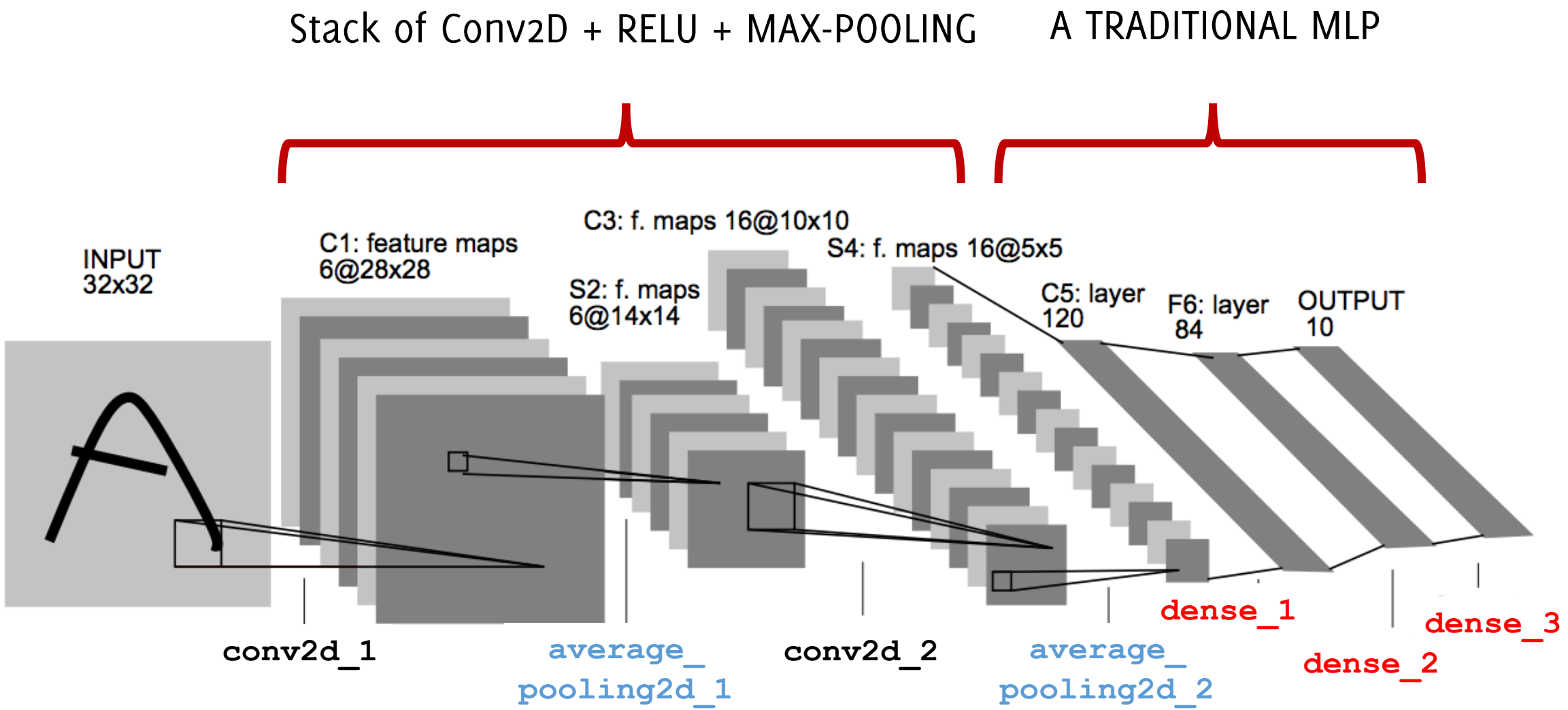


LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998)

model.summary()

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 6)         156
_________________________________________________________________
average_pooling2d_1 (Average (None, 14, 14, 6)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 16)        2416
_________________________________________________________________
average_pooling2d_2 (Average (None, 5, 5, 16)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 400)               0
_________________________________________________________________
dense_1 (Dense)              (None, 120)               48120
_________________________________________________________________
dense_2 (Dense)              (None, 84)                10164
_________________________________________________________________
dense_3 (Dense)              (None, 10)                850
=================================================================
Total params: 61,706
Trainable params: 61,706
Non-trainable params: 0
```

LeNet-5 (1998)



LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998)

Most of parameters are in MLP

What about a MLP taking as input the whole image?

Input 32 x 32 = 1024 pixels, fed to a 84 neurons -> 86950 parameters

But.. If you take an RGB input: 32 x 32 x 3,

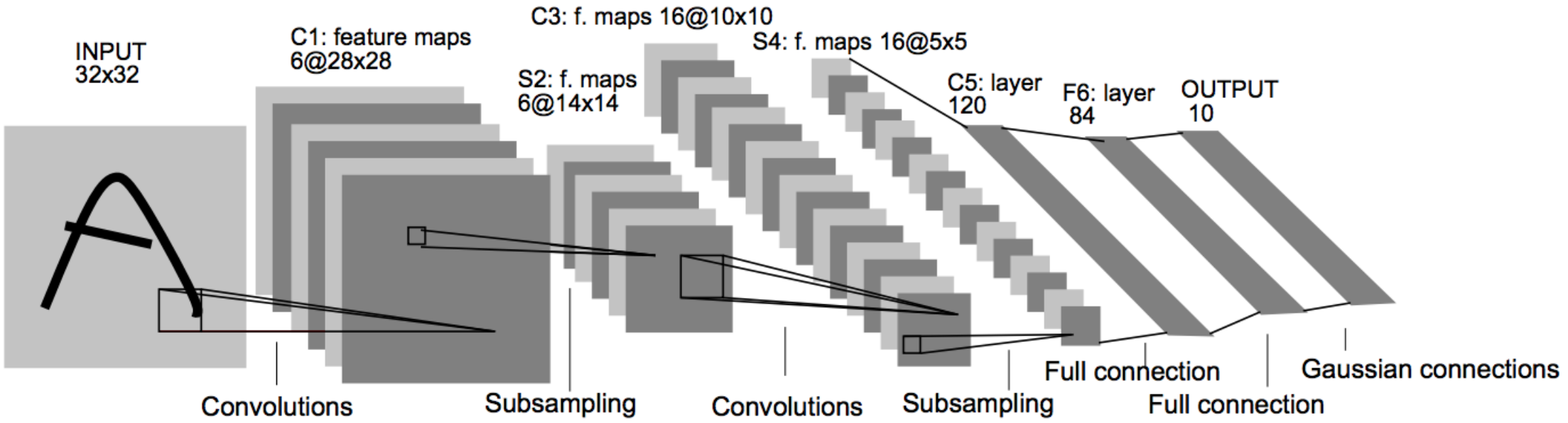
CNN: only the nr. of parameters in the first filters increases 156 -> 456

MLP: everything increases by a factor 3

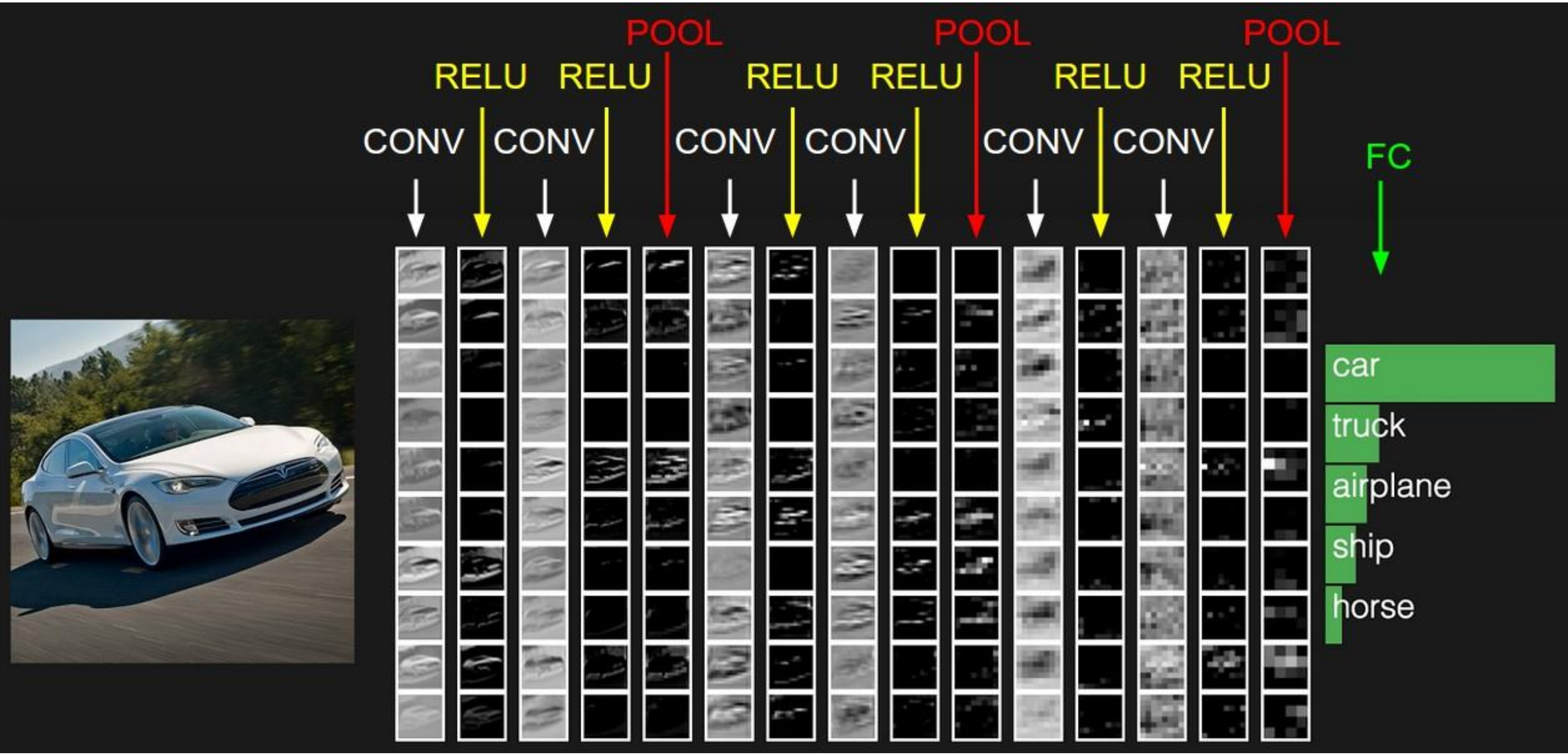
Weight Sharing / Spatial Invariance

In a CNN, all the neurons in the same depth slice use the same weights and bias: this dramatically reduce the nr. of parameters in the CNN.

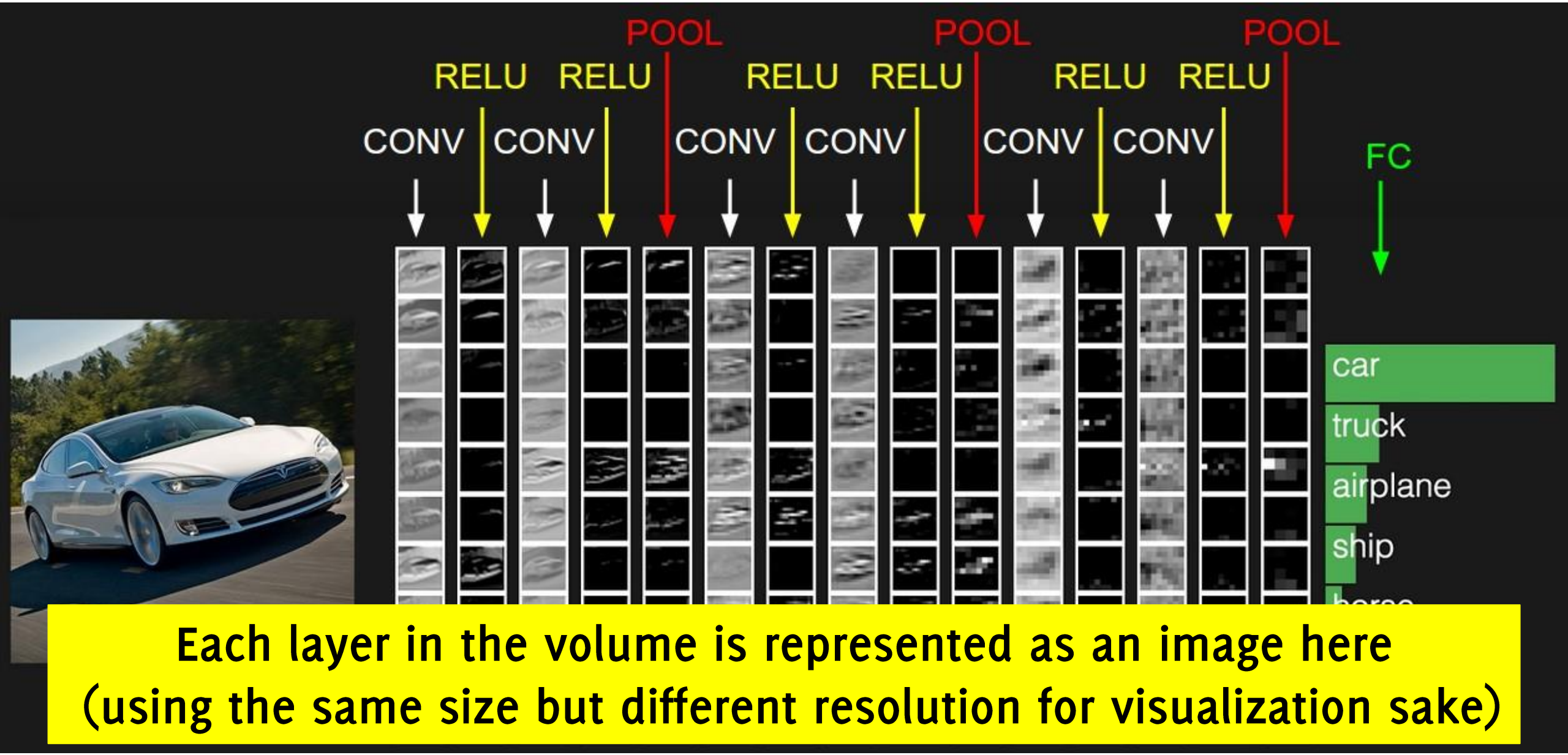
Underlying assumption: **if one feature is useful** to compute **at some spatial position** (x,y), **then it should also** be useful to compute at a **different position** (x2,y2)



Activations in a convolutional network

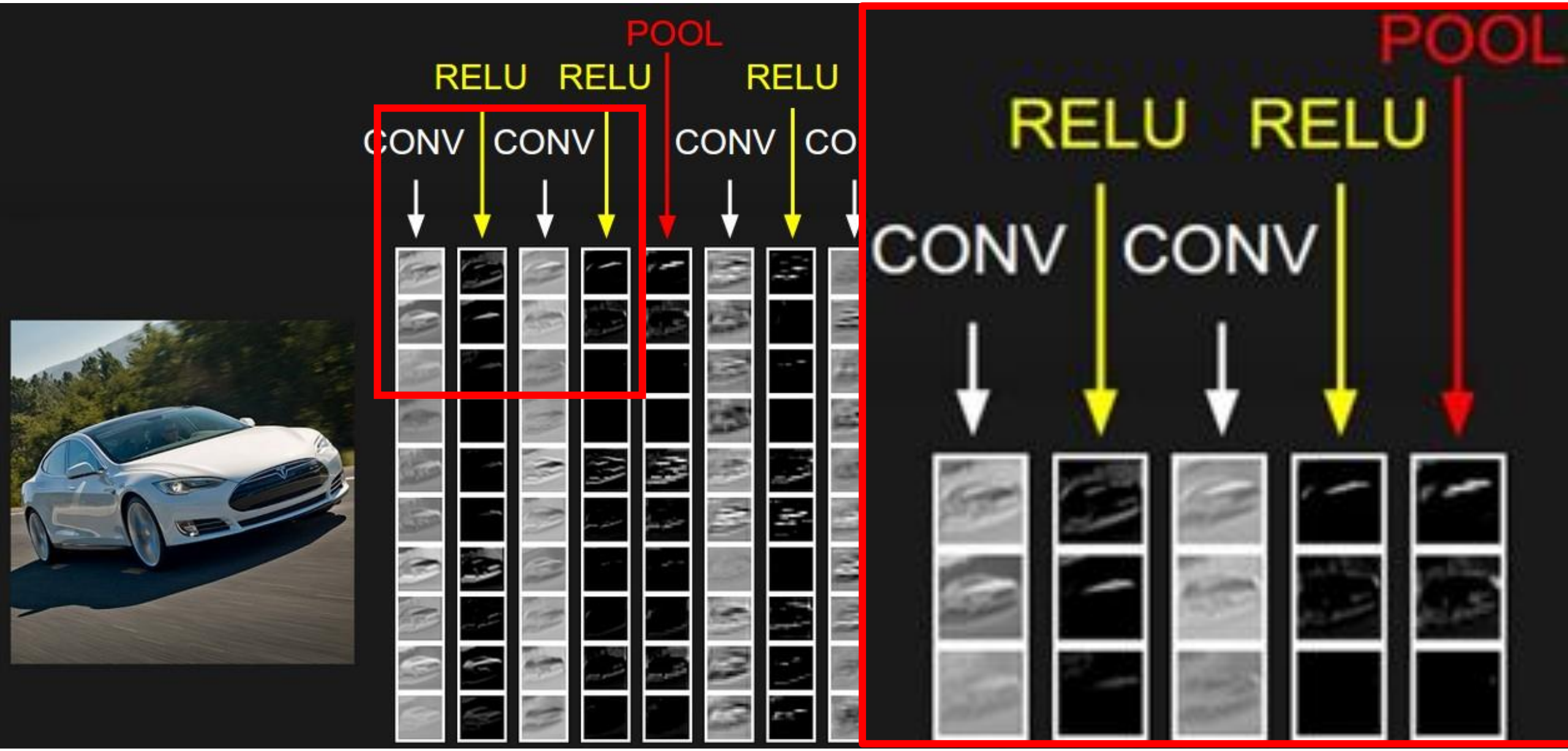


Activations in a convolutional network

