DTU Course 02456 Deep learning
2 Convolutional neural networks

Ole Winther

Dept for Applied Mathematics and Computer Science
Technical University of Denmark (DTU)

August 20, 2016

Objectives of lectures week 2

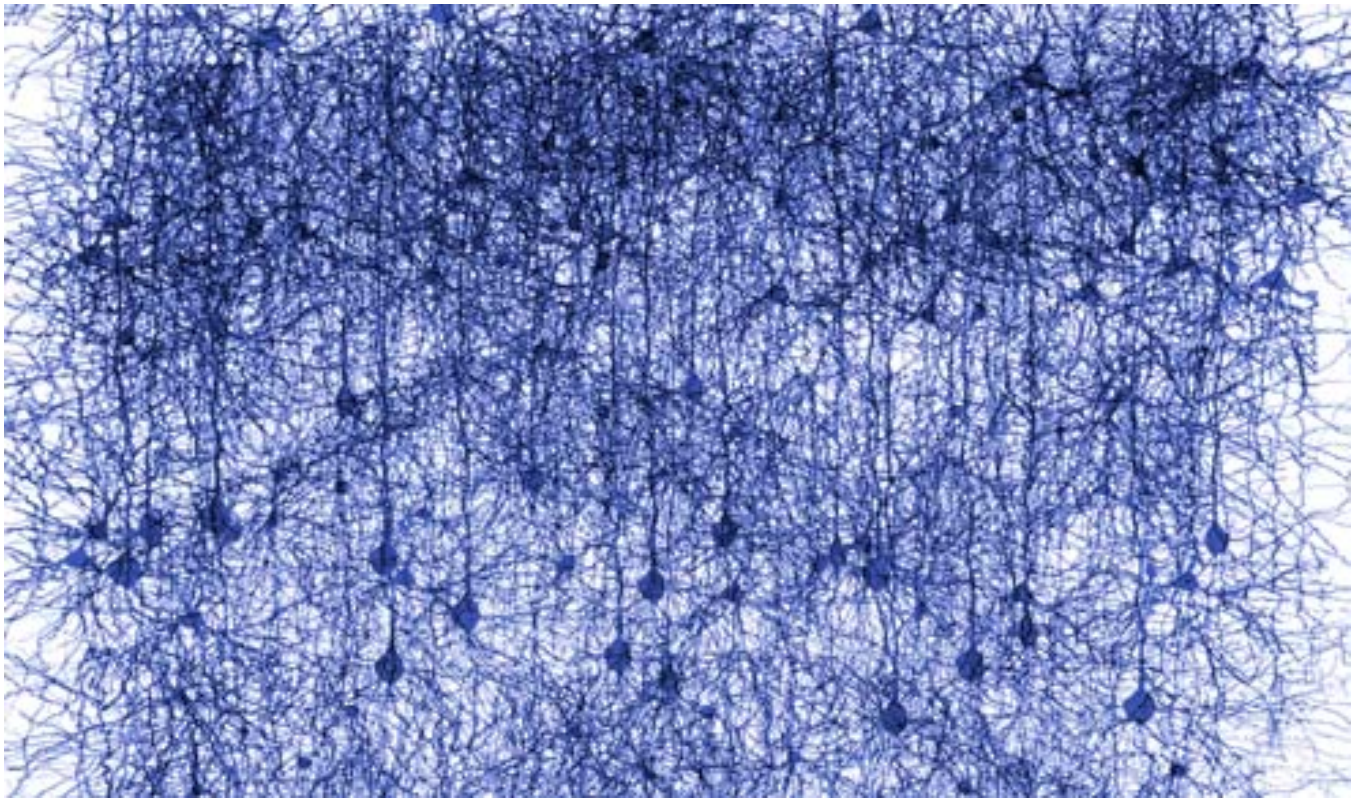
- P1: Convolutional neural networks and
- what they can be used for.
- P2: Details about convolutional filters and pooling

Part 1:

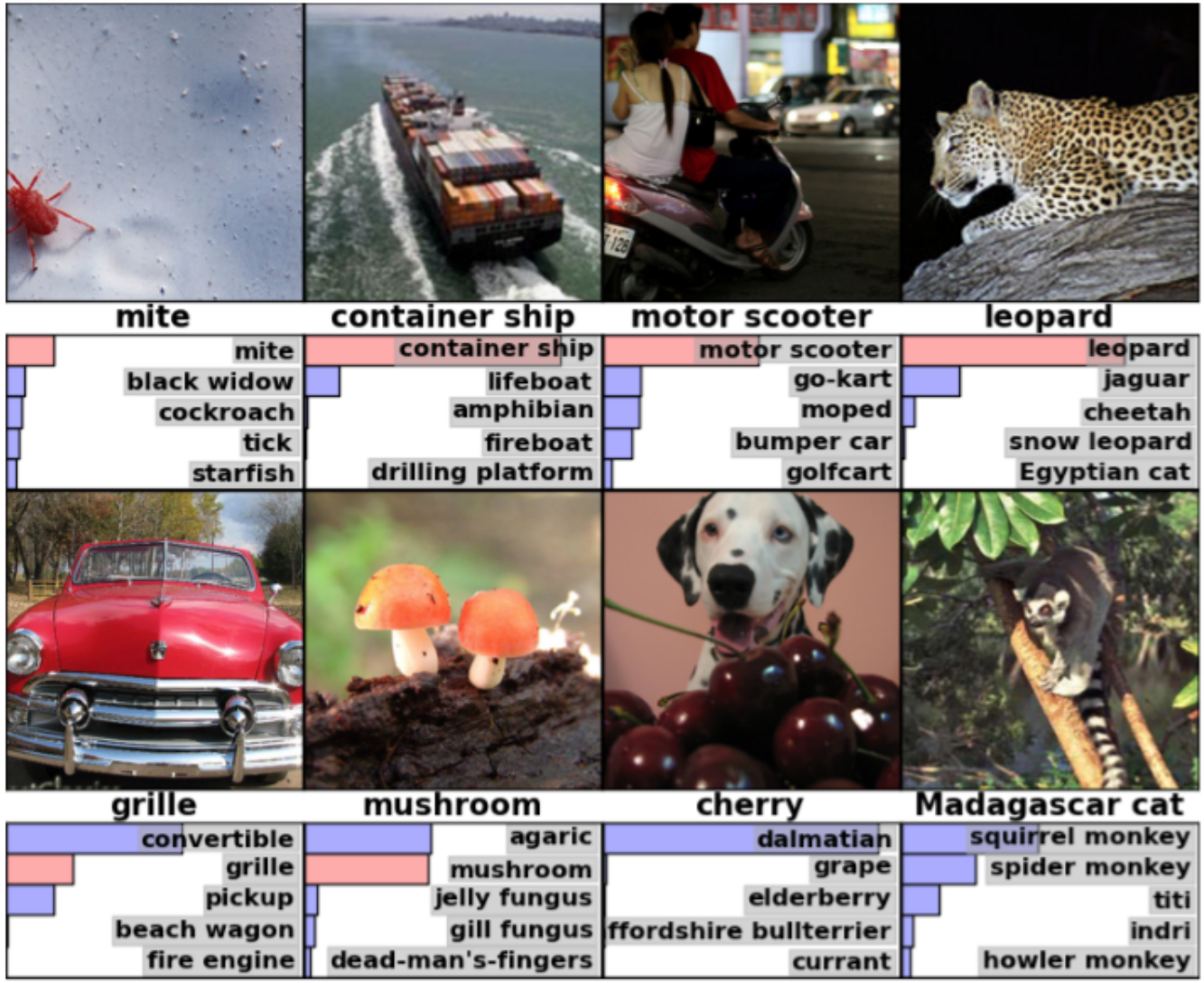
Convolutional NNs
Definition and uses

Neural networks (NNs)

- Feedforward neural networks (FFNNs)
- Convolutional neural networks (CNNs)
- Recurrent Neural Networks (RNNs)
- Auto-encoders (AE)

