Computational Intelligence Laboratory

Lecture 8

Convolutional Neural Networks & Generative Models

Aurelien Lucchi & Thomas Hofmann

ETH Zurich – `cil.inf.ethz.ch`

May 5, 2017

Section 1

Convolutional Neural Networks

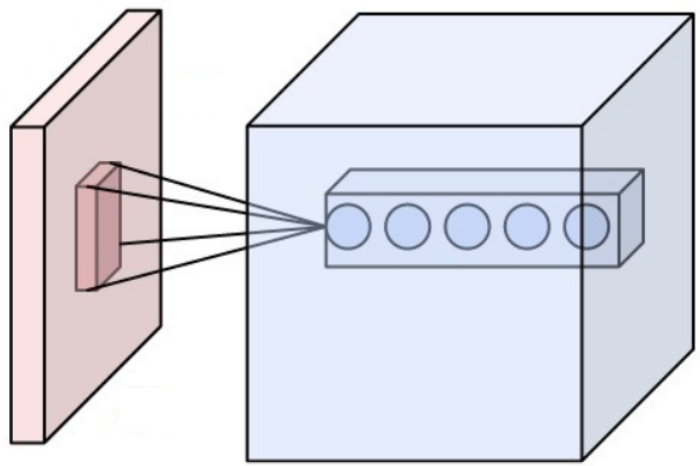
No Free Lunch!

- No learning machine can do well on all problems.
- Need to constrain function class appropriately.

Neural Networks for Images: Receptive Fields

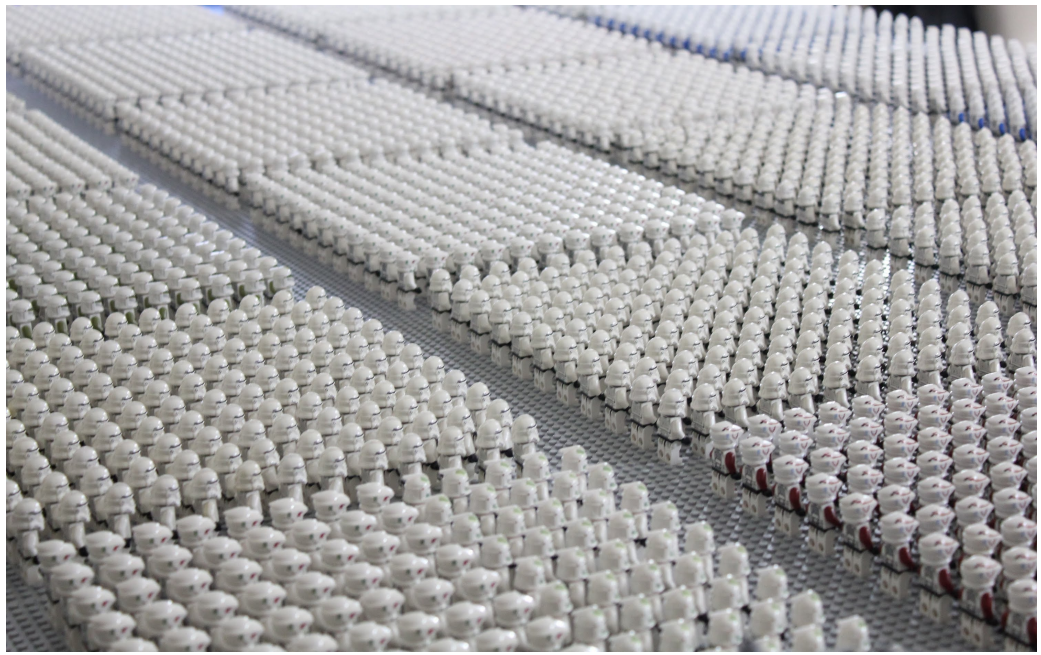
- Topological connectivity
  - encourage network to first extract localized features
  - subsequent layers: less and less localized features

- Receptive field
  - inputs that can affect a neuron (other weights $= 0$)
  - small images patches as receptive fields
  - can have multiple channels (in figure: 5)



Neural Networks for Images: Translation Invariance

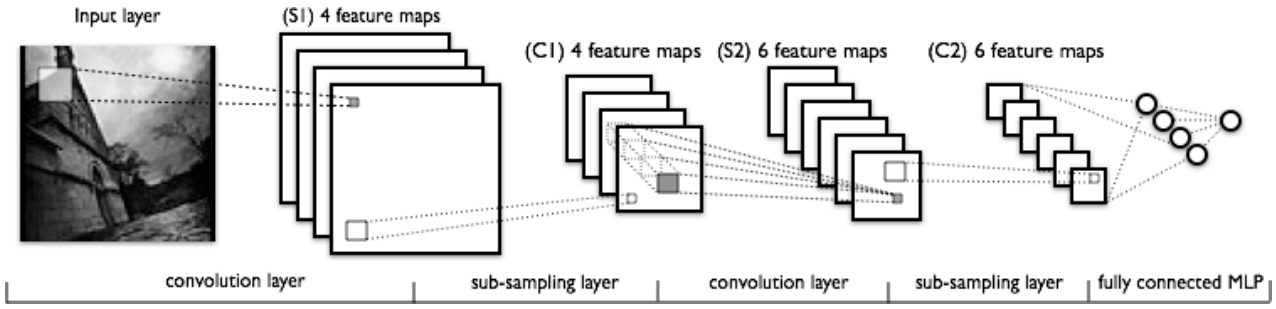
- Translation invariance of images
  - image patches look the same, irrespective of their location
  - idea: extract translation invariant features
  - what does that mean for a neural network?