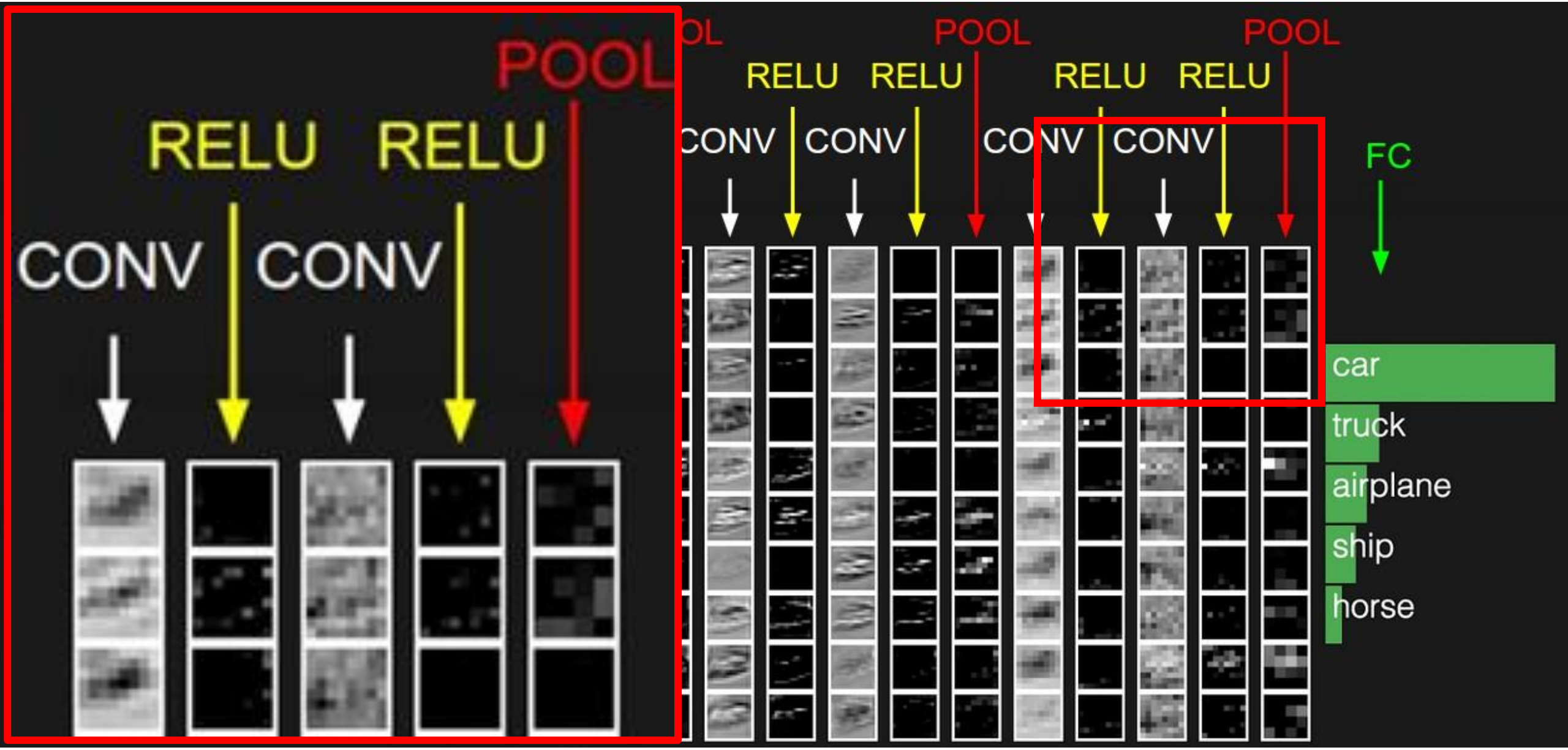


Each layer in the volume is represented as an image here
(using the same size but different resolution for visualization sake)

Activations in a convolutional network



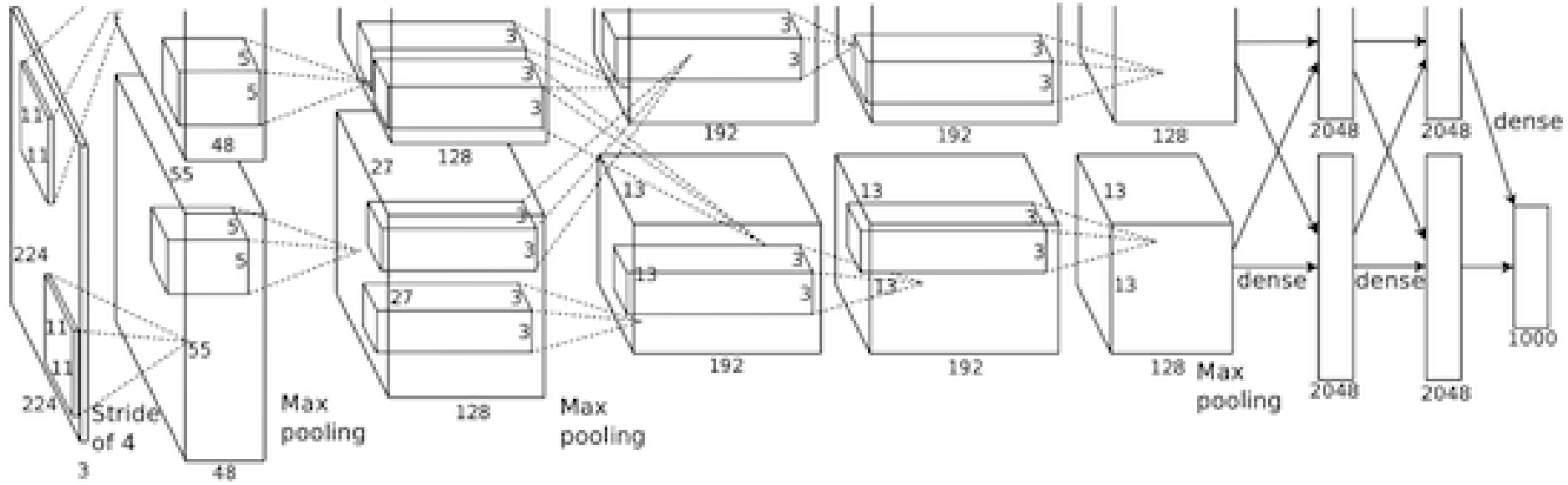
Activations in a convolutional network



AlexNet (2012)

AlexNet was developed by Alex Krizhevsky et al. in 2012 to compete in the ImageNet competition. The general architecture is quite similar to LeNet-5, although this model is considerably larger. It won the ImageNet 2012 competition

Parameters: 60 million



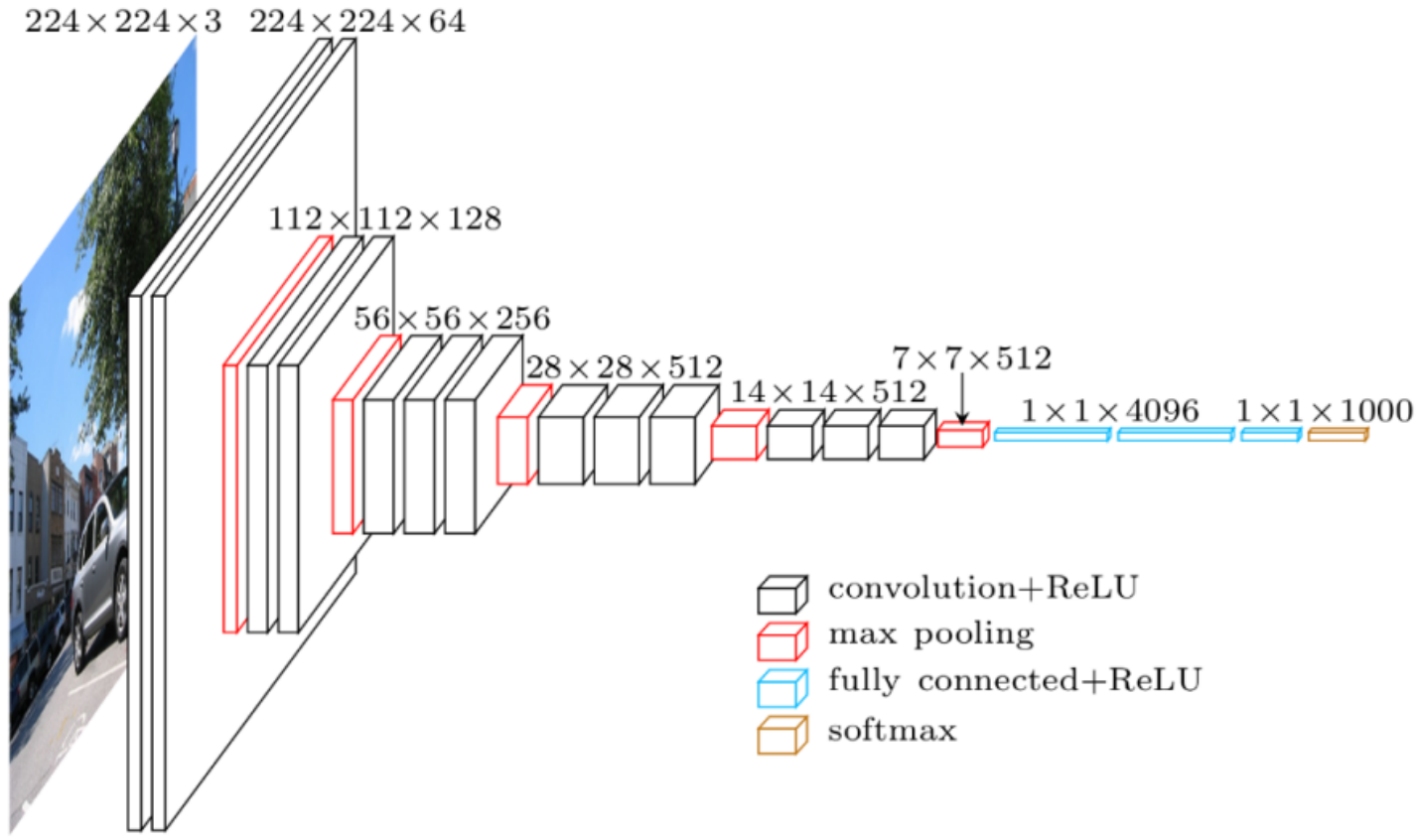
Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012.

VGG16 (2014)

The VGG16, introduced in 2014 is a deeper variant of the AlexNet convolutional structure

Parameters: 138 million

Considers small filters: Multiple 3×3 convolution in sequence can emulate the effect of larger receptive fields, for examples 5×5 and 7×7.



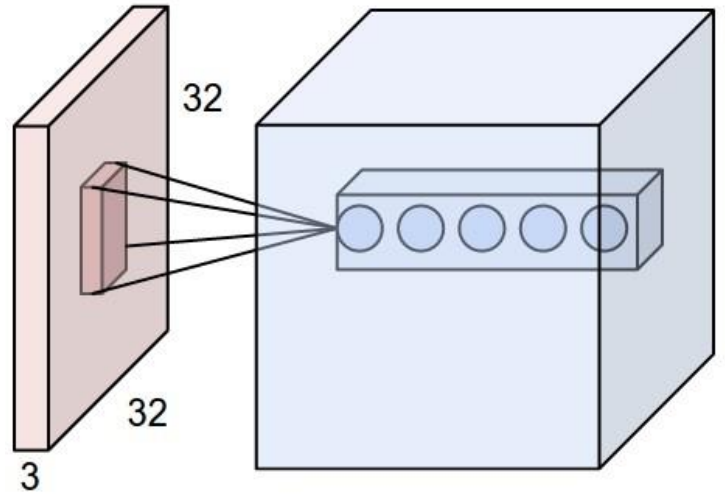
Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

The Receptive field

In a CNN each neuron connects to a local region of the input volume: the **receptive field** of the neuron.

The deeper you go, the wider the receptive field: maxpooling, convolutions and stride > 1 increase the receptive field

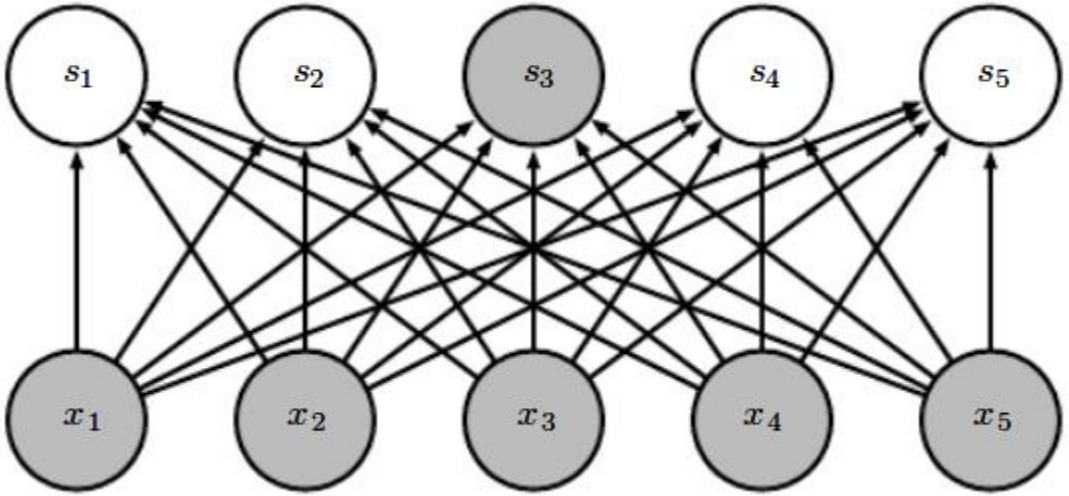
Usually, the receptive field refers to the final output unit of the network in relation to the network input



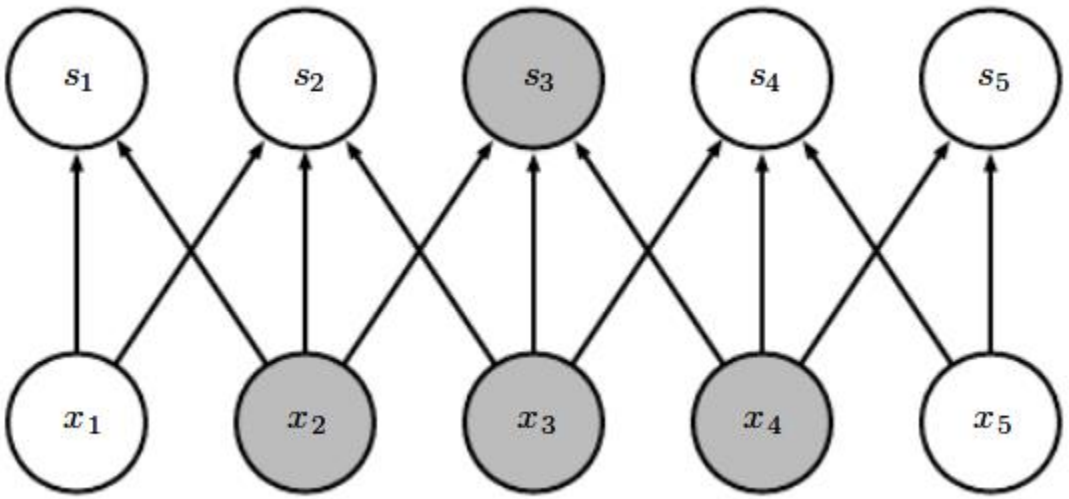
By Aphex34 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=45659236

Receptive fields

Fully connected



3x1 convolutional

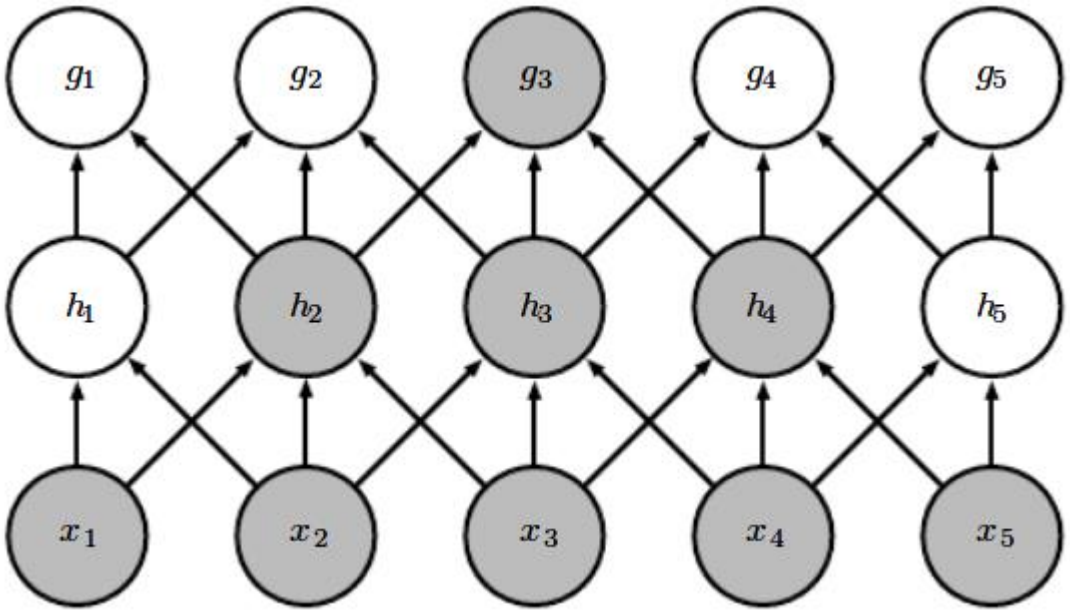


Receptive fields

Deeper neurons depend on wider patches of the input (convolution is enough to increase receptive field, no need of maxpooling)

3x1 convolutional

3x1 convolutional



As we move deeper…

As we move to deeper layers:

- spatial resolution is reduced
- the number of maps increases

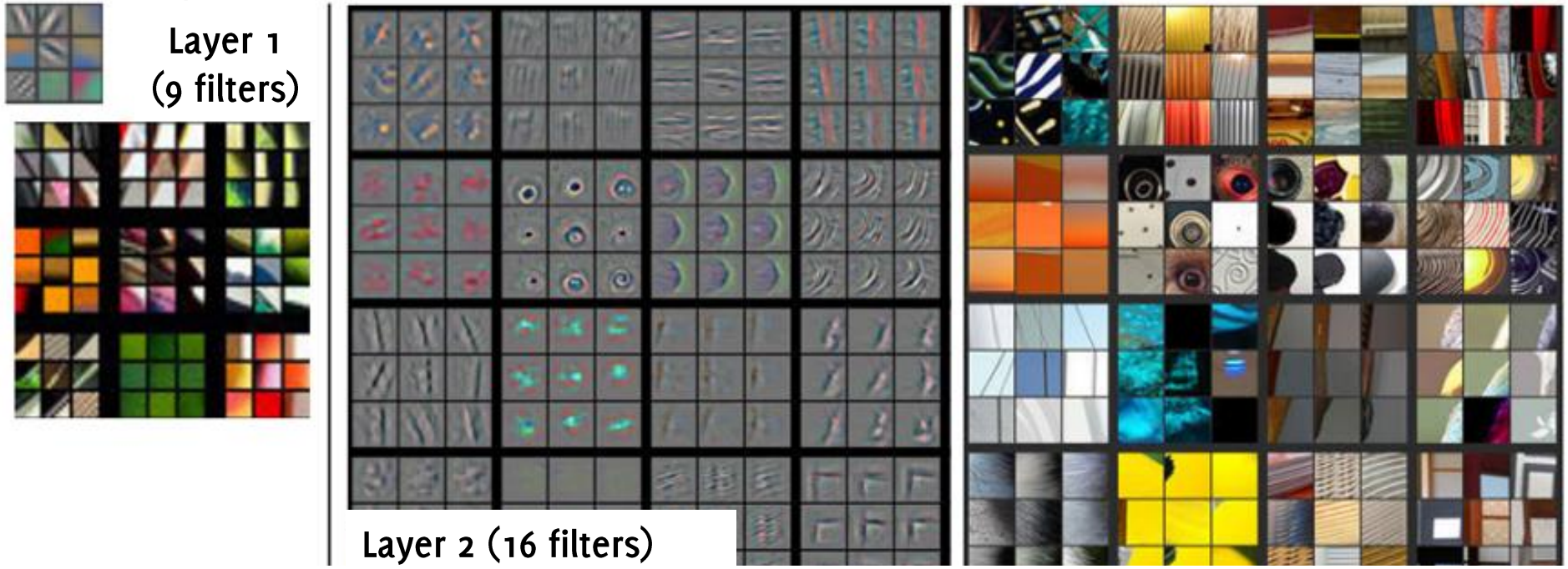
We search for higher-level patterns, and don’t care too much about their exact location.