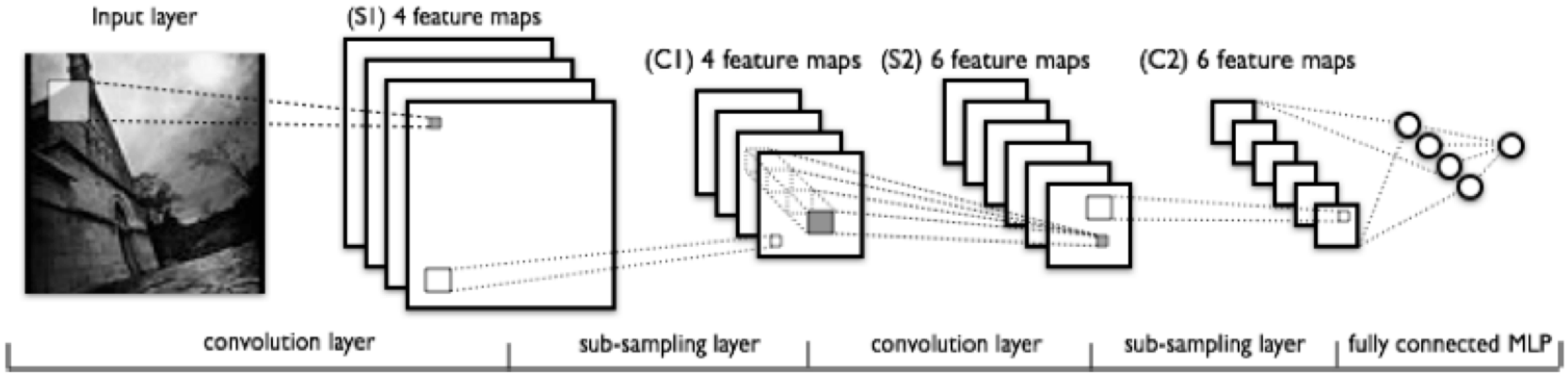


ImageNet - image classification



- 1.000 different classes - including many types of dogs!
- 1.000.000 training images

Convolutional neural networks



$$\begin{bmatrix} 10 & 0 & -10 \\ 0 & 0 & 0 \\ -10 & 0 & 10 \end{bmatrix}$$

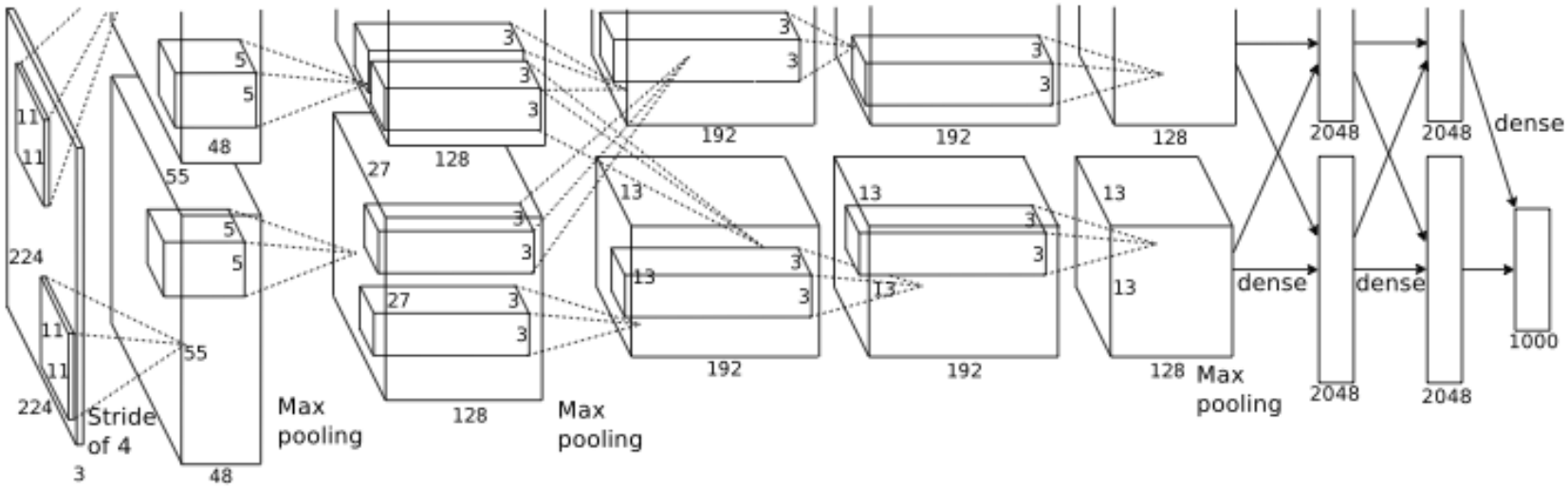
Feature engineering vs engineered models

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

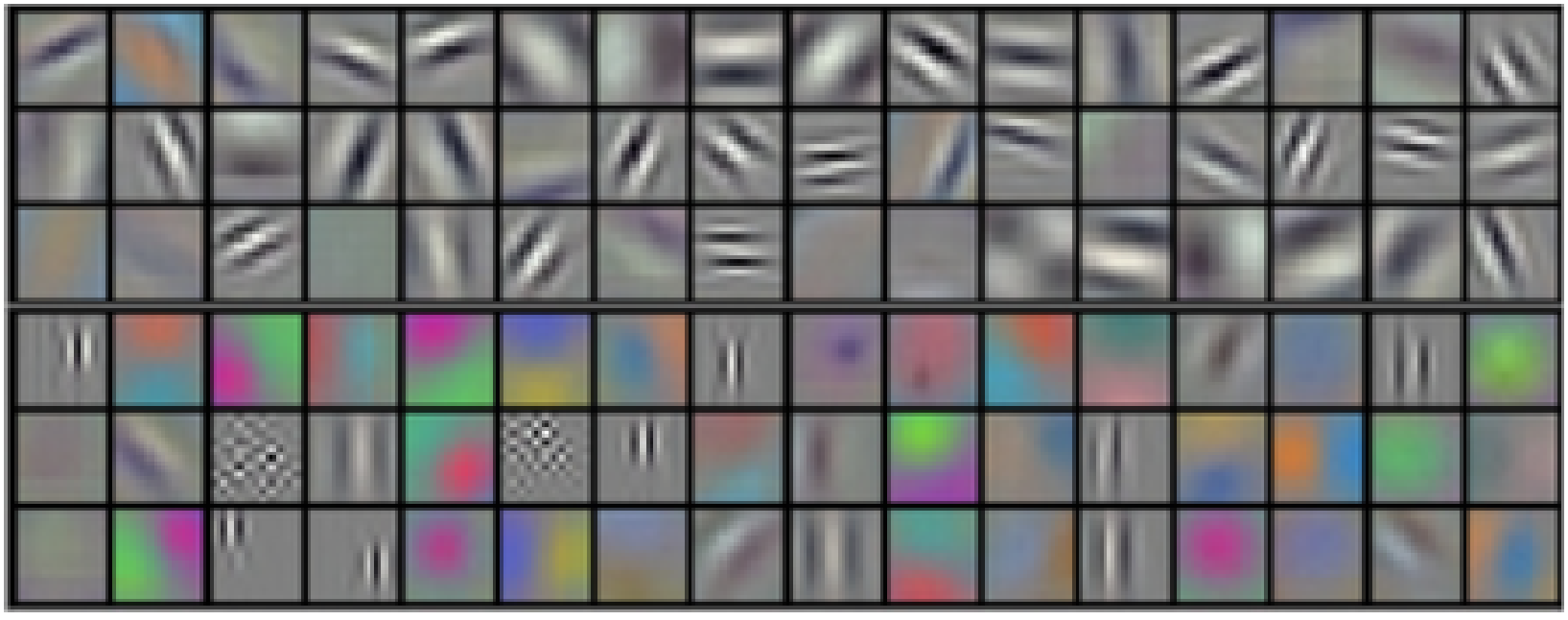
Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca



www.cs.toronto.edu/~fritz/absps/imagenet.pdf

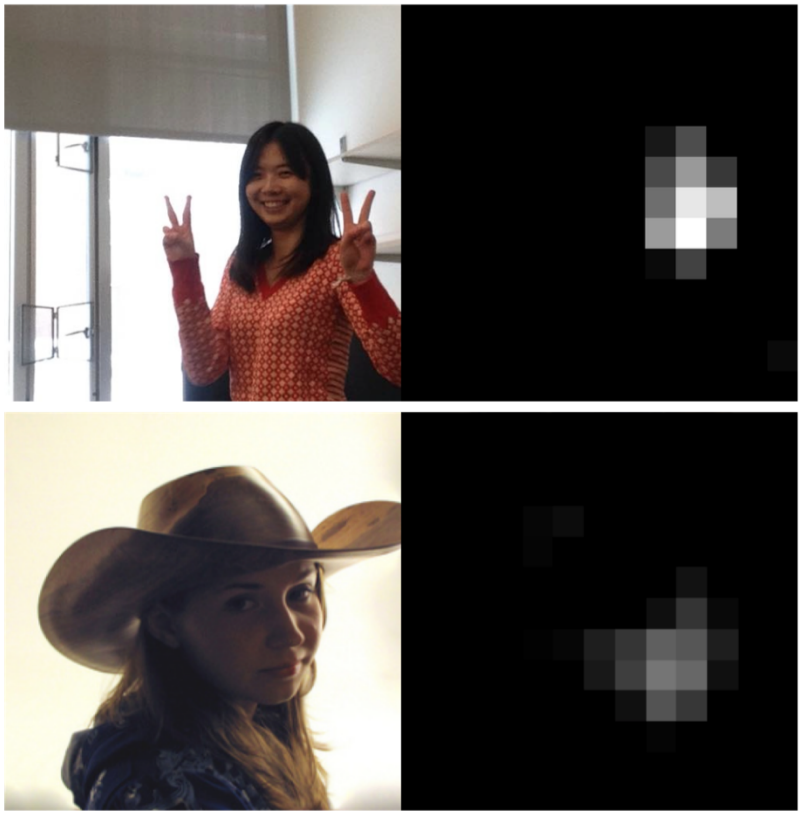
Learned filters in first layer



- Seminal paper:
- Krizhevsky et al, ImageNet Classification with Deep Convolutional Neural Networks
- `www.cs.toronto.edu/~fritz/absps/imagenet.pdf`

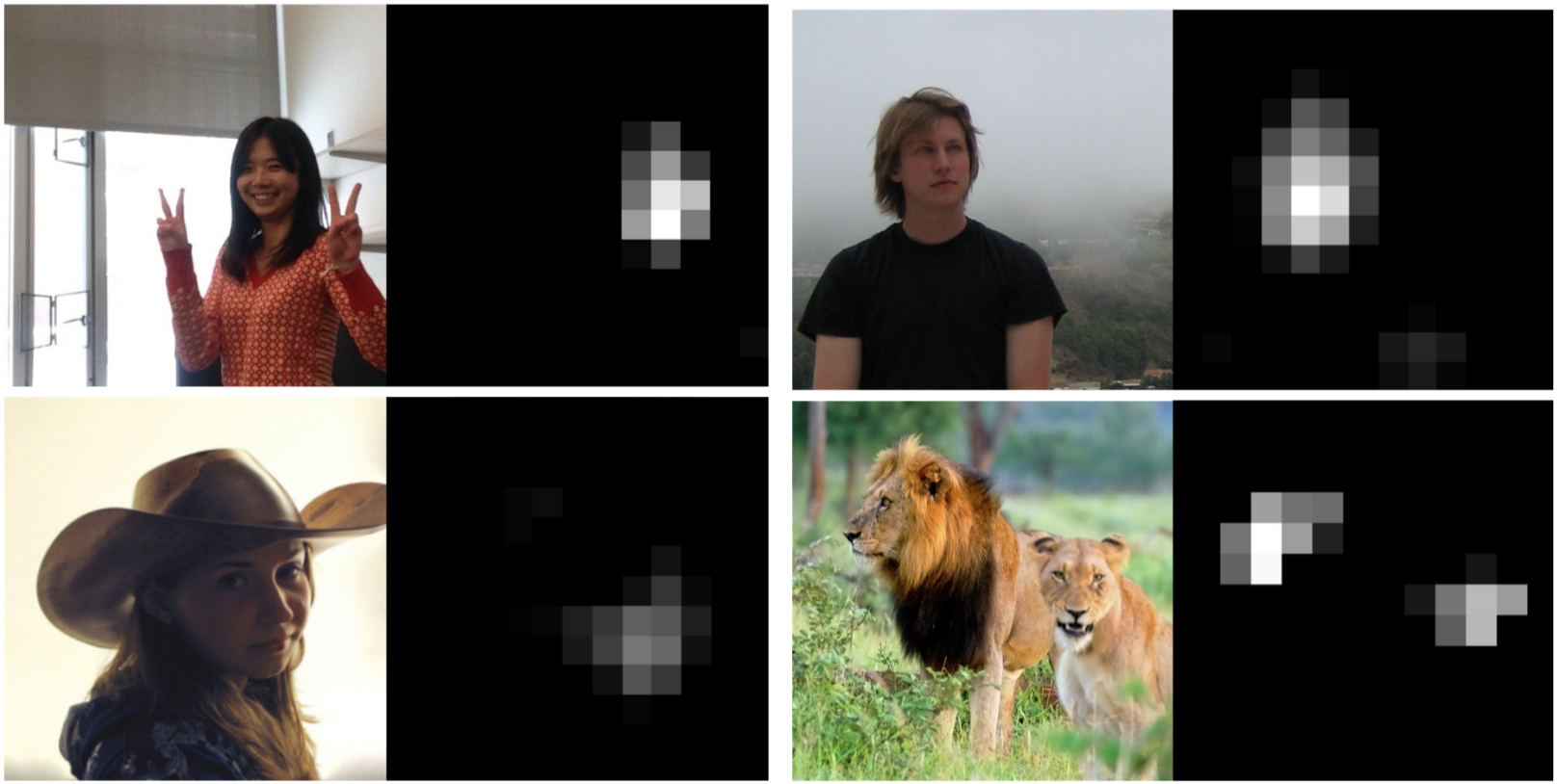
Emergent higher level abstractions

- Look at output of filter in 5th layer!



Emergent higher level abstractions

- Look at output of filter in 5th layer!