There are more high-level patterns than low-level details!

Feature Maps Visualization

Convolutional layers: compute **feature maps** as the convolution between a learnable filter and some volume (either the input image or the output of some previous layer).

Learned filters represent the features that are most important for image classification. Here there is an example of filters and examples of image regions that are strongly activated by these



ZFNet: Zeiler and Fergus “Visualizing and Understanding Convolutional Networks”

Semantic Segmentation

Semantic Segmentation Task

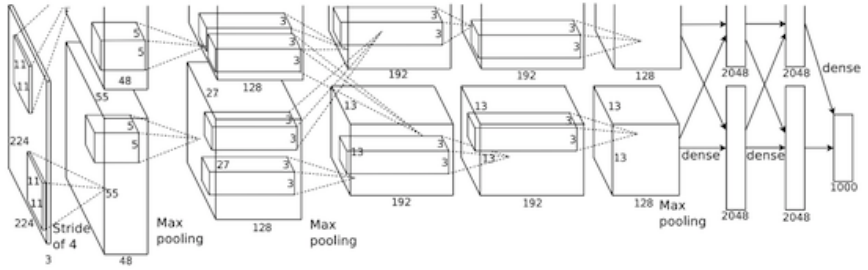
http://www.robots.ox.ac.uk/~szheng/crfasrnndemo

Zheng et al. *"Conditional Random Fields as Recurrent Neural Networks"*, ICCV 2015

The Straightforward Solution

1000 x 2000 pixels

- A pretrained model is meant to process a fixed input size (e.g. 224 x 224)
- Slide on the image a window of that size and classify each region.
- Assign the predicted label to the central pixel

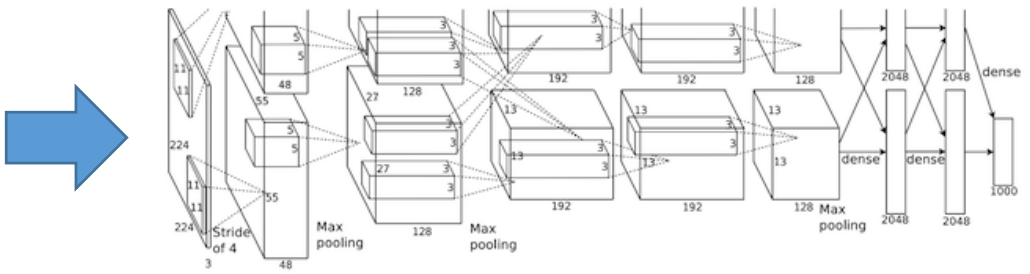


car

The Straightforward Solution

1000 x 2000 pixels

- A pretrained model is meant to process a fixed input size (e.g. 224 x 224)
- Slide on the image a window of that size and classify each region.
- Assign the predicted label to the central pixel



wheel

Many drawbacks…

Cons:

- **Very inefficient!** Does not re-use features that are «shared» among overlapping crops
- Not able to capture details at different scale!

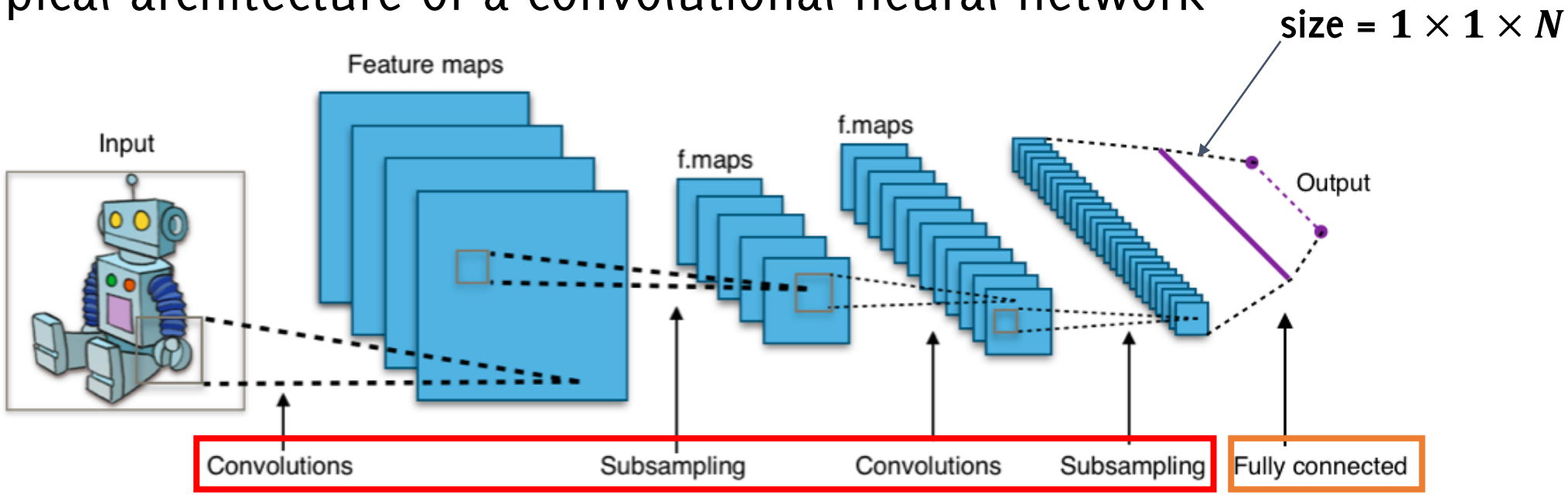
Plus:

- No need of retraining the CNN

What happens when chaning the image size?

Convolutional Neural Networks (CNN)

The typical architecture of a convolutional neural network



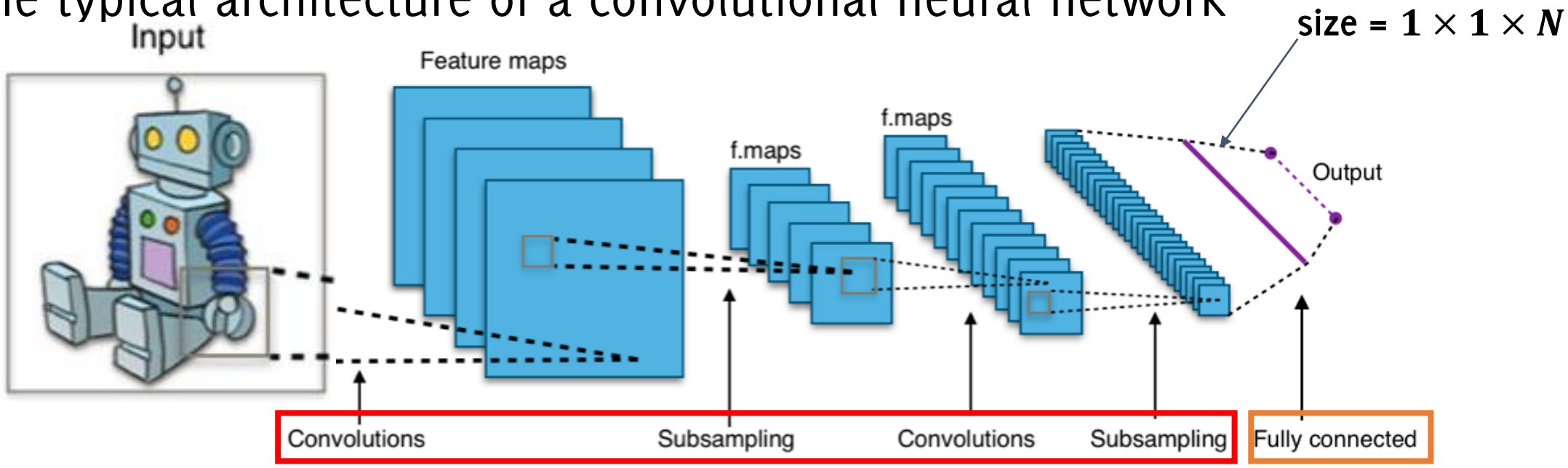
CNNs are meant to process input of a fixed size (e.g. 200 x 200).

The **convolutional and subsampling layers** operate in a sliding manner over image having arbitrary size

The **fully connected** layer constrains the input to a fixed size.

Convolutional Neural Networks (CNN)

The typical architecture of a convolutional neural network



What happens when we feed a larger image to the network?

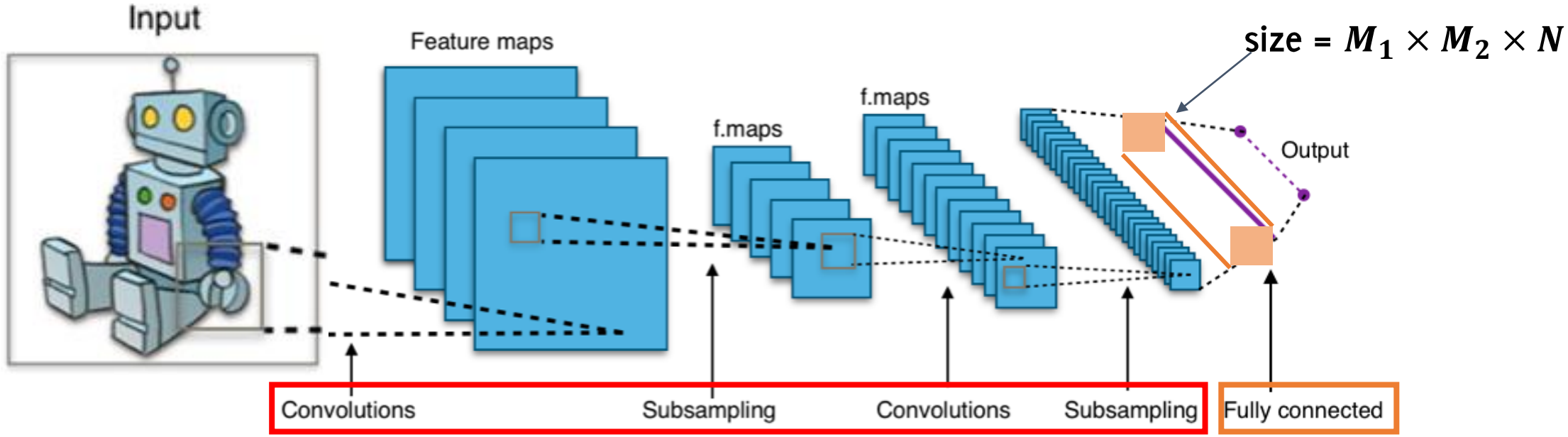
By Aphex34 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=45679374

Convolutional Neural Networks (CNN)

Convolutional filters can be applied to volumes of any size, yielding larger volumes in the network until the FC layer.

The FC network however requires a fixed input size

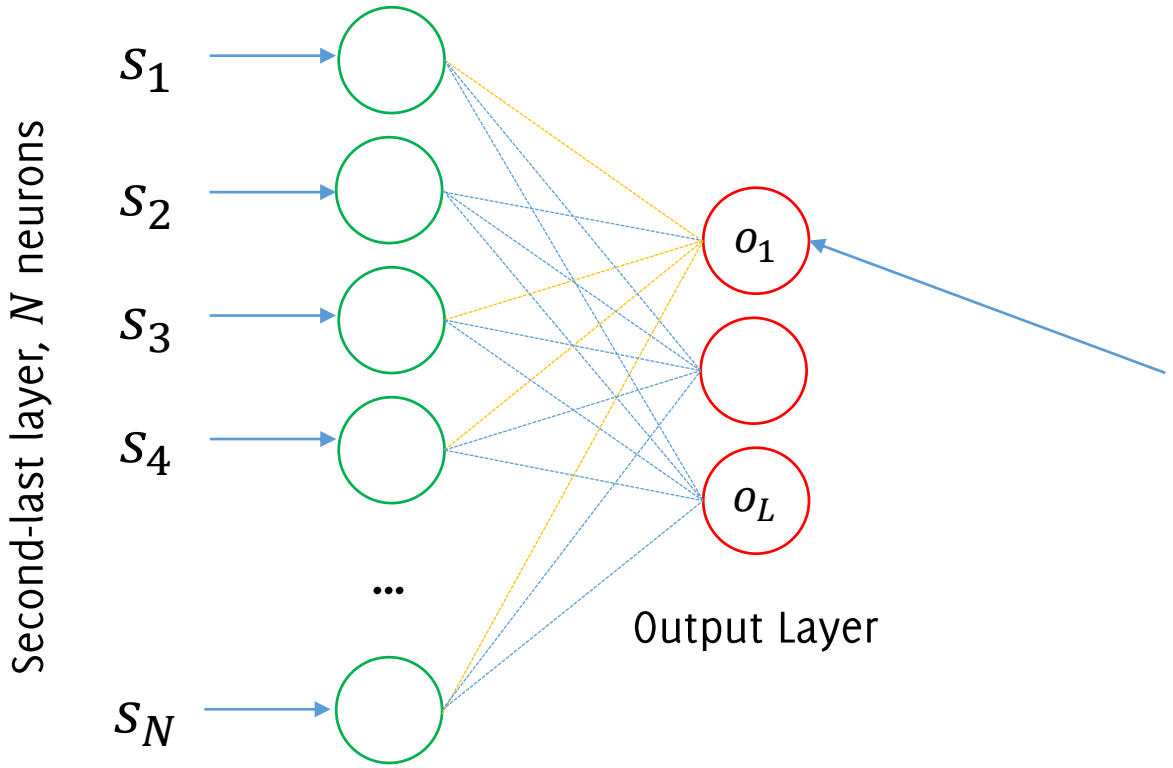
Thus, CNN cannot compute class scores, yet can extract features!



Fully Convolutional Neural Networks (f-CNN)

However, since the FC is linear, it can be represented as convolution!