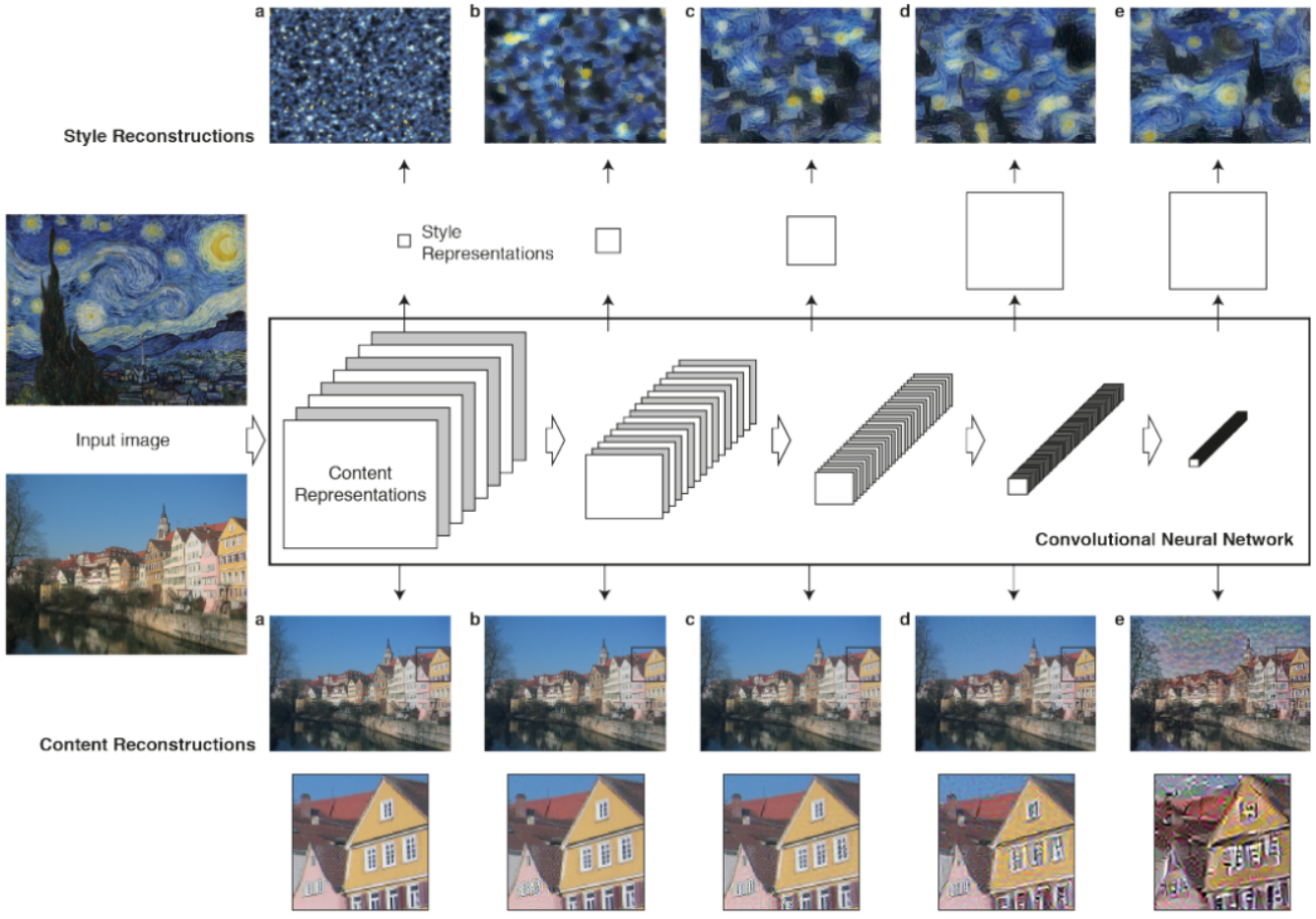
Yosinski et. al., ICML, google: deepvis

Google Deep Dream

http://deepdreamgenerator.com/

A Neural Algorithm of Artistic Style

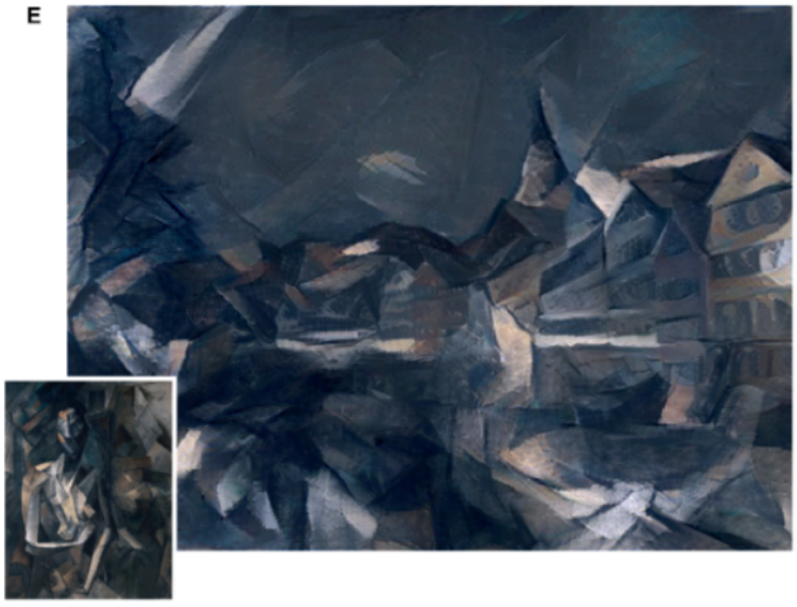
- Gatys, Ecker and Bethge
  `http://arxiv.org/pdf/1508.06576v2.pdf`
- Idea: Separate content and style



Borrowing the style from the masters!

Borrowing the style from the masters!

Borrowing the style from the masters!



Here should have been a live demo

DTU president

©Anders Boesen Lindbo Larsen

Neural artistic style - The Movie

https://www.youtube.com/watch?v=Khuj4ASldmU

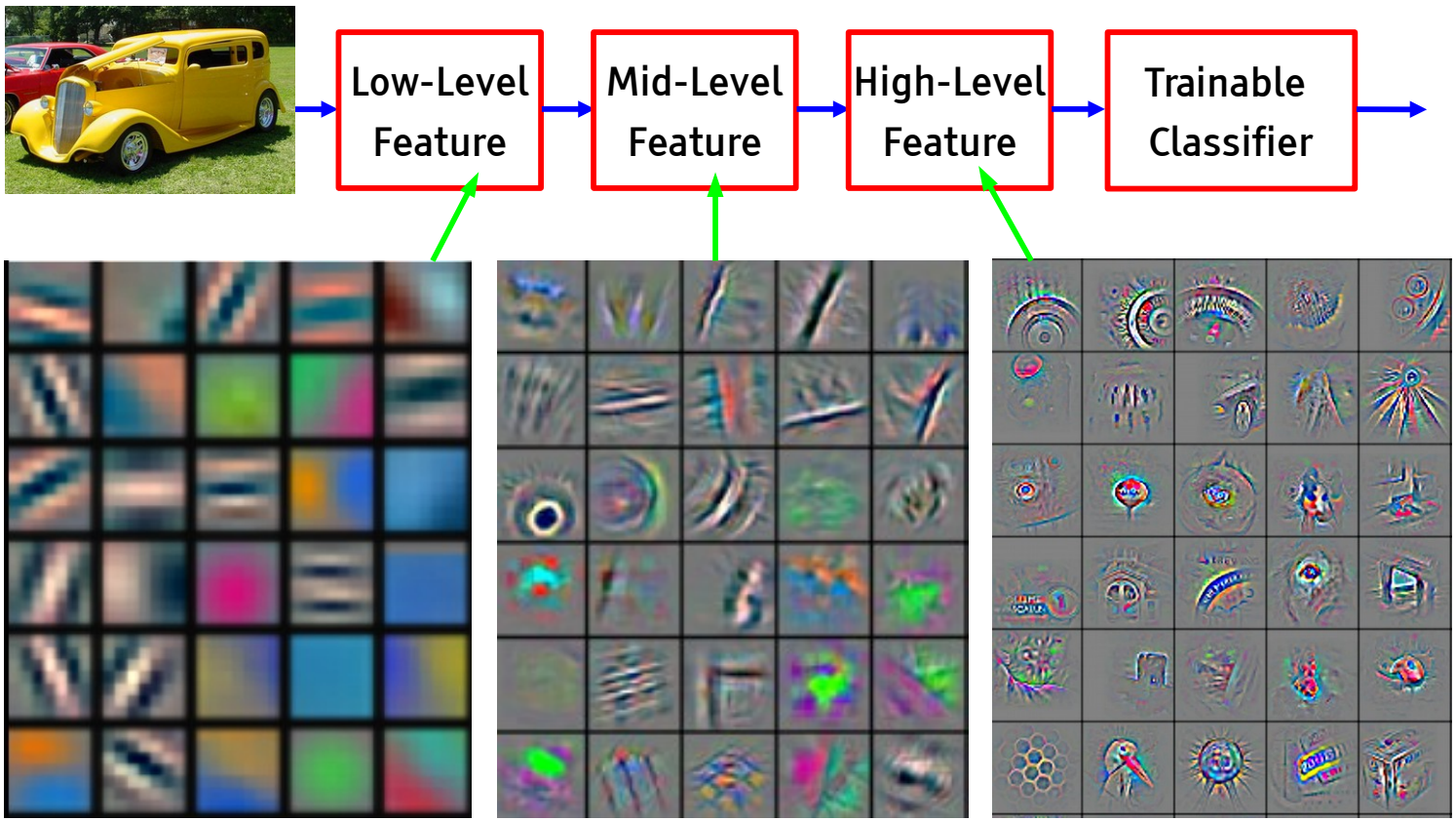
Part 2:

Convolutional NNs
The details

Deep Learning = Learning Hierarchical Representations

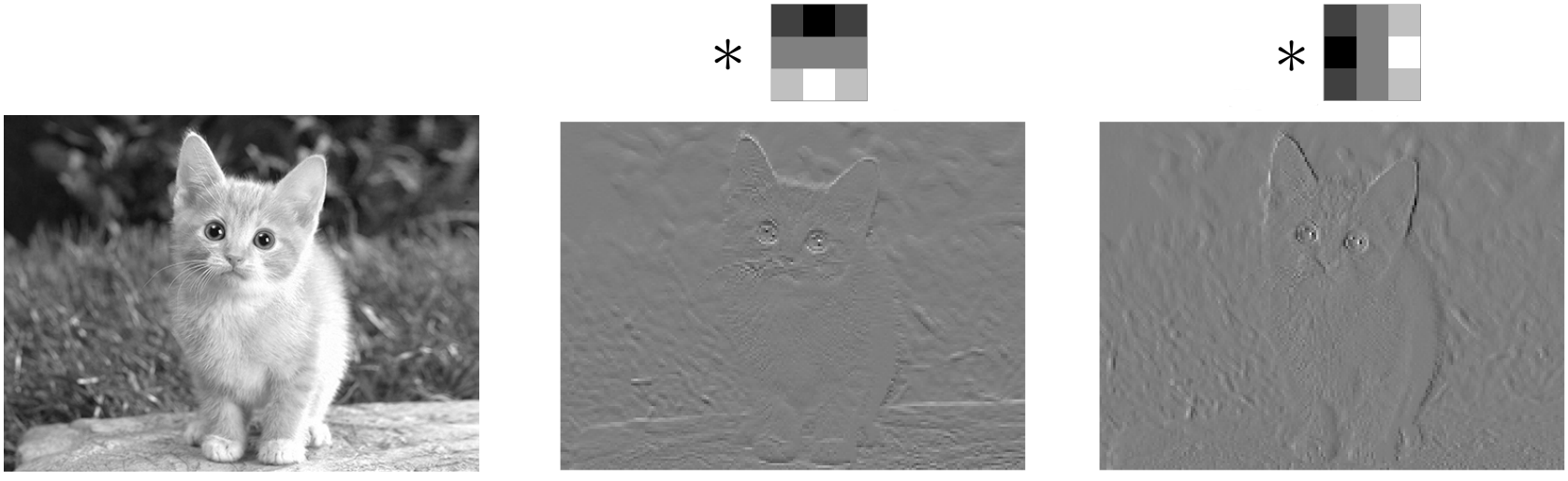
Y LeCun

- It's deep if it has more than one stage of non-linear feature transformation



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Convolution $*$



$$(w * x)[t] = \sum_a w[a]x[t-a]$$

Convolutional Networks (LeCun et al., 1989)

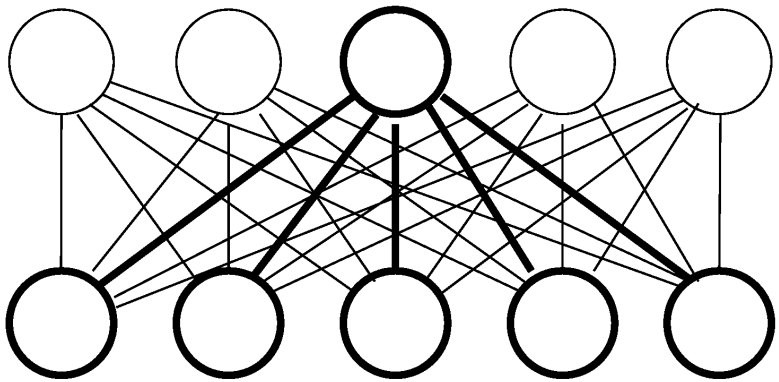
- How would MNIST intro network scale to real images?
- Instead of hundreds of pixels, a million.
- Would weight matrices $\mathbf{W}$ be million $\times$ million?