Weights associated to output neuron i : $\boldsymbol{w_i} = \{w_{i,j}\}_{i=j:N}$



$$o_i = \sum_{j=1:N} w_{i,j}\, s_j + b_i$$

$$o_i = \boldsymbol{w_i'}\, \boldsymbol{s} + b_i$$

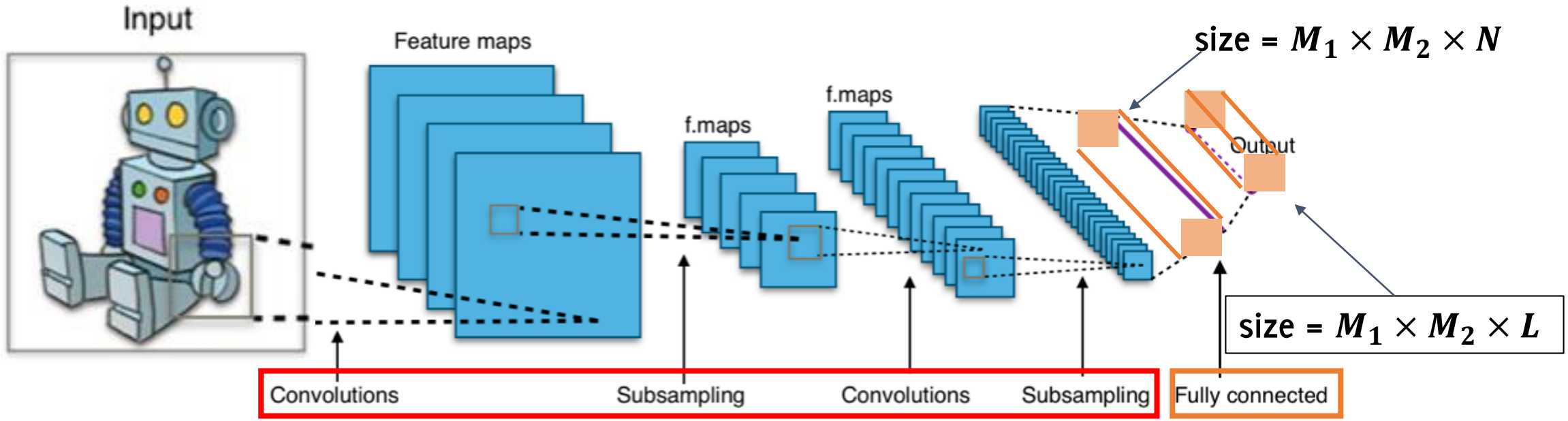
$$o_i = (\boldsymbol{w_i} \circledast \boldsymbol{s})(0,0) + b_i$$

A FC layer of L outputs is a 2DConv against $\boldsymbol{L}$ **filters of size** $\mathbf{1 \times 1 \times N}$

Fully Convolutional Neural Networks (f-CNN)

However, since the FC is linear, it can be represented as convolution against L filters of size $1 \times 1 \times N$

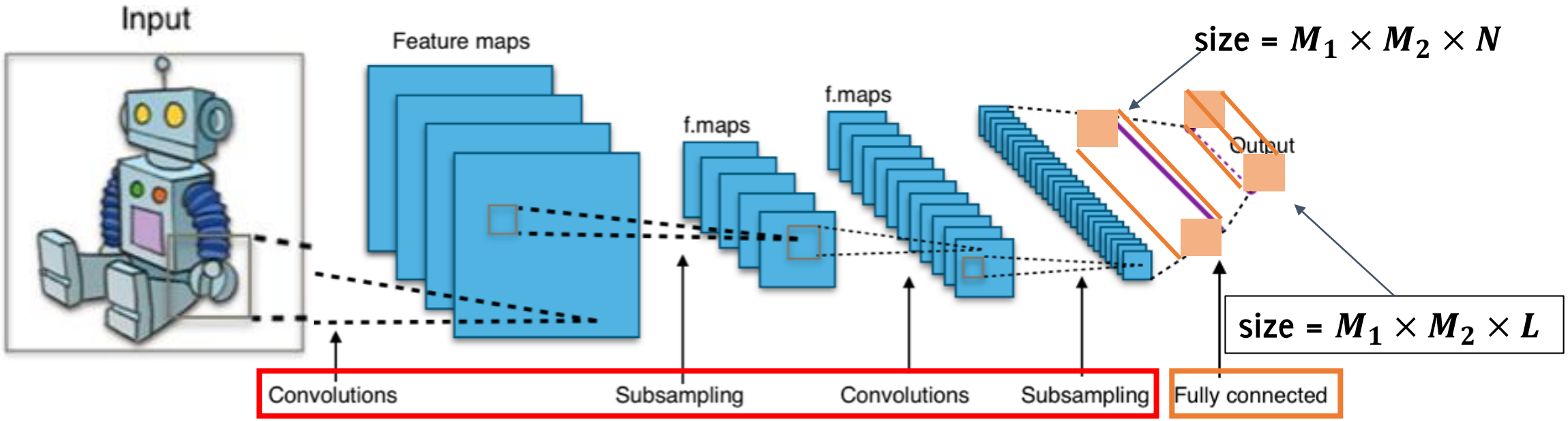
Each of these convolutional filters contains the weights of the FC for the corresponding output neuron



Long, J., Shelhamer, E., Darrell, T. "Fully convolutional networks for semantic segmentation". CVPR 2015

Fully Convolutional Neural Networks (f-CNN)

The CNN becomes **fully convolutional FCNN!**

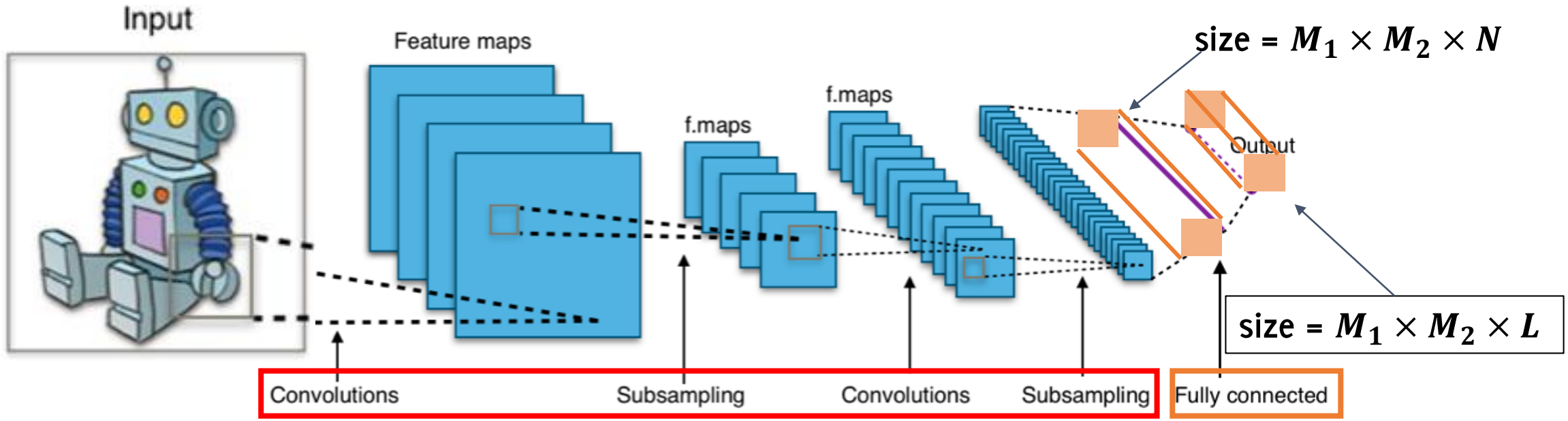


Long, J., Shelhamer, E., Darrell, T. "Fully convolutional networks for semantic segmentation". CVPR 2015

Fully Convolutional Neural Networks (f-CNN)

For each output class we obtain an image, having:

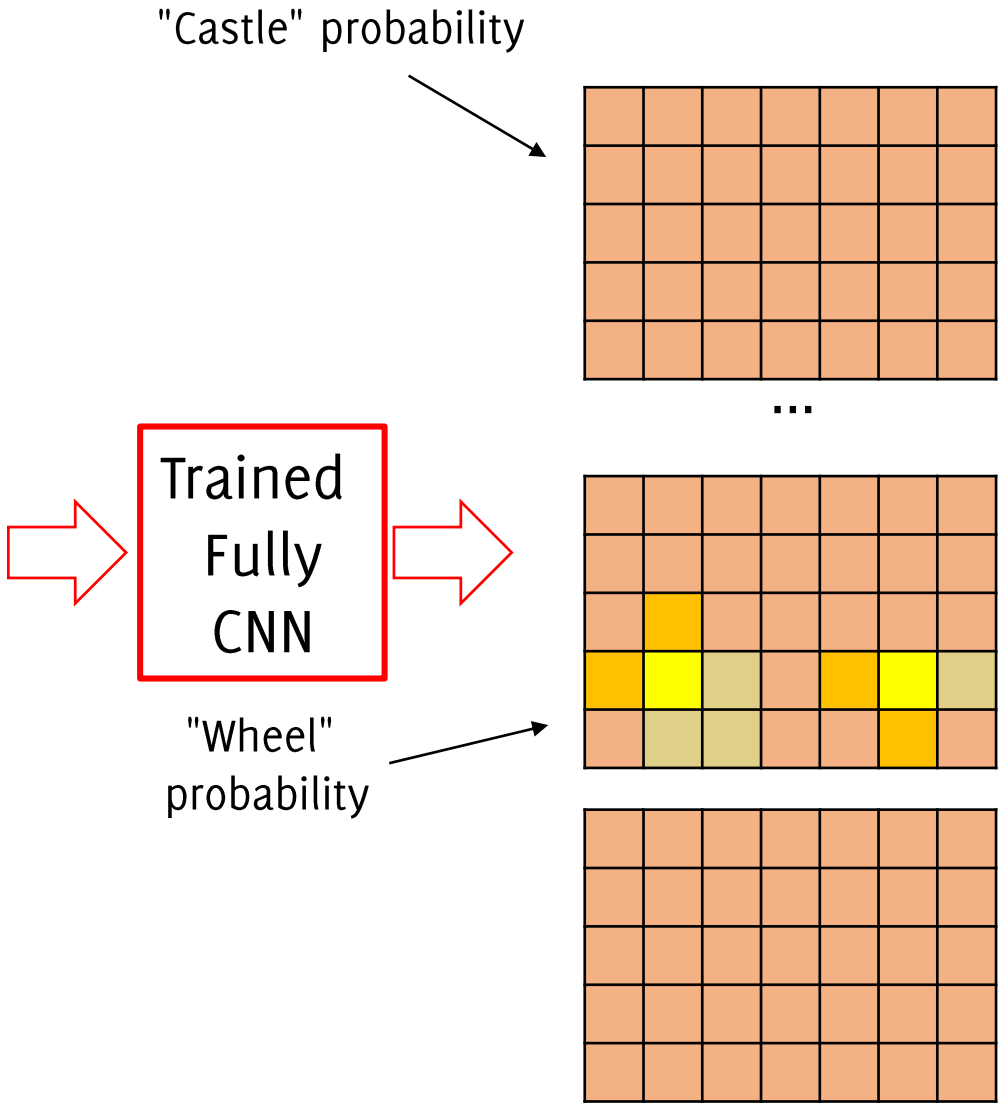
- **Lower resolution** than the input image
- Containing **class probabilities** for the image region referring to each pixel (receptive field)



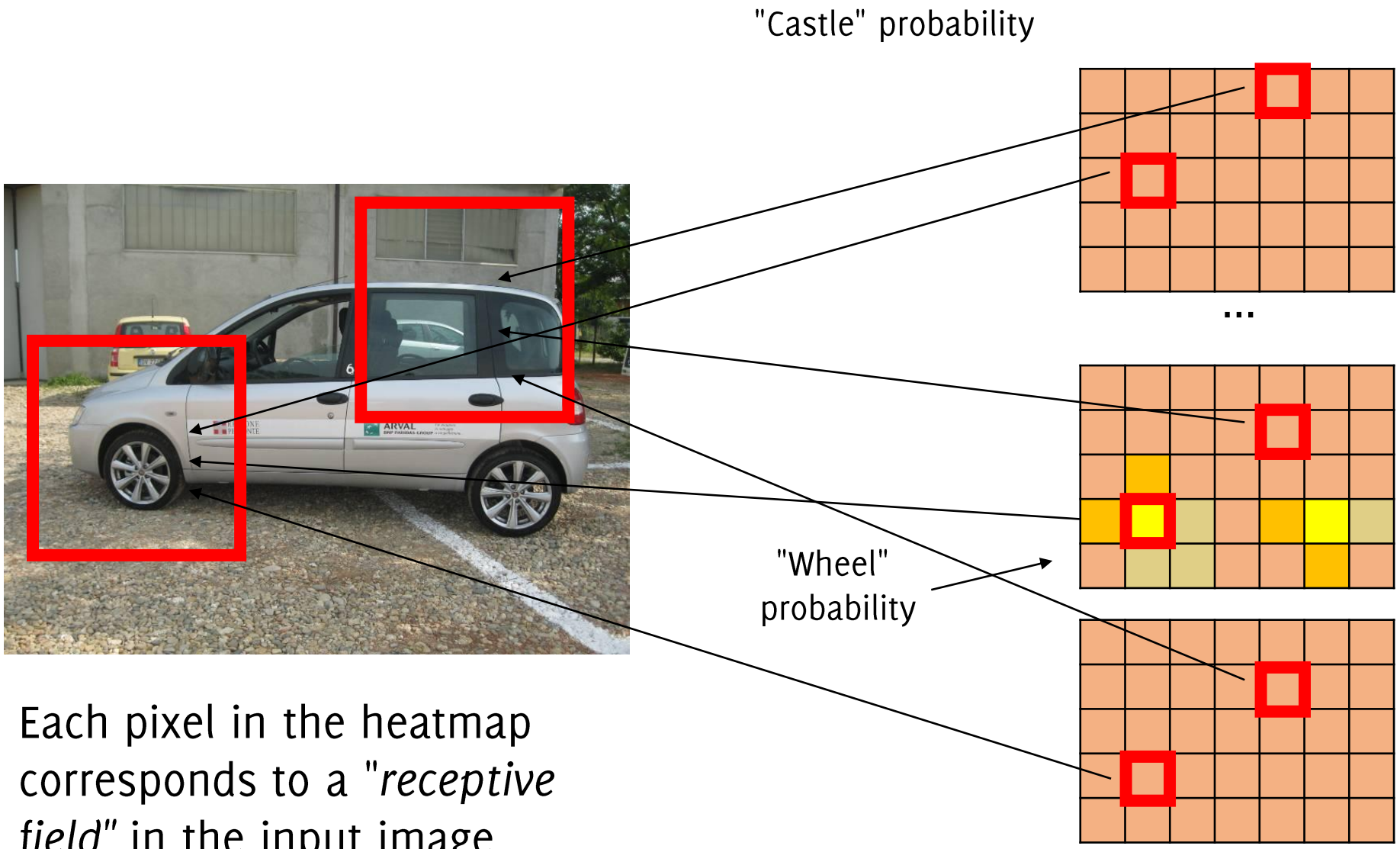
Long, J., Shelhamer, E., Darrell, T. "Fully convolutional networks for semantic segmentation". CVPR 2015

Output of a FCNN as heatmaps

A larger image than those used for training the network

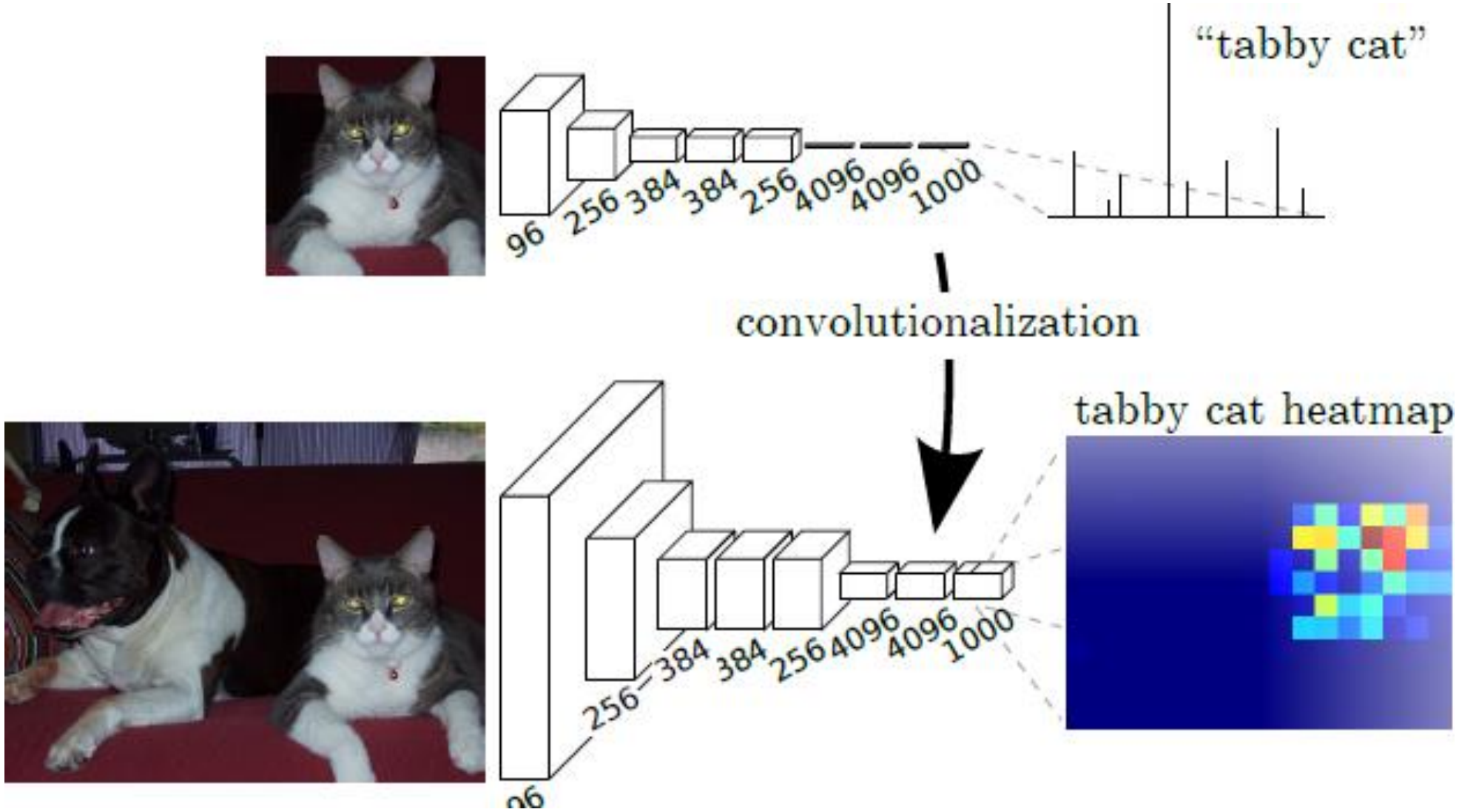


Output of a FCNN as heatmaps



Each pixel in the heatmap corresponds to a "*receptive field*" in the input image

Migration to FCNN of a pretrained model

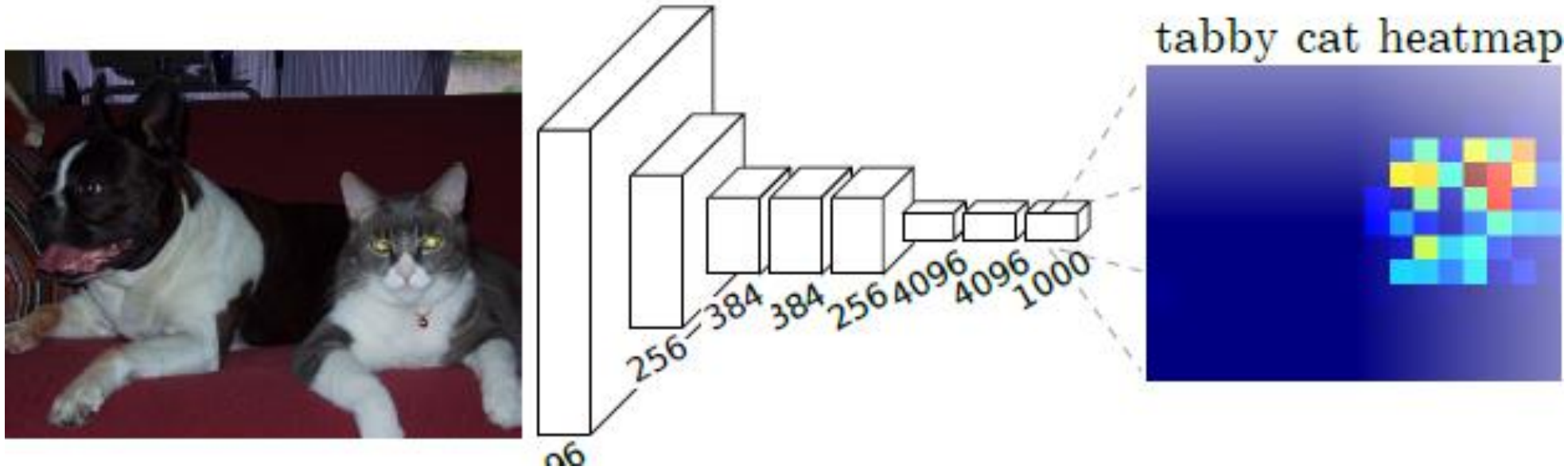


Long, Jonathan, *Evan* Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Migration to FCNN of a pretrained model

This stack of convlutions operates on the whole image as a filter.

Significantly more efficient than patch extraction and classification (avoids multiple repeated computations within overlapping patches)



Long, Jonathan, *Evan* Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Fully convolutional networks in keras

It is necessary to get and set the weights of networks by means of the methods **`get_weights`** and **`set_weights`**

- get the weights of the trained CNN

```
w7, b7 = model.layers[7].get_weights()
```

- reshape these weights to become a convolution

```
w7.reshape(20, 20, 10, 256)
```

- assign these weights to the FCNN architecture

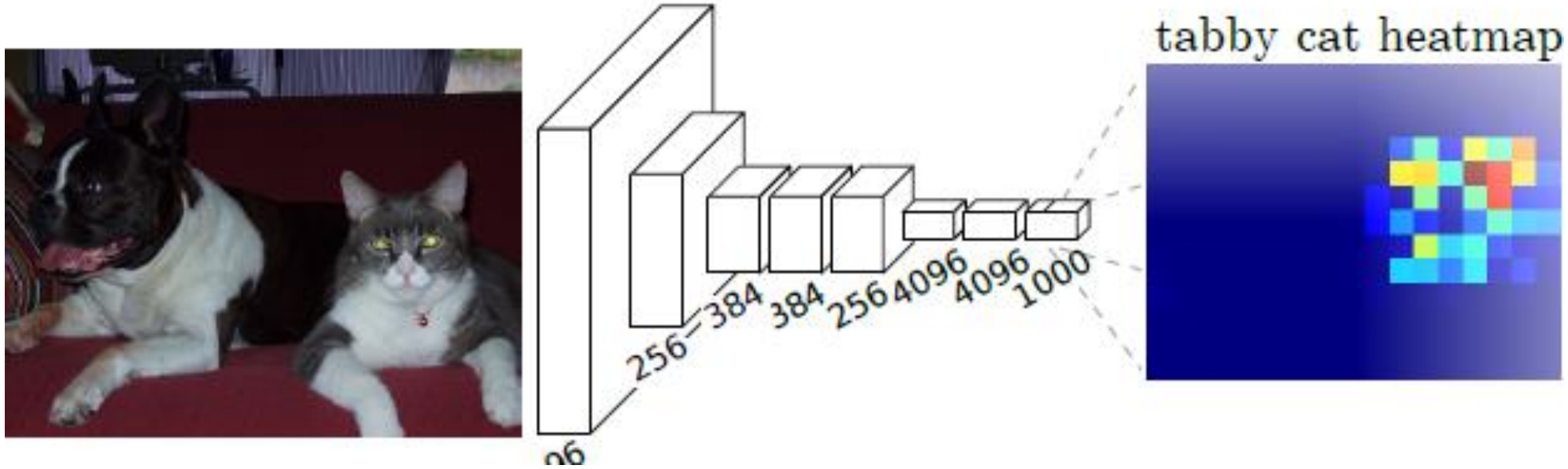
```
model2.layers[i].set_weights(w7, b7)
```

Semantic Segmentation using CNN

Predicting dense outputs for abritrary-sized inputs

Direct Heatmap Predictions

We can assign the predicted label in the heatmap to the whole receptive field, however that would be a very coarse estimate

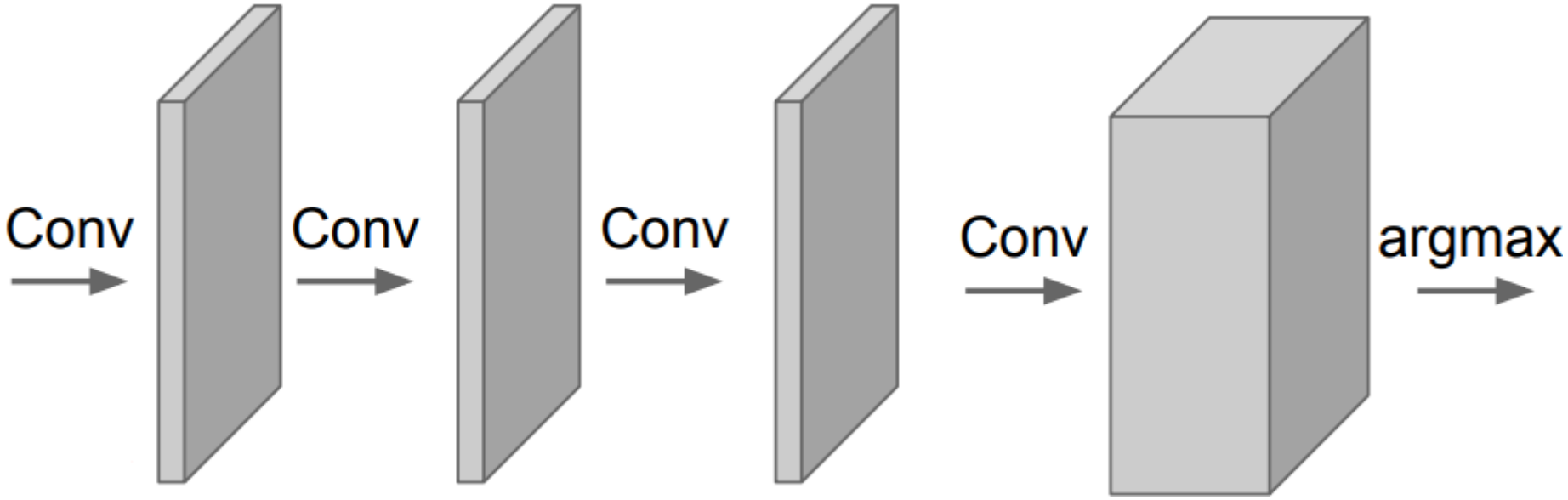


Long, Jonathan, *Evan* Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Only Convolutions

What if we avoid any pooling (just conv2d and activation layers)?

- Very small receptive field
- Very inefficient



CS231n: Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/

Drawbacks…

On the one hand we need to “go deep” to extract high level information on the image

On the other hand we want to stay local not to loose spatial resolution in the predictions

Semantic segmentation faces an inherent tension between semantics and location:

- ***global information resolves what**, while*
- ***local information resolves where** […]*

Combining fine layers and coarse layers lets the model make local predictions that respect global structure.

Long, Jonathan, *Evan* Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation.“ CVPR 2015

The Shift and Stich

Shift and Stich: Consider a ratio f between input size and the heatmap

- Compute heatmaps for all f^2 possible shifts of the input
- Map predictions to the image: each pixel in the heatmap provides prediction to the central pixel of the receptive field
- Interleave the heatmaps to form an image as large as the input

This allows exploiting the whole depth of the network

Efficient implementation through the à trous algorithm in wavelet

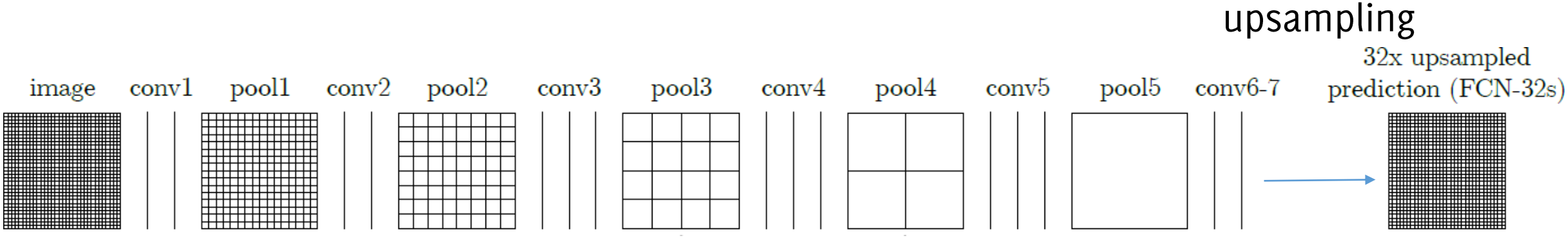
However, sort of rigid: upsampling is better

Long, Jonathan, *Evan* Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Prediction Upsampling

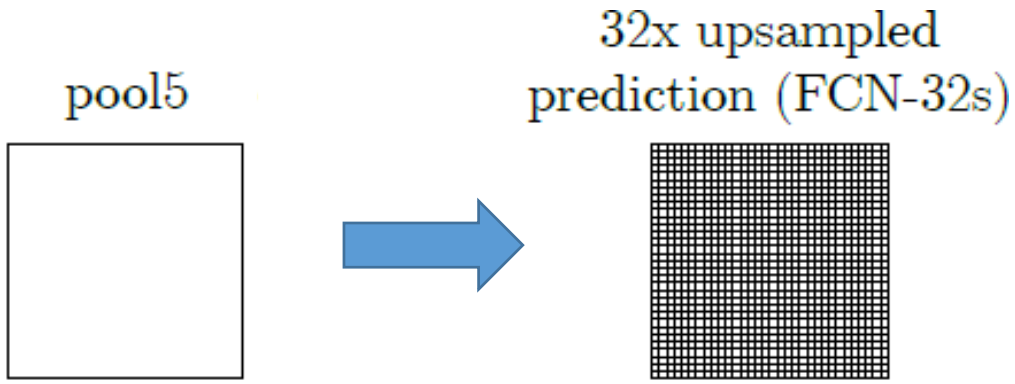
Linear upsampling can be implemented as a convolution filter (with a fractional input stride)

Upsampling filter can thus be learned during network training.



Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

How to perform upsampling?



Nearest Neighbor

1	2
3	4

→

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2 Output: 4 x 4

"Bed of Nails"

1	2
3	4

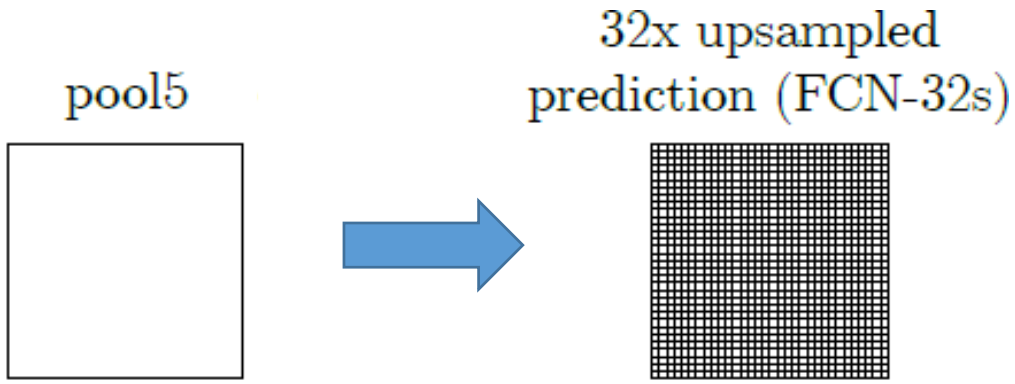
→

1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Input: 2 x 2 Output: 4 x 4

CS231n: Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/

How to perform upsampling?



You have to keep track of the maxpooling locations

Max Pooling
Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

5	6
7	8

Output: 2 x 2

Rest of the network

Max Unpooling
Use positions from
pooling layer

1	2
3	4

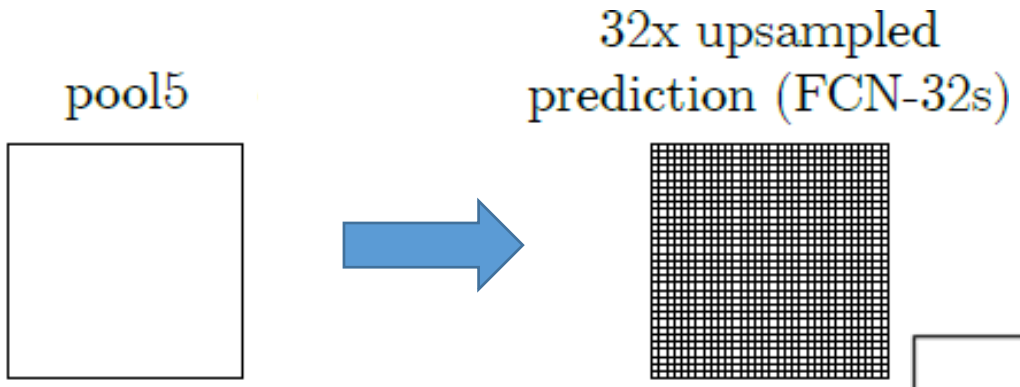
Input: 2 x 2

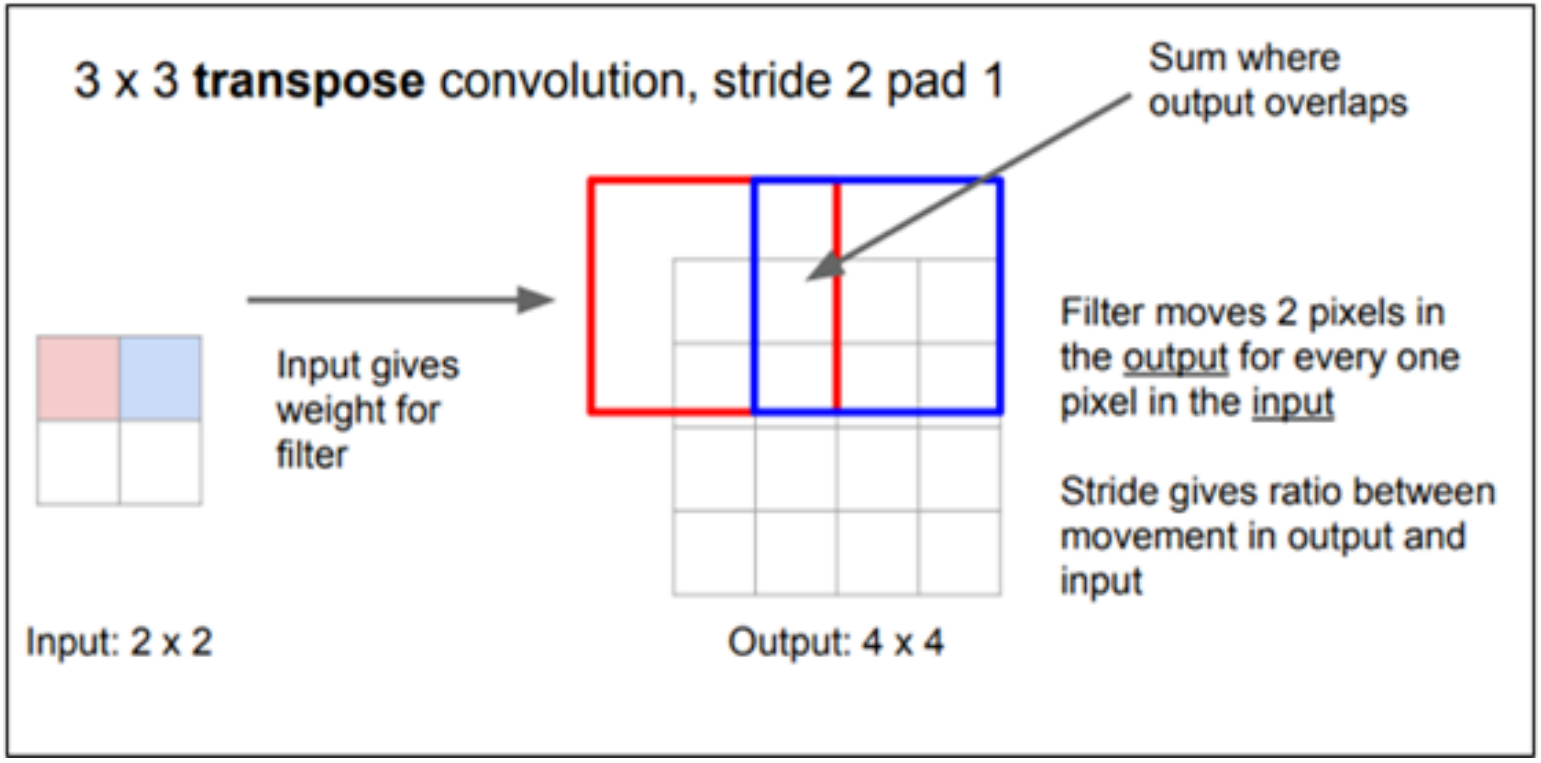
0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Output: 4 x 4

CS231n: Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/

How to perform upsampling?