Convolutional Networks (LeCun et al., 1989)

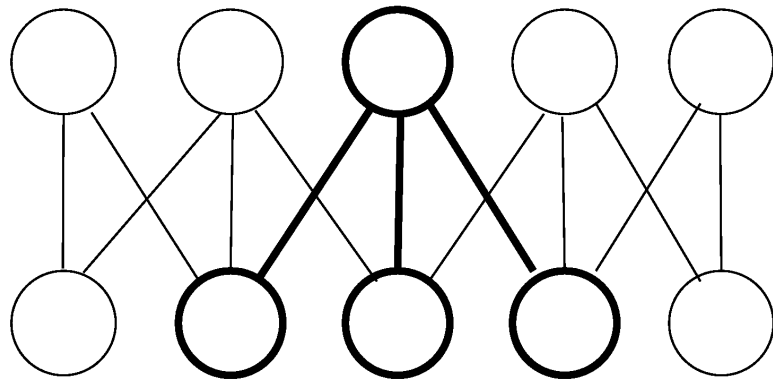
- How would MNIST intro network scale to real images?
- Instead of hundreds of pixels, a million.
- Would weight matrices $\mathbf{W}$ be million $\times$ million?

- Change notation: Replace vector signals with tensors indexed by x, y, c for x, y coordinates and channel c.
- Weight between $h^{(1)}_{x_1,y_1,c_1}$ and $h^{(2)}_{x_2,y_2,c_2}$ is $W^{(2)}_{x_1,x_2,y_1,y_2,c_1,c_2}$.
- Two ideas: Local connectivity and parameter sharing

Local connectivity



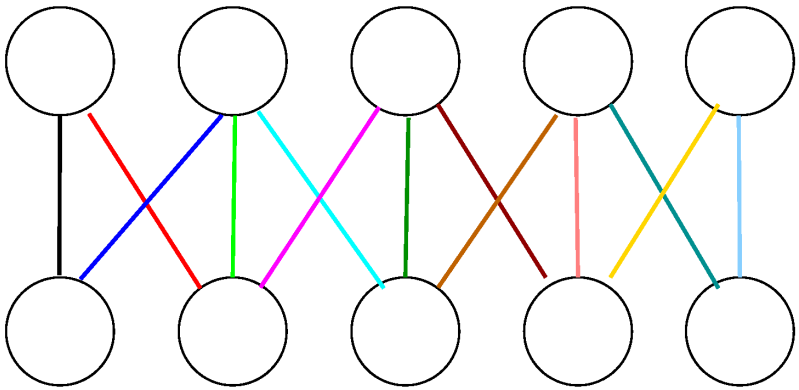
25 parameters



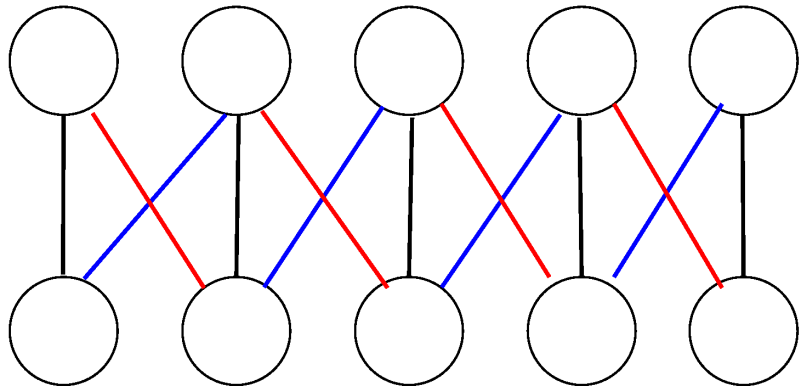
13 parameters

- Weight is zero unless $|x_1 - x_2| \leq k$ and $|y_1 - y_2| \leq k$

Parameter sharing



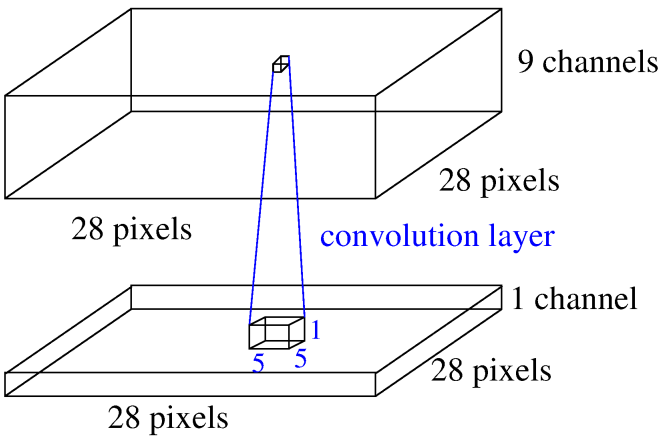
13 parameters

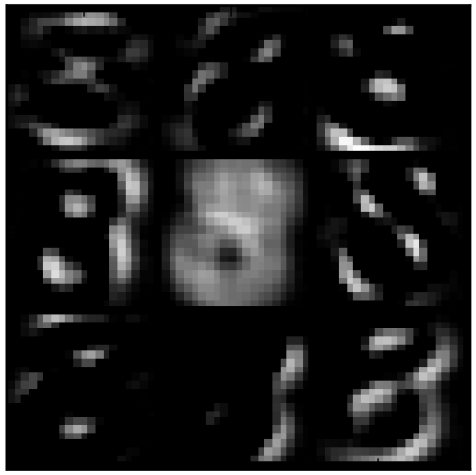


3 parameters

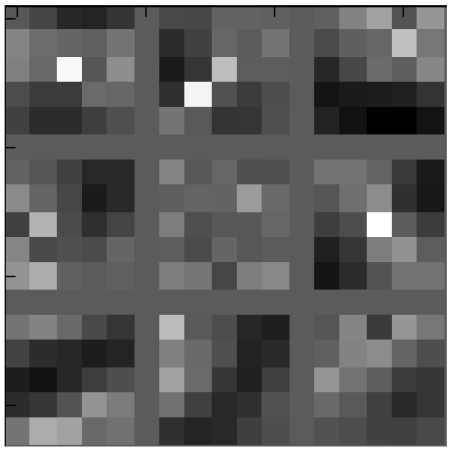
- Weights indexed by $\Delta x = x_1 - x_2$ and $\Delta y = y_1 - y_2$.
- $W_{\Delta x, \Delta y, c_1, c_2}$ replaces $W_{x_1, x_2, y_1, y_2, c_1, c_2}$.

Example (MNIST again)

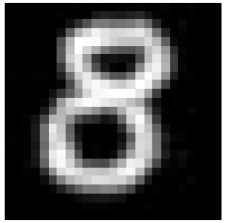




signals



weights



- MNIST data is 28×28 pixels and 1 channel.
- First hidden layer has 28×28 pixels and 9 channels.
- Use 5×5 filters ($\Delta x \leq 2$ and $\Delta y \leq 2$).

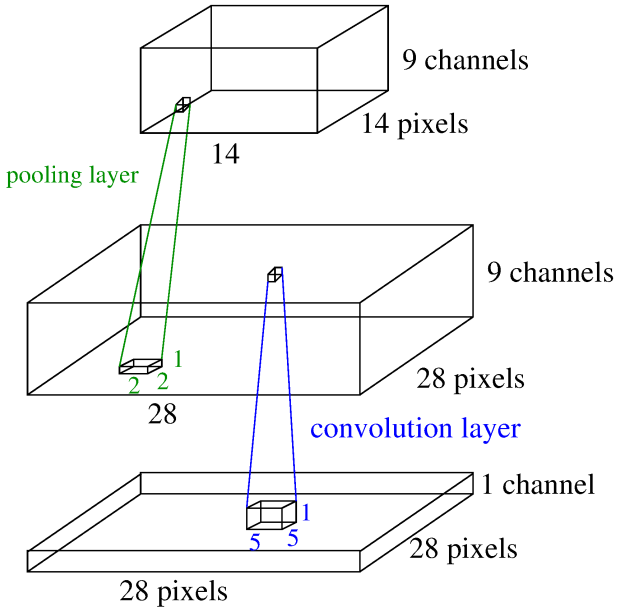
Hence Convolutional

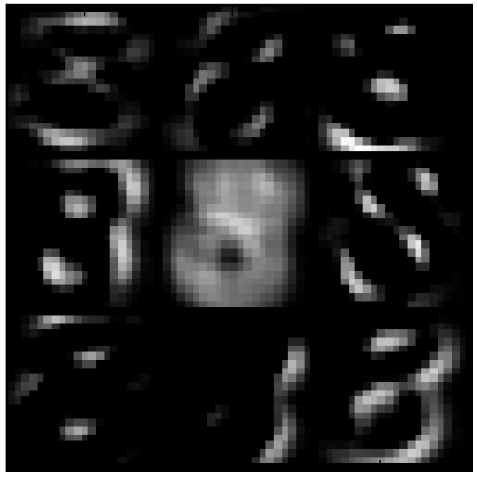
$$h_{x,y,c_1} = \text{relu}\left(\sum_{c_0,\Delta x,\Delta y} W_{\Delta x,\Delta y,c_0,c_1} x_{x+\Delta x,y+\Delta y,c_0} + b_{c_1}\right)$$

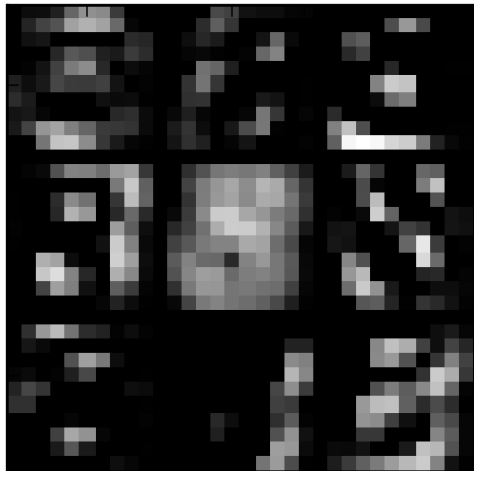
$$\Leftrightarrow \mathbf{h}_{:,:,c_1} = \text{relu}\left(\sum_{c_0} \mathbf{W}_{:,:,c_0,c_1} * \mathbf{x}_{:,:,c_0} + b_{c_1}\right)$$

- Can be written using the convolution operator $*$.
- (Convolution involves flipping one of the two inputs around.
- This is not needed, so the actual operation is cross-correlation.)
- **Padding:** Input can be padded with zeroes to keep the same size.

Pooling

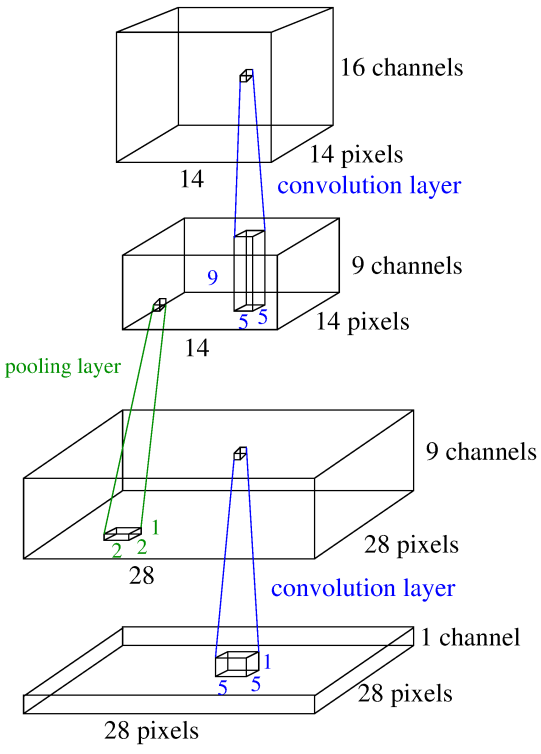


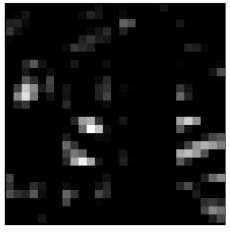


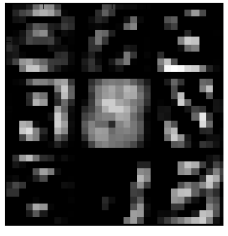


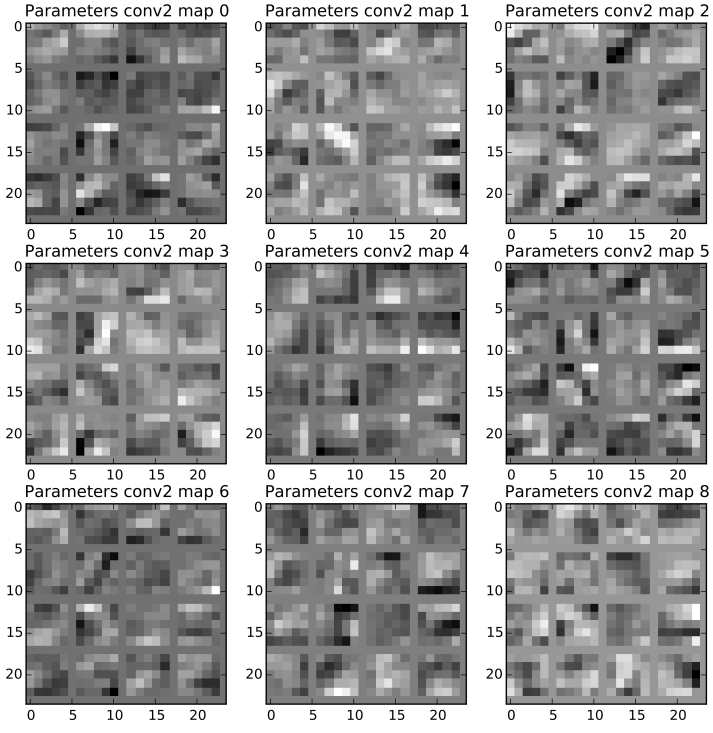
- Decrease resolution and increase channels going up.
- Often down-sampling by hard-coded pooling layers.
- We use $\max(\cdot)$ of 2×2 activations.
- Analysis of pooling functions (Boureau et al., 2010)