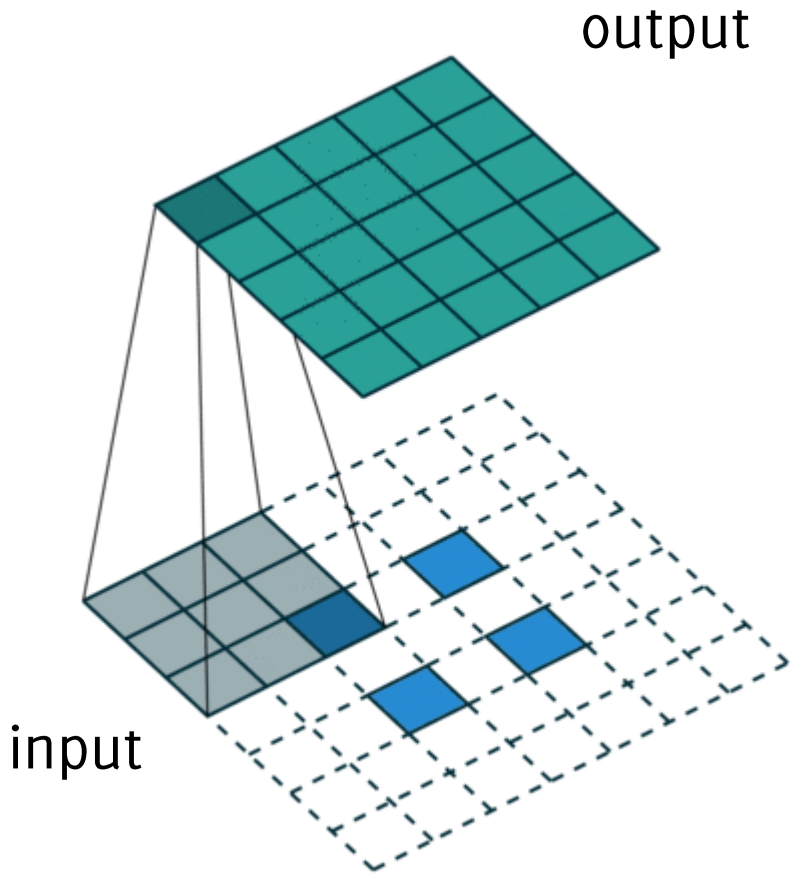
CS231n: Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/

How to perform upsampling?

Transpose Convolution can be seen as a traditional convolution after having upsampled the input image

Many names for transpose convolution: fractional strided convolution, backward strided convolution, deconvolution (very misleading!!!)

Upsamping based on convolution gives more degrees of freedom **as the filters can be learned**!



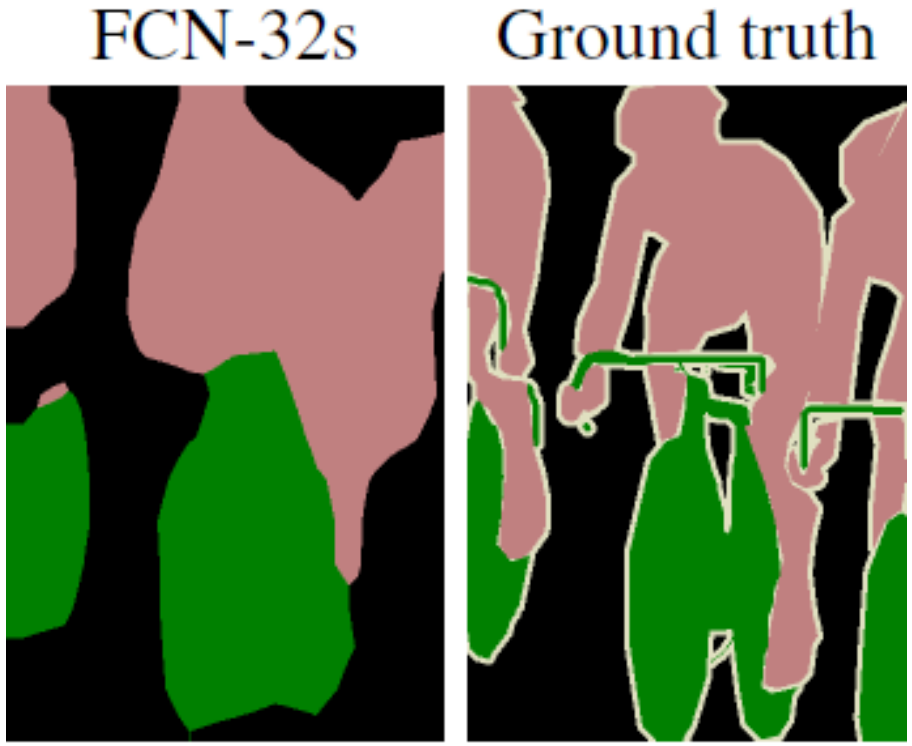
https://github.com/vdumoulin/conv_arithmetic

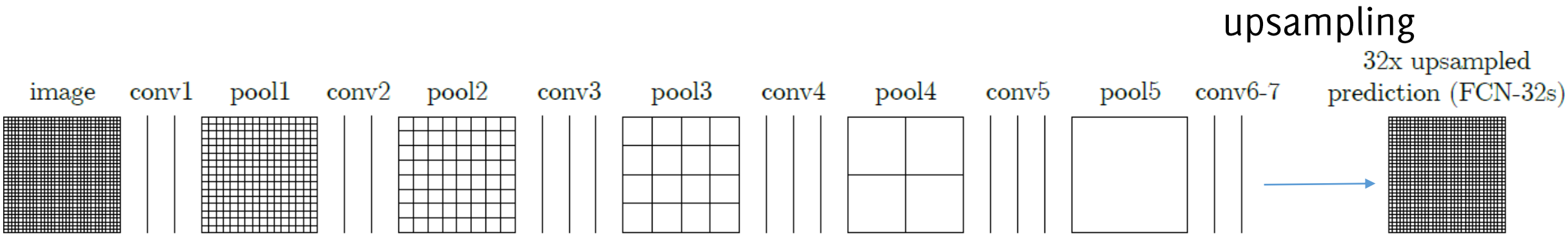
Dumoulin, Vincent, and Francesco Visin. "A guide to convolution arithmetic for deep learning." *arXiv preprint arXiv:1603.07285* (2016).

Prediction Upsampling

These predictions however are very coarse

Upsampling filters are learned with initialization equal to the bilinear interpolation

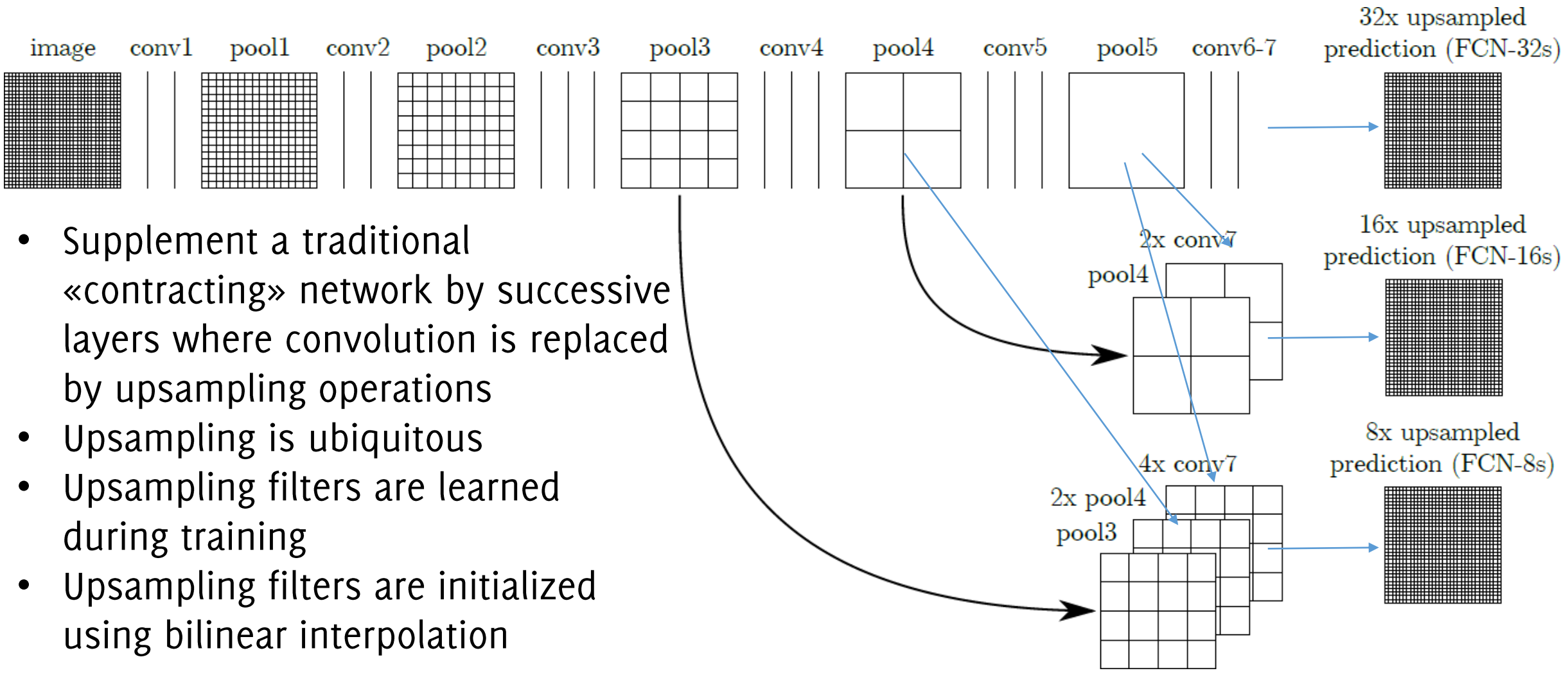




Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Solution: Skip Connections!

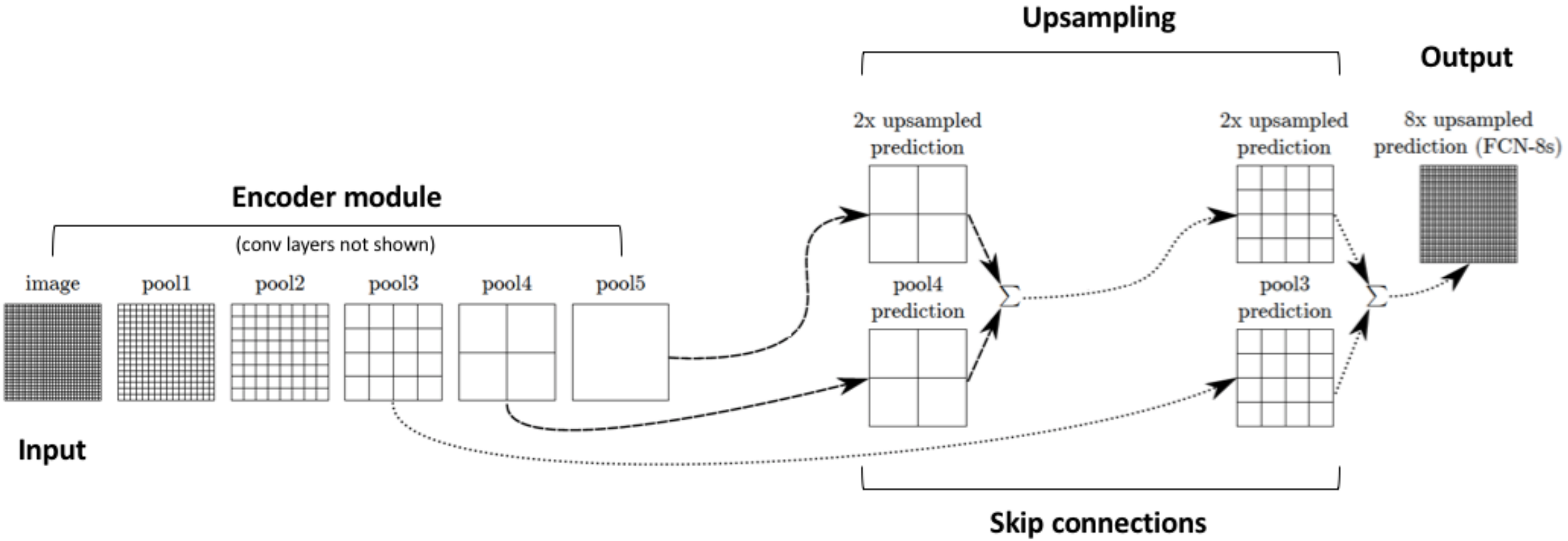
upsampling



- Supplement a traditional «contracting» network by successive layers where convolution is replaced by upsampling operations
- Upsampling is ubiquitous
- Upsampling filters are learned during training
- Upsampling filters are initialized using bilinear interpolation

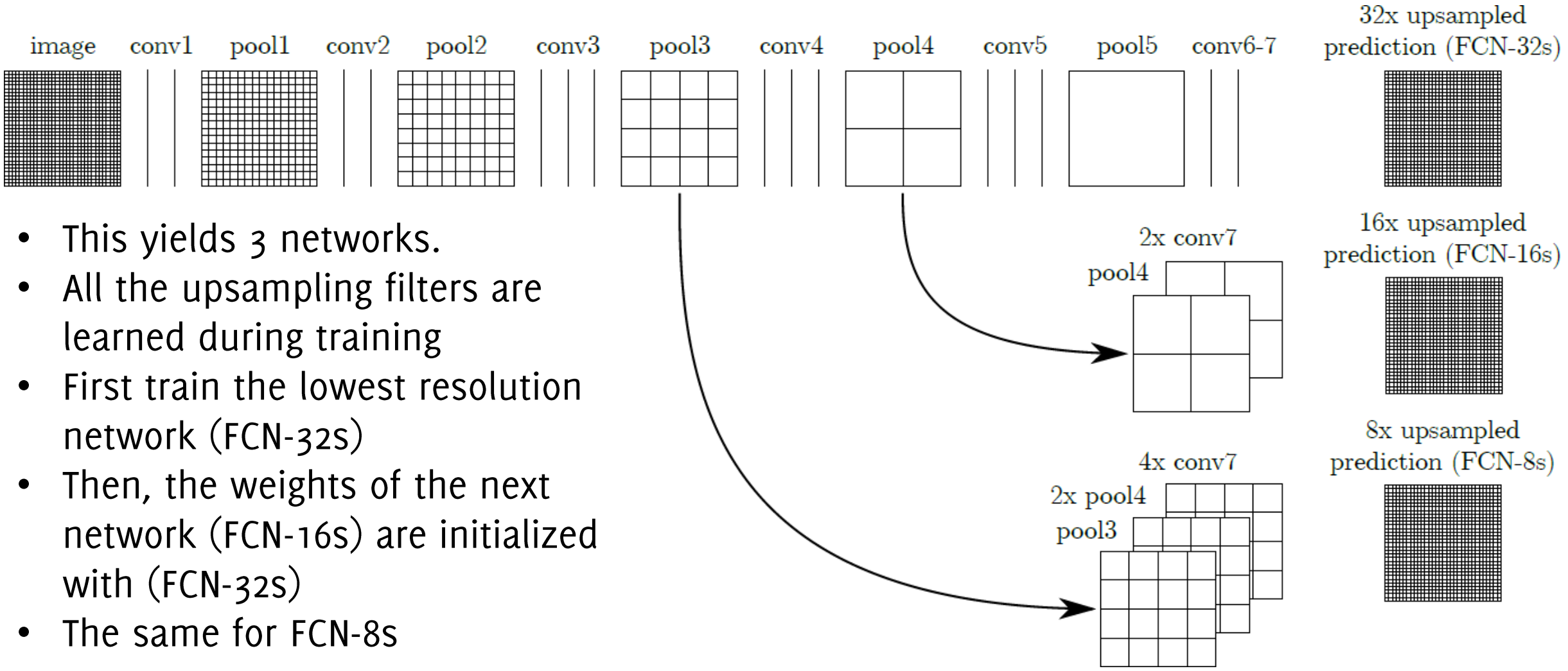
Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Solution: Skip Connections!



https://www.jeremyjordan.me/semantic-segmentation/

Solution: Skip Connections!

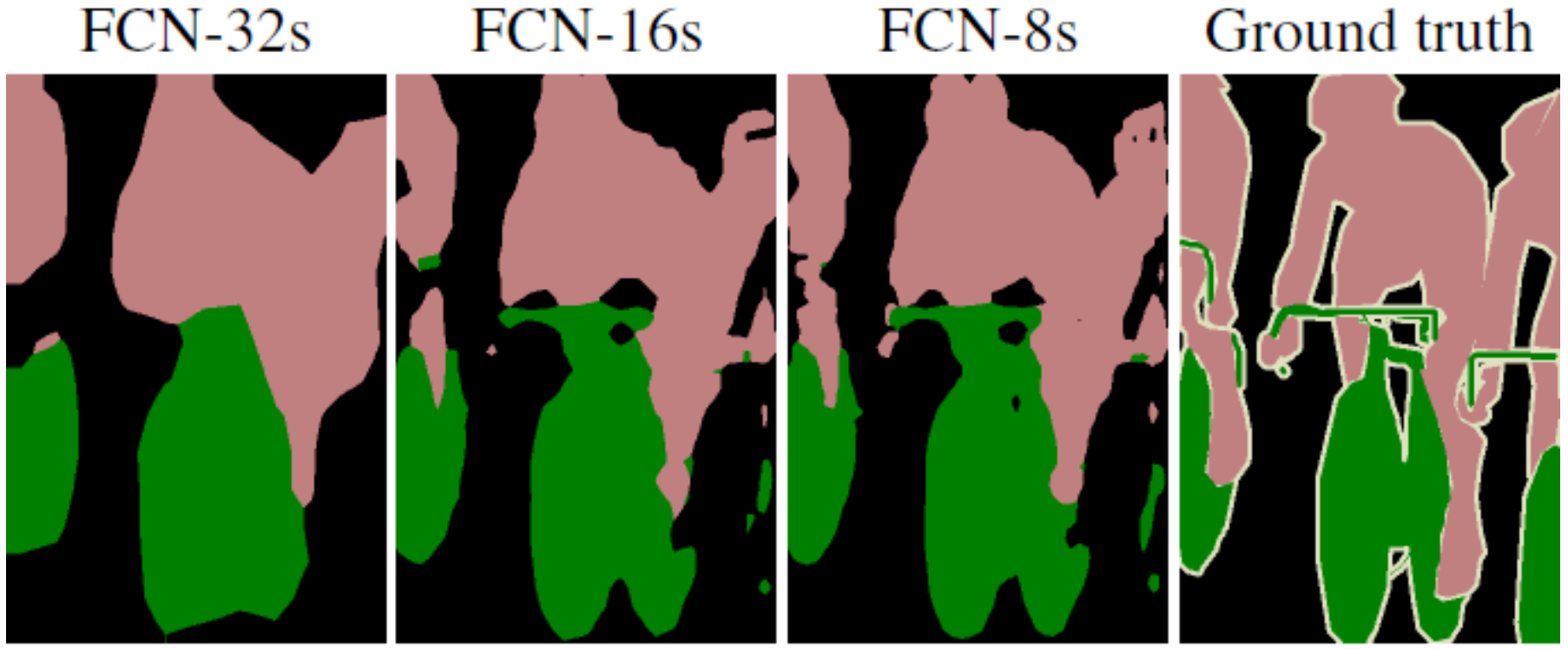


- This yields 3 networks.