Stack more layers



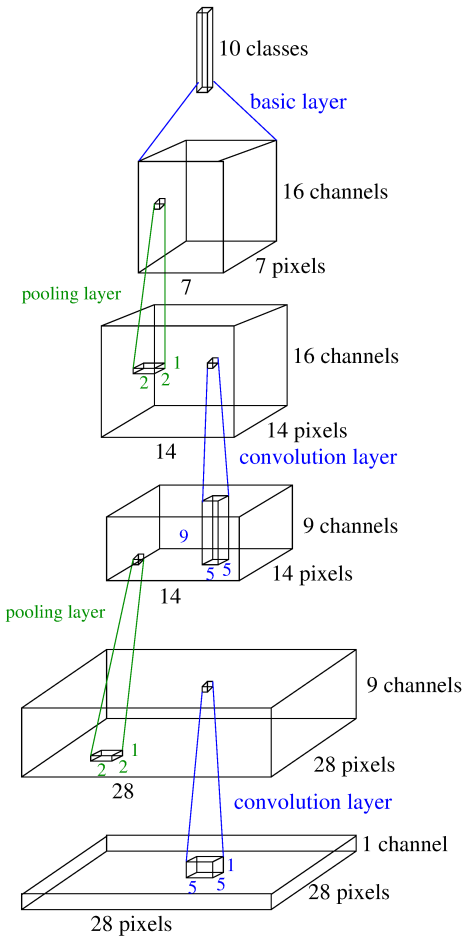


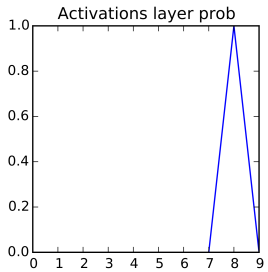


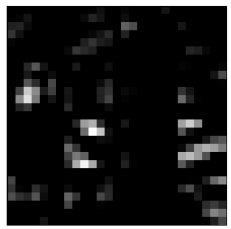


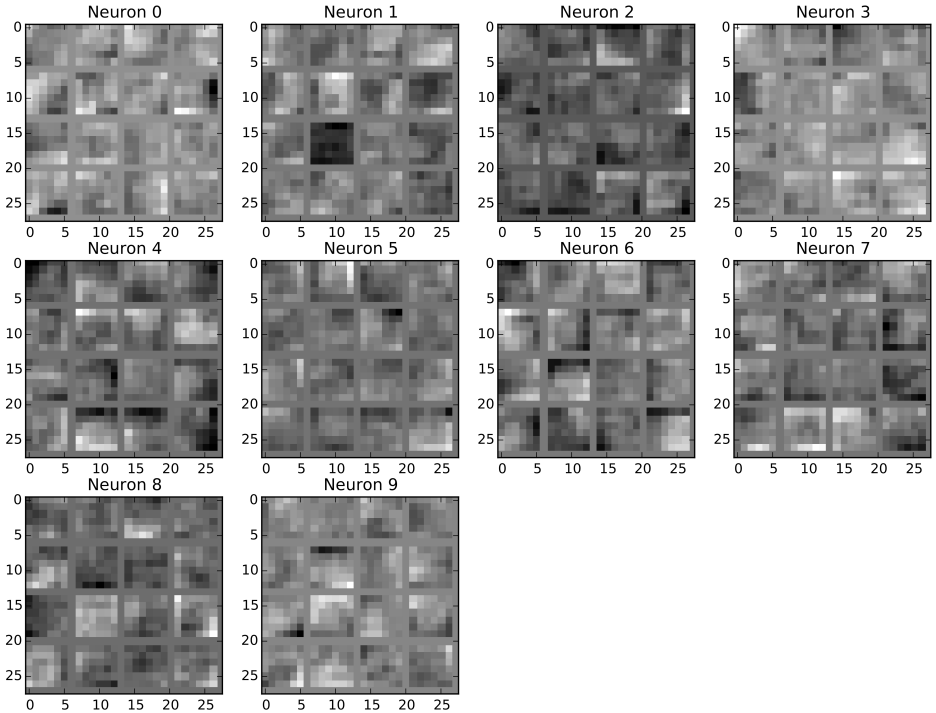
- Note how each unit looks at all channels of the previous layer.

Whole network



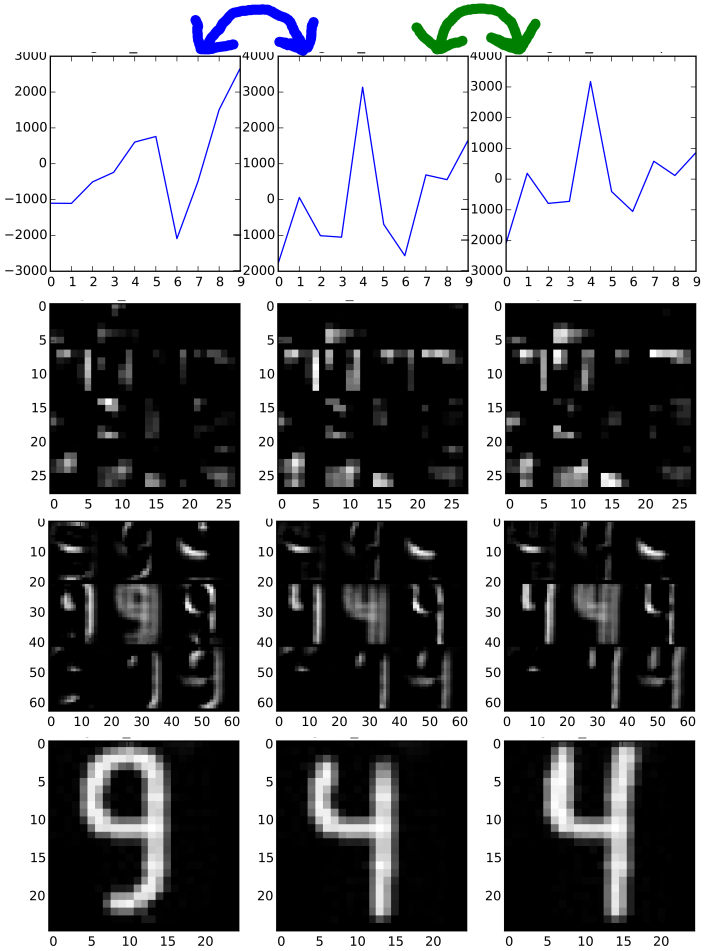


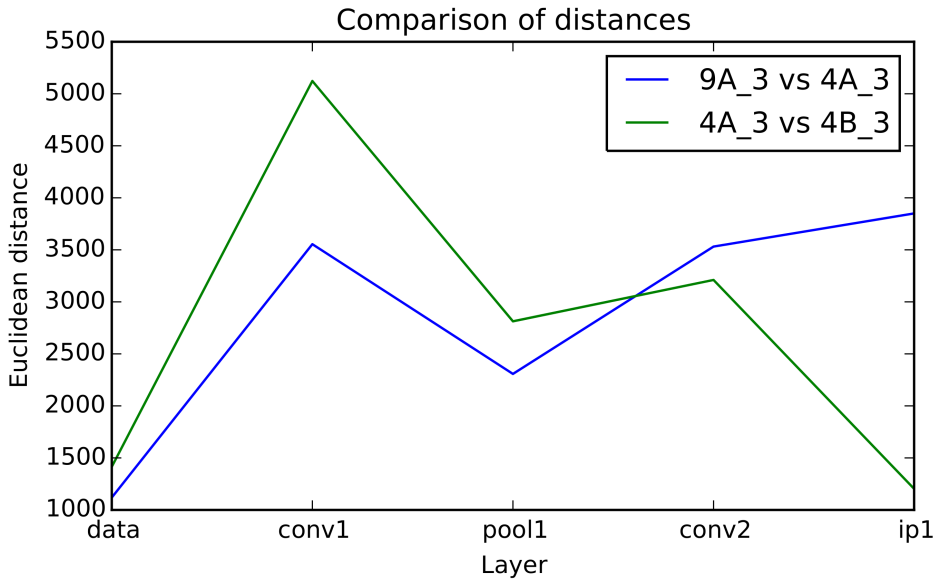




- At the top, the resolution drops to
- 1×1 pixel.
- Train by back-propagation as before.

Good representation?





- In pixel space, 9–4 is closer.
- In feature space, 4–4 is closer.

Convolutional vs. non-convolutional

- Number of weights (ignoring biases):

  $5 \cdot 5 \cdot 9 + 5 \cdot 5 \cdot 9 \cdot 16 + 7 \cdot 7 \cdot 16 \cdot 10 = 225 + 3600 + 7840 = 11665$
- Sizes of signals $\mathbf{h}$:

  $28 \cdot 28, 28 \cdot 28 \cdot 9, 14 \cdot 14 \cdot 16 = 784, 7056, 3136$

Convolutional vs. non-convolutional

- Number of weights (ignoring biases):

  $5 \cdot 5 \cdot 9 + 5 \cdot 5 \cdot 9 \cdot 16 + 7 \cdot 7 \cdot 16 \cdot 10 = 225 + 3600 + 7840 = 11665$
- Sizes of signals $\mathbf{h}$:

  $28 \cdot 28, 28 \cdot 28 \cdot 9, 14 \cdot 14 \cdot 16 = 784, 7056, 3136$

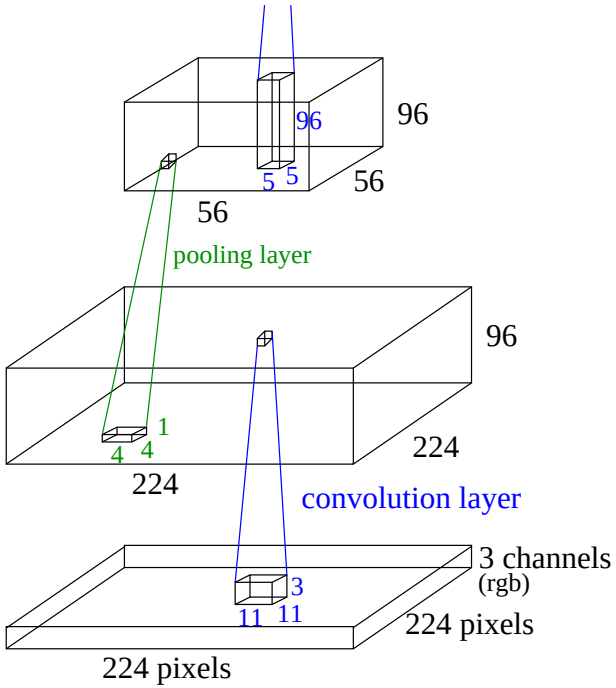
- Compare to the introductory MNIST FFNN example
- Weights:

  $28 \cdot 28 \cdot 225 + 225 \cdot 144 + 144 \cdot 10 =$
  $176400 + 32400 + 1440 = 210240$
- Signals:

  $784, 225, 144$

- Convolutional network has more signals but less params.