- All the upsampling filters are learned during training
- First train the lowest resolution network (FCN-32s)
- Then, the weights of the next network (FCN-16s) are initialized with (FCN-32s)
- The same for FCN-8s

Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Semantic Segmentation

Retaining intermediate information is beneficial, the deeper layers contribute to provide a better refined estimate of segments



Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

FCNN Training

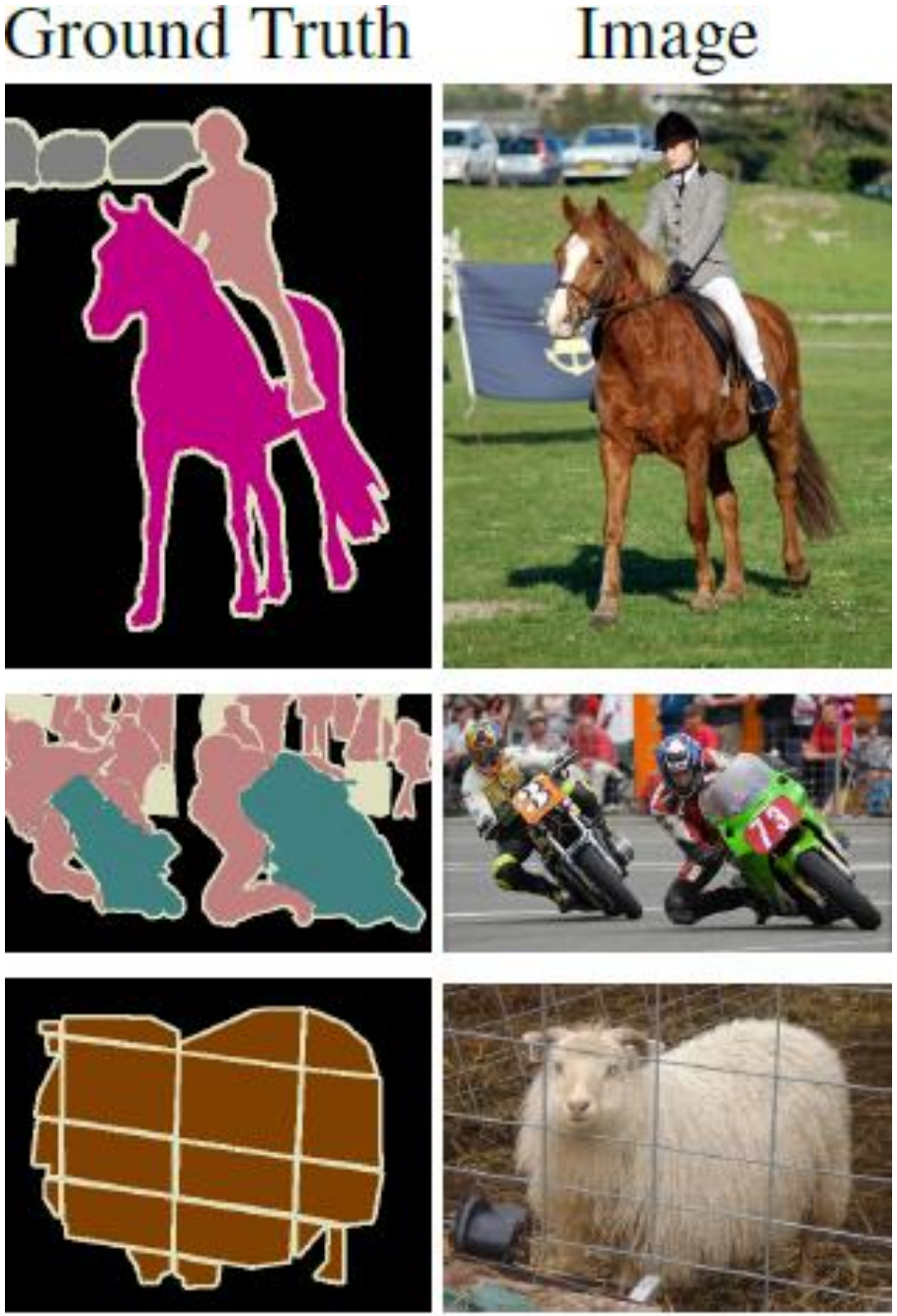
A pixel-wise ground truth (annotated images) can be used for training a classification or segmentation model in an **end-to-end manner**

Full-Image Training:

- Provides dense output by re-projecting heatmap predictions to their receptive fields.
- The global loss of an input image is averaged over the spatial dimension of the output layer

$$\ell(\boldsymbol{x}, \theta) = \sum_{\boldsymbol{x}_{i,j}} \ell'(\boldsymbol{x}_{i,j}, \theta)$$

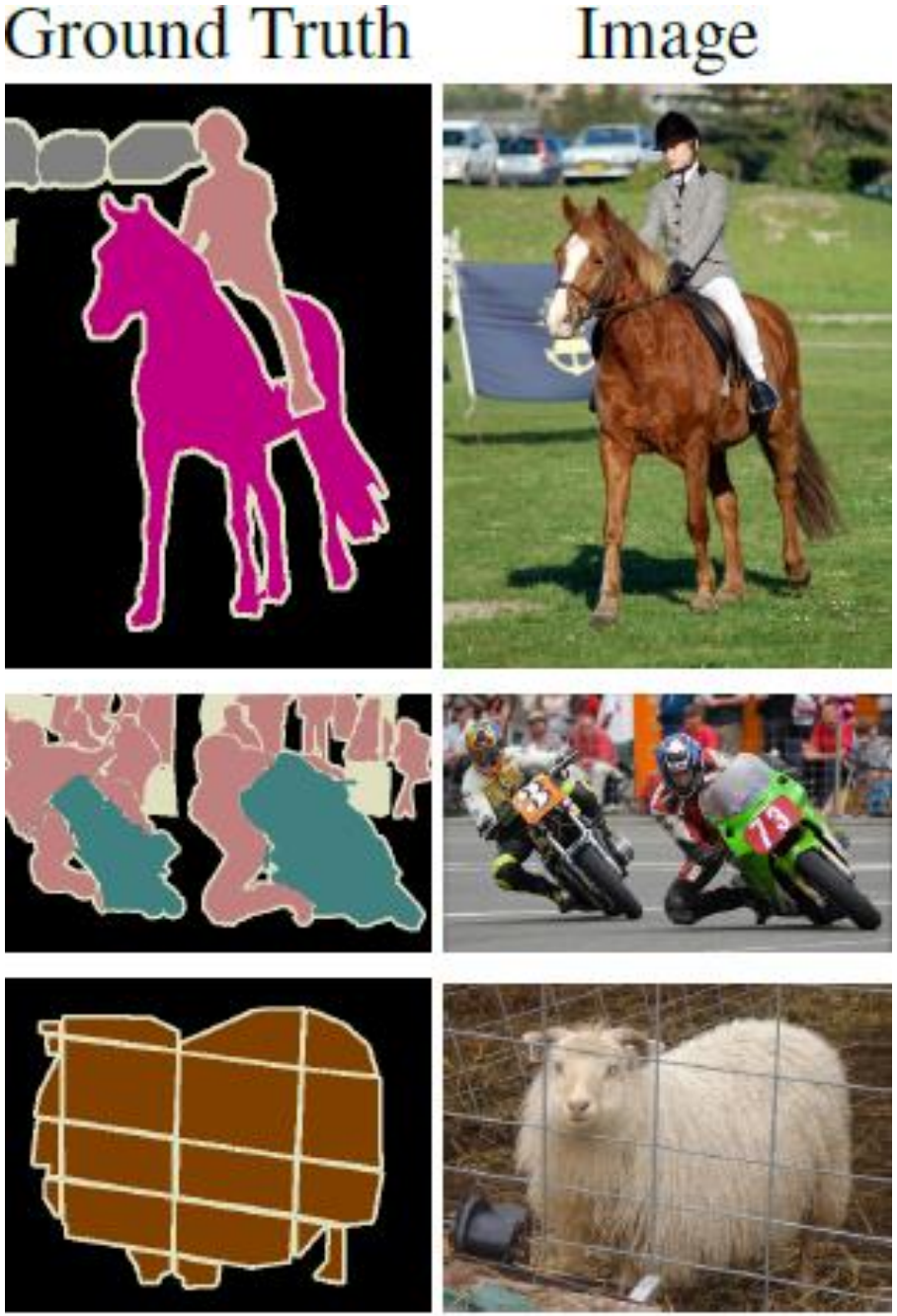
- The true label of each pixel in the heatmap is the label of the central pixel



Long, Jonathan, *Evan* Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

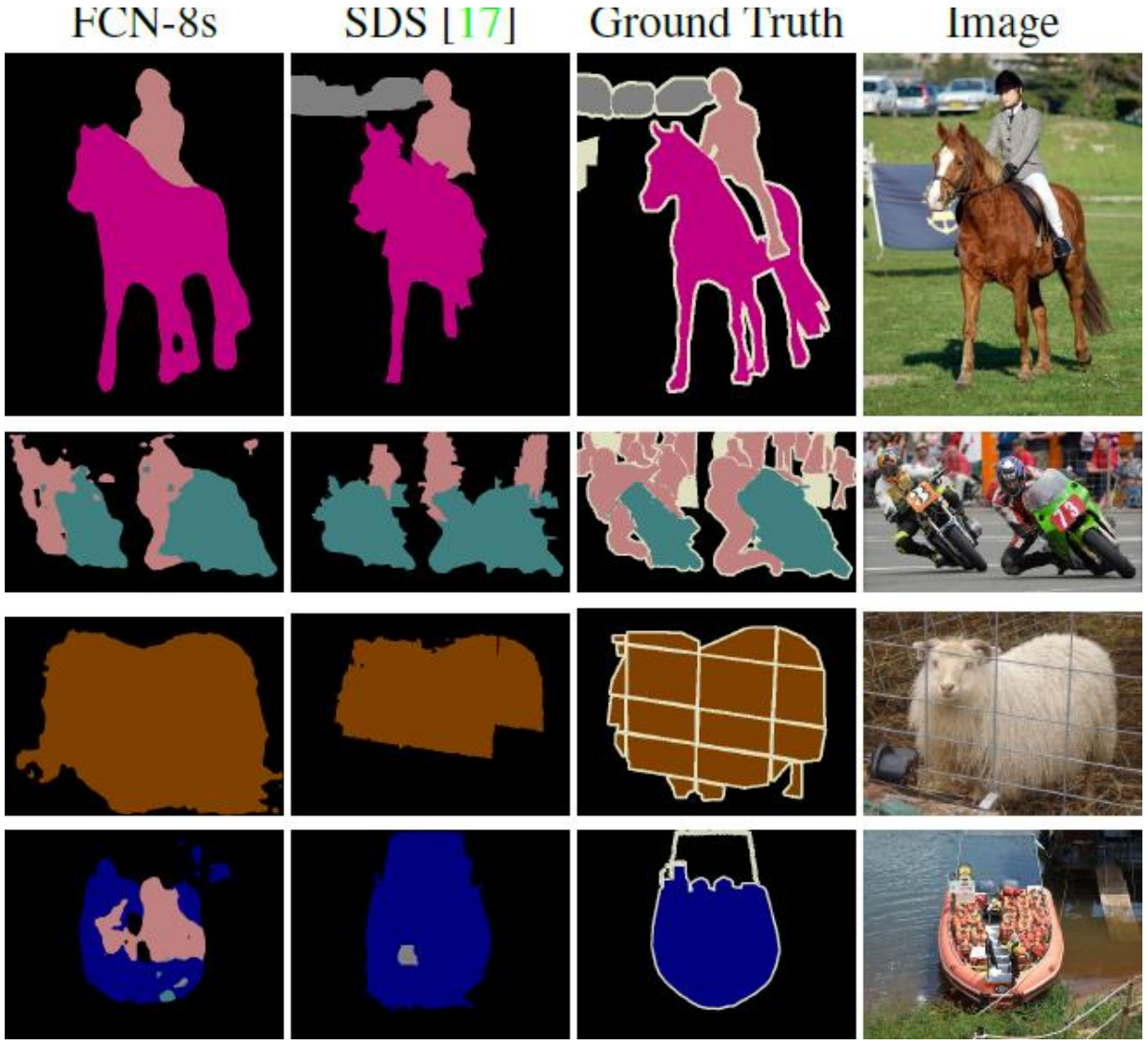
Full-Image Training

- «Equivalent» to patch-wise training
- Take advantage of FCNN efficiency, no re-computation of features in overlapping regions
- Batches in patch-wise training are randomly assembled
  - Some form of loss function masking needed here
- Patch resampling for solving class imbalance not possible
  - Adopt loss re-weighting



Long, Jonathan, *Evan* Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

Semantic Segmentation



Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." CVPR 2015

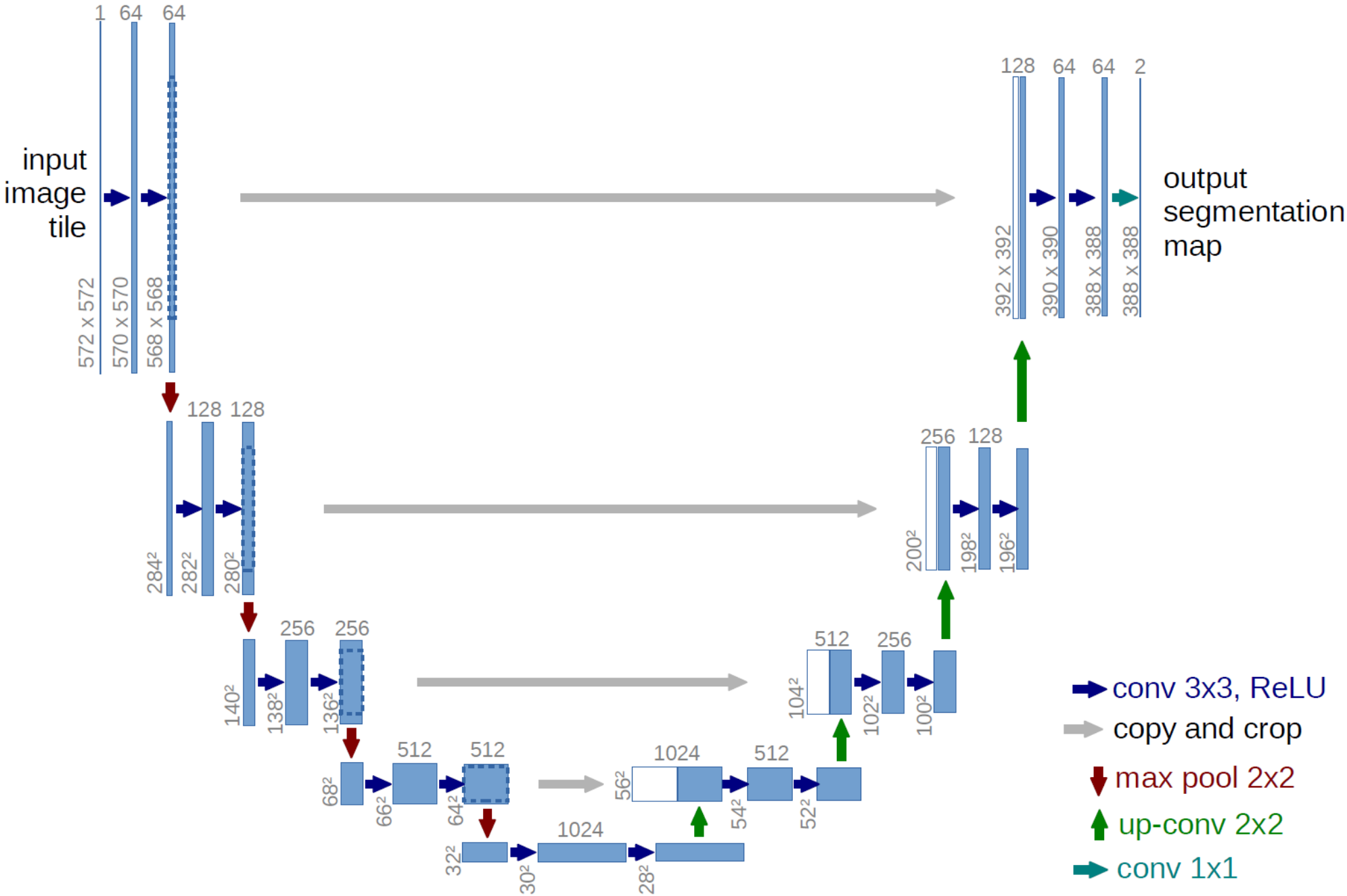
U-net

Network formed by:

- A contracting path
- An expansive path

No fully connected layers

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*, 2015.



Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*, 2015.