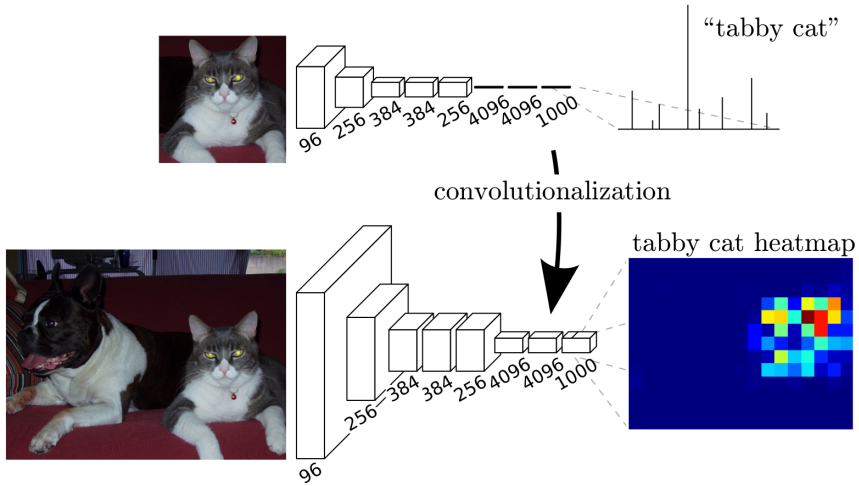
Scaling up

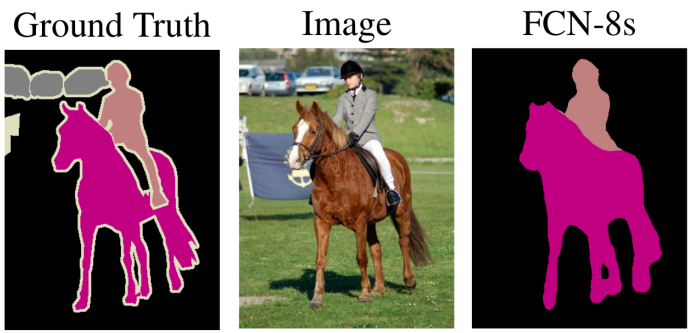


Ipython notebook example

- First layers of (Krizhevsky, 2012).
- Trend in 2015 towards small filters (3×3) and poolings (2×2),
- but lots of layers (10–40).

Semantic segmentation (Long et al., 2015)

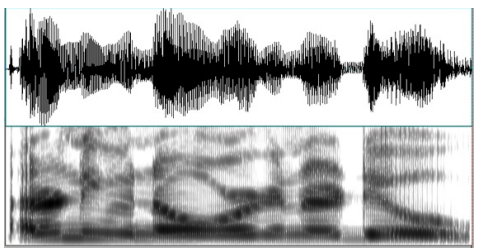


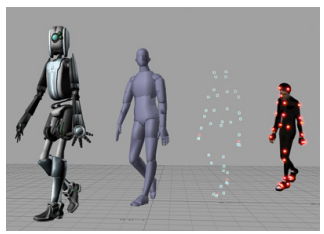


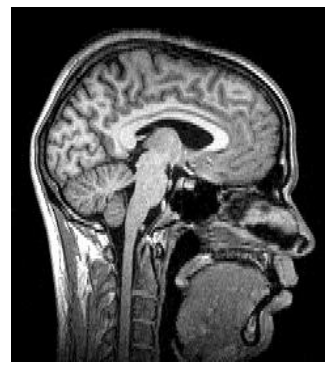
- Sliding a convolutional network to classify each location.
- Lots of shared computation.

Other applications

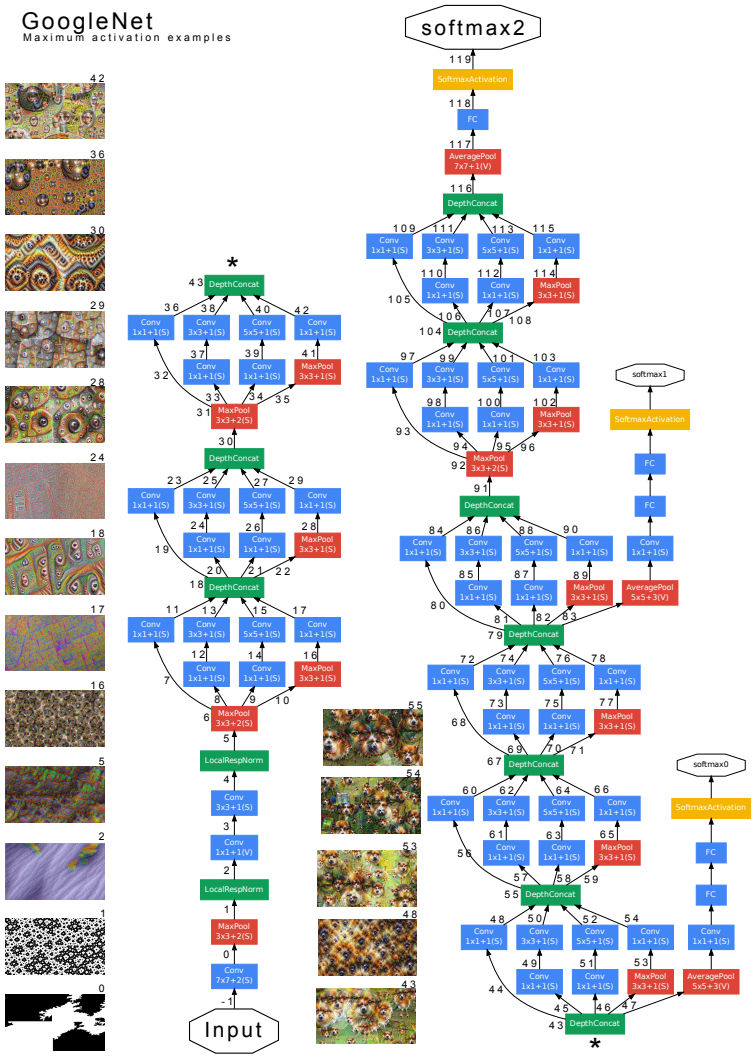
Tensor	Single channel	Multi-channel
1-D	Raw audio (mono)	Motion capture
2-D	Audio + Fourier transform	Game of Go
3-D	Brain imaging	Colour video

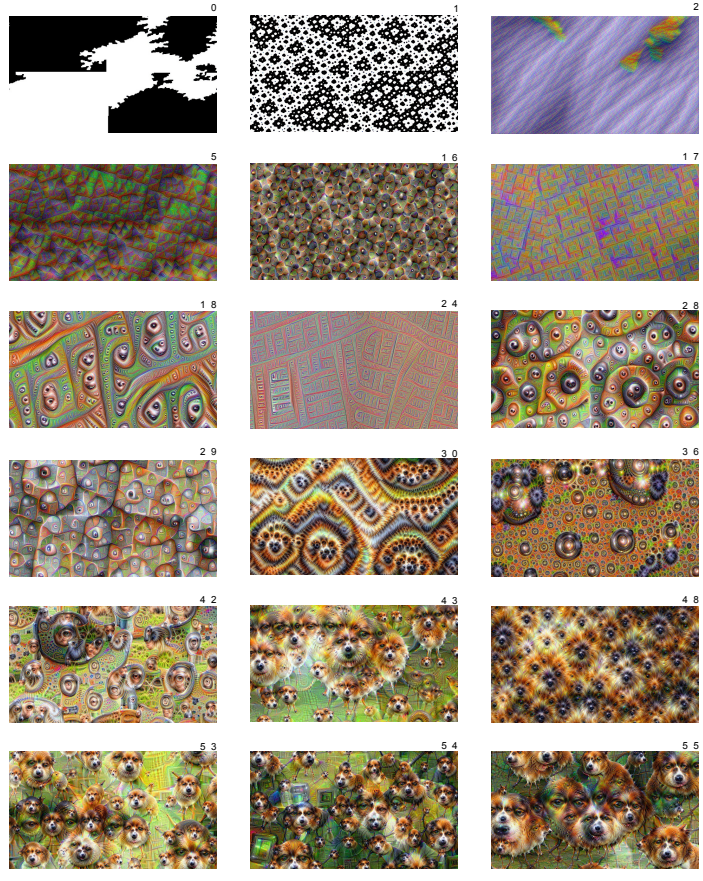






Inceptionism (Mordvintsev, 2015) Video by Miquel Perello Nieto





https://youtube.com/watch?v=w5U7EL72ngI

users.ics.aalto.fi/perellm1/deep_dreams.shtml

References

- LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4), 541-551.
- Boureau, Y-Lan, Jean Ponce, and Yann LeCun. "A theoretical analysis of feature pooling in visual recognition." Proceedings of the 27th International Conference on Machine Learning (ICML-10). 2010.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).
- Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." arXiv preprint arXiv:1411.4038 (2015).
- Mathieu, Michael, Mikael Henaff, and Yann LeCun. "Fast training of convolutional networks through FFTs." arXiv preprint arXiv:1312.5851 (2013).
- Kivinen, Jyri J., and Christopher KI Williams (2011). "Transformation equivariant Boltzmann machines." Artificial Neural Networks and Machine Learning (ICANN).
- Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." Computer Vision–ECCV 2014. Springer International Publishing, 2014. 818-833.
- Mordvintsev, A., Olah, C., & Tyka, M. (2015) "Inceptionism: Going Deeper into Neural Networks", Google research blogspot

Thanks!
Ole Winther