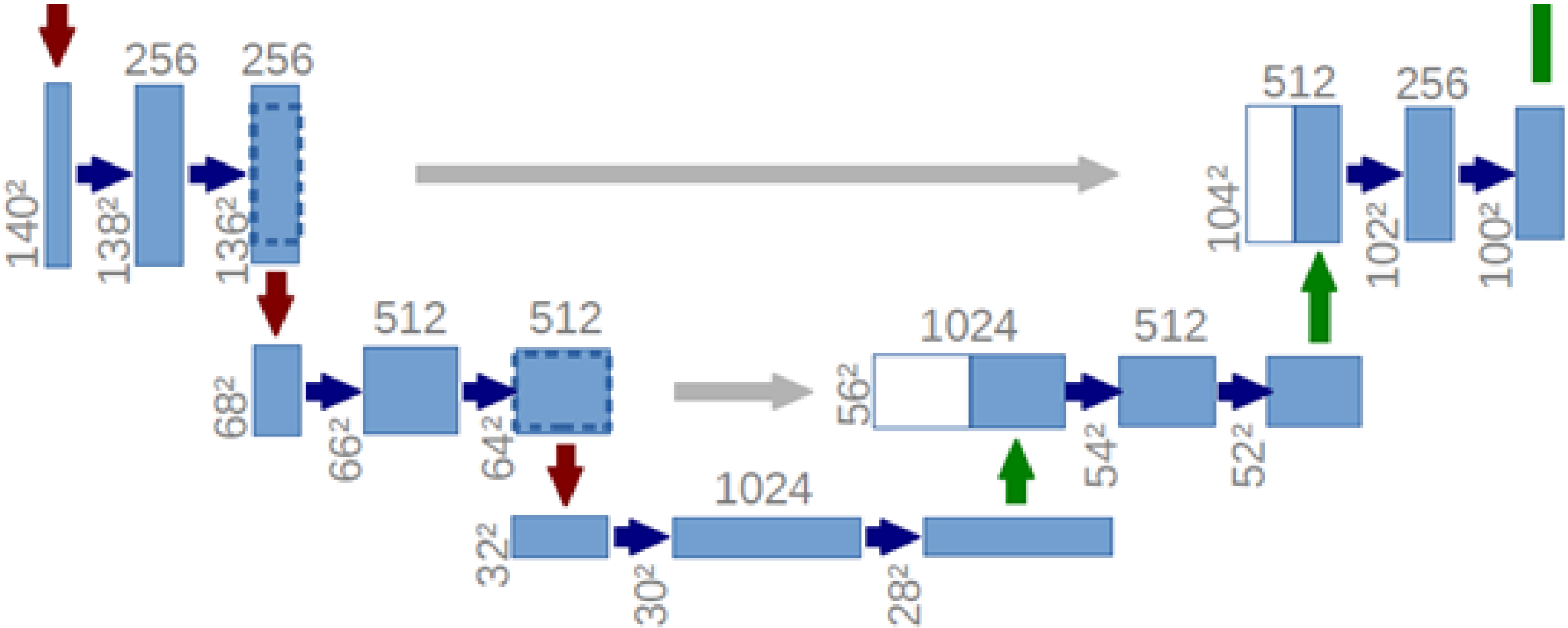
U-net: contracting path

Repeats blocks of:

- 3×3 convolution + ReLU
- 3×3 convolution + ReLU
- Maxpooling 2×2

At each downsampling the number of feature maps is doubled

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*, 2015.



Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*, 2015.

U-net: Expanding path

repeats blocks of:

- 2×2 transpose convolution, **halving the number** of feature maps
- Concatenation of corresponding cropped features in the contracting
- 3×3 convolution + ReLU
- 3×3 convolution + ReLU
- Maxpooling 2×2

No fully connected layers: at the end there are $1 \times 1 \times N$ convolutions to yield predictions out of the convolutional feature maps

Output image is smaller than the input image by a constant border

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*, 2015.

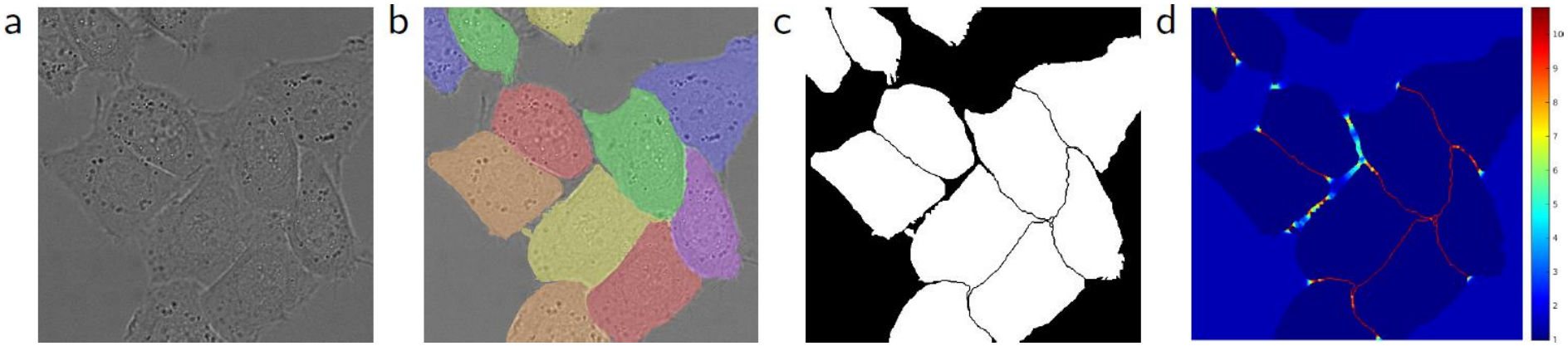
U-net: Training

Full-image training, by using a weigthed loss function

$$w(\boldsymbol{x}) = w_c(\boldsymbol{x}) + w_0\, e^{-\frac{\left(d_1(\boldsymbol{x})+d_2(\boldsymbol{x})\right)^2}{2\sigma^2}}$$

To balance class proportions (w_c) and enhance segmentation at borders:

- d_1 is the distance to the border of the closest cell
- d_2 is the distance to the border of the second closest cell



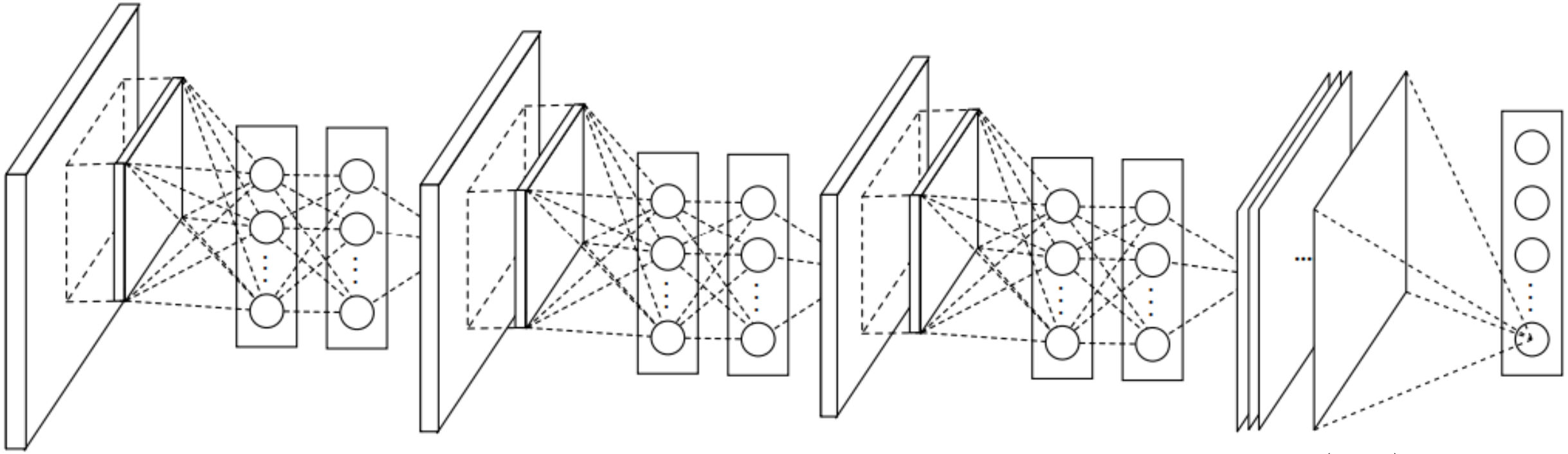
Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*, 2015.

Global Averaging Pooling

Networks in Network

A different architecture composed of many layers of a MLP having limited spatial extent

No pooling and no fully connected at the very end



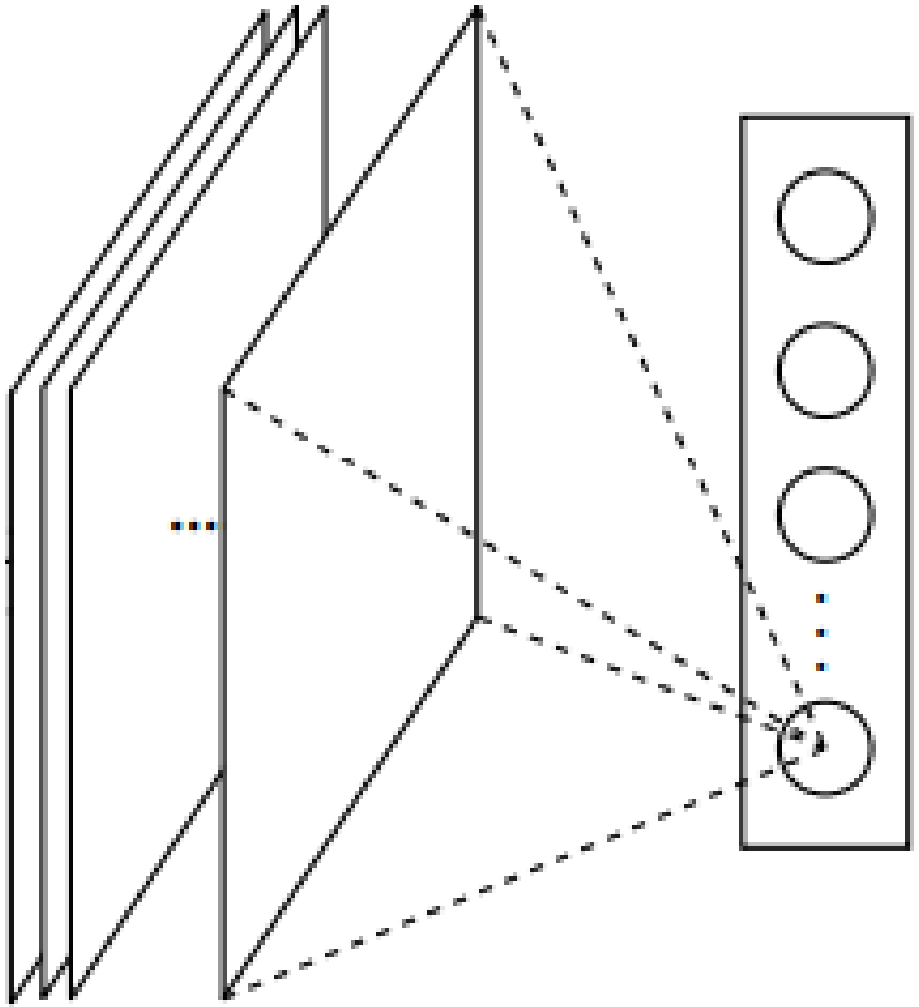
Lin, Min, Qiang Chen, and Shuicheng Yan. "Network in network." *arXiv preprint arXiv:1312.4400v3* (2014).

The Global Averaging Pooling (GAP) Layer

GAP squeezes each layer in the convolutional feature volume to an equivalent vector that is as long as the volume depth

Then, just a soft-max over these averages provides accurate classification

Rmk: the number of feature maps has to be equal to the number of output classes!



Lin, Min, Qiang Chen, and Shuicheng Yan. "Network in network." *arXiv preprint arXiv:1312.4400v3* (2014).

Rationale behind GAP

Fully connected layers are prone to overfitting

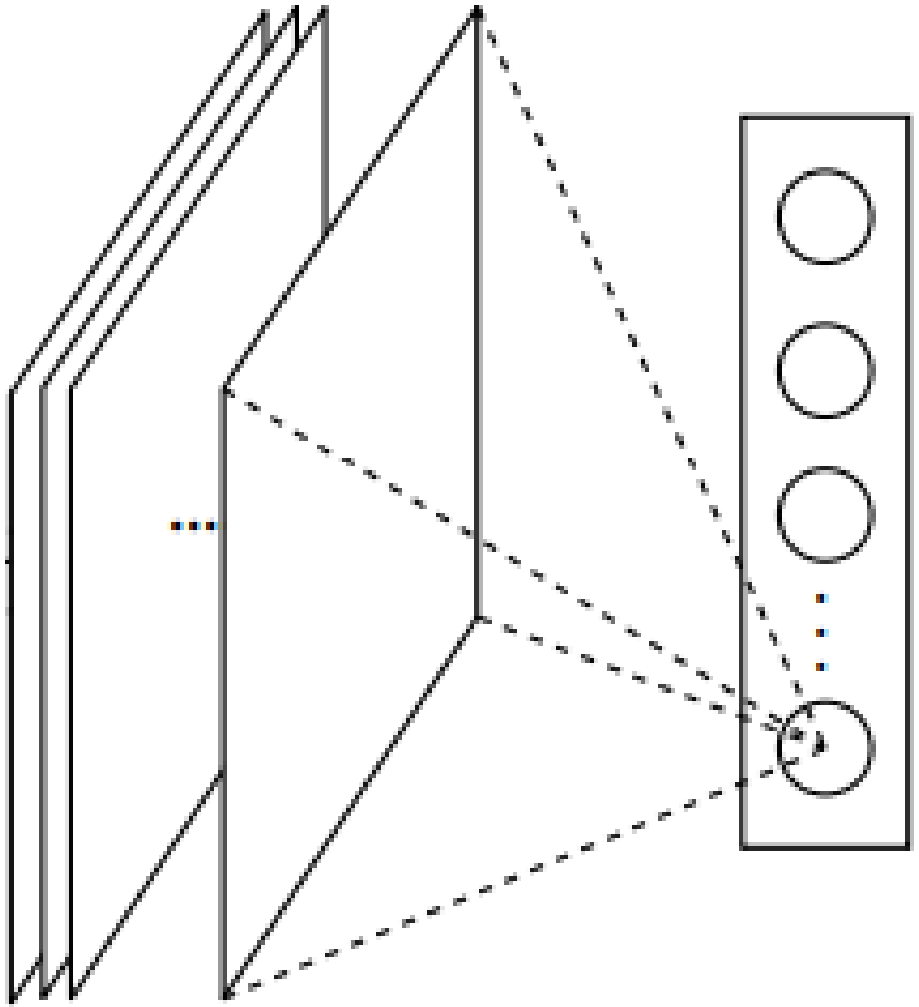
- They have many parameters
- Dropout was proposed as a regularized that randomly sets to zero a percentage of activations in the FC nayers during training

The GAP strategy is:

- Remove the fully connected layer at the end of the network!

The Global Averaging Pooling (GAP) Layer

- Since just a soft-max is employed, using **GAP introduces a correspondence** between **feature maps** and object **categories** (feature maps as confidence score on classes).
- This makes GAP a structural regularizer
- No parameters, no risk of overfitting
- Makes the network independent of the input size
- Robust to spatial variations in the input (e.g. shifts)

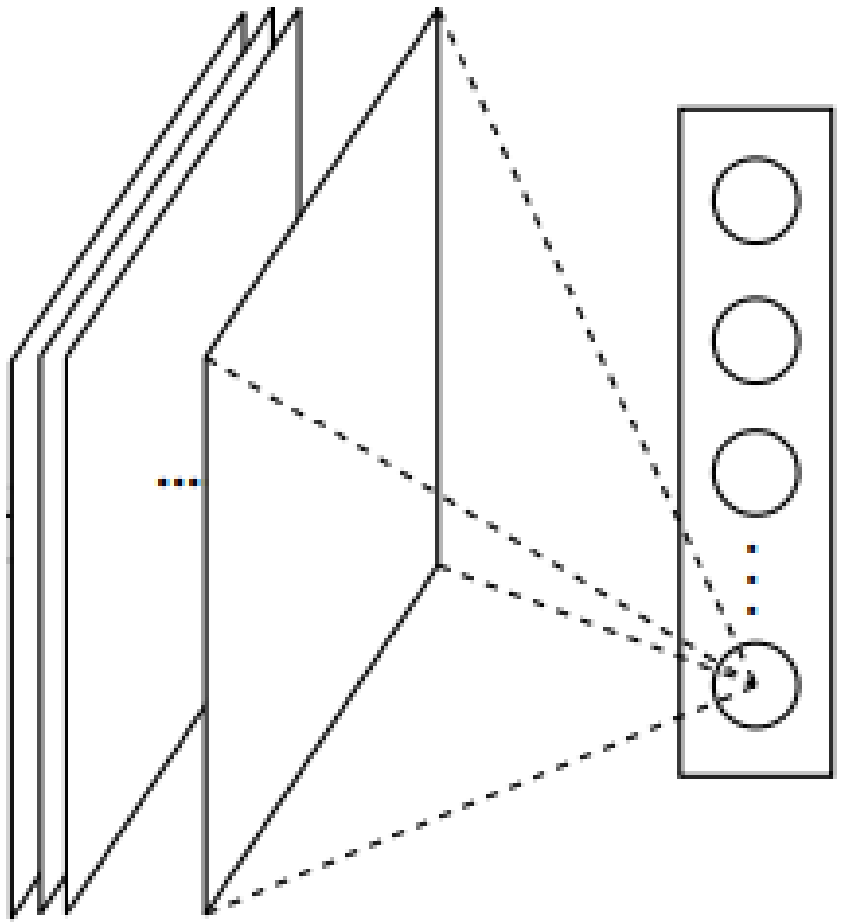


Lin, Min, Qiang Chen, and Shuicheng Yan. "Network in network." *arXiv preprint arXiv:1312.4400v3* (2014).

The Global Averaging Pooling (GAP) Layer

We indeed see that GAP is acting as a (structural) regularizer

Method	Testing Error
mlpconv + Fully Connected	11.59%
mlpconv + Fully Connected + Dropout	10.88%
mlpconv + Global Average Pooling	10.41%



Lin, Min, Qiang Chen, and Shuicheng Yan. "Network in network." *arXiv preprint arXiv:1312.4400v3* (2014).