Neural Networks for Images: Weight Sharing

- Weight Sharing
  - neurons share the same weights = compute same function
  - differ in location of their receptive field = different input
  - mirrors what has been done in image processing (manually)
- Shift-invariant Filters
  - layers learn shift-invariant filters
  - weights define a filter mask (e.g. 3x3 or 5x5)
  - typically as many neurons as inputs (border padding etc.)
    - e.g. 64x64 pixel per image $\Rightarrow$ 64x64 neurons per channel
  - color images: 3 color channels, 3-dimensional filter mask

CNN: Buildings blocks



- **Three building blocks:**
  - Convolutional layer
  - Pooling layer
  - Fully-connected layer

Convolutional Layers

- **Convolution:**
  - Mathematical operation on two functions (f and g)
  - It produces a third function that is typically viewed as a modified version of one of the original function
  - This operation can be used to detect edges in an image

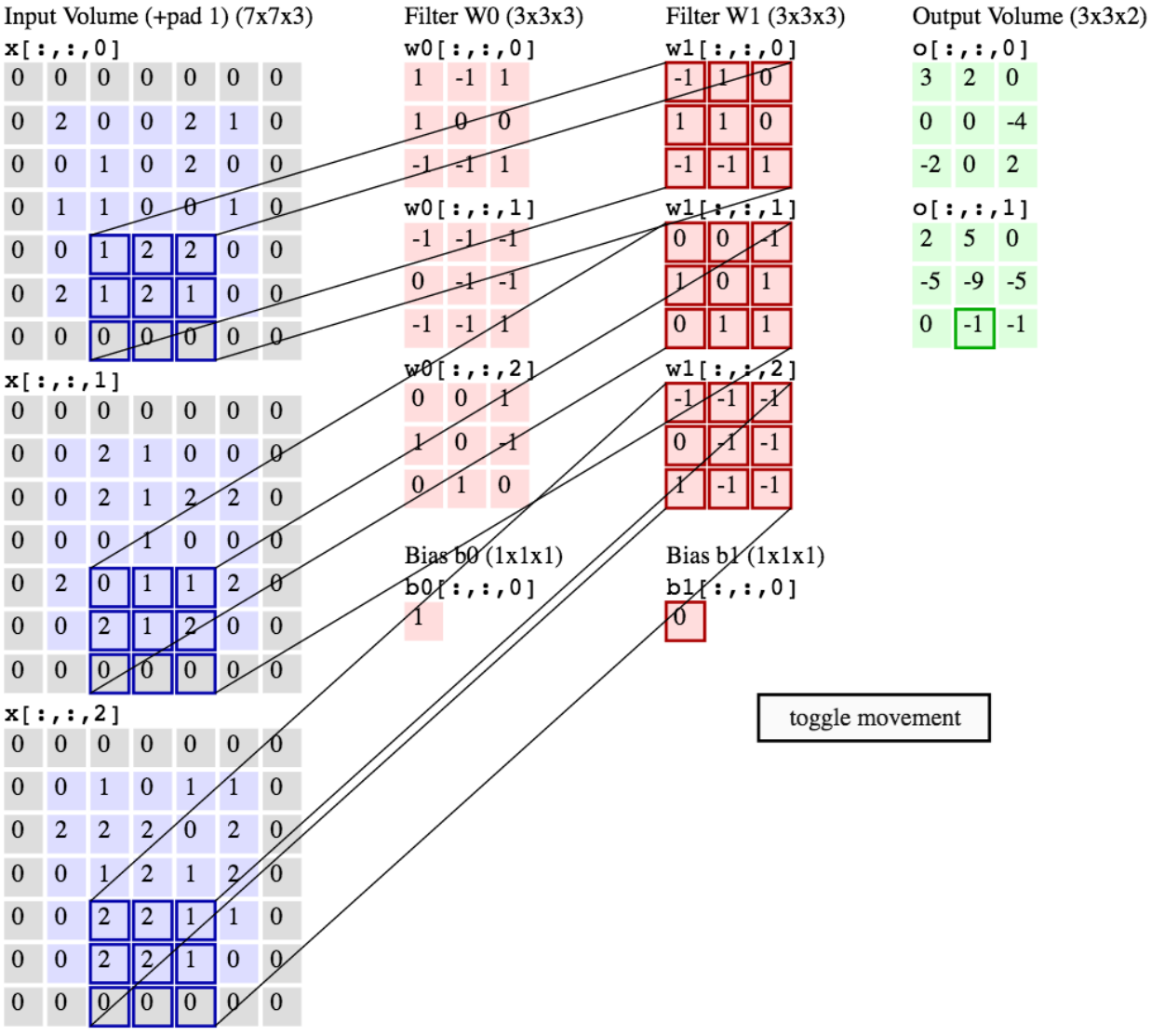
-1	0	1
-2	0	2
-1	0	1

Horizontal

-1	-2	-1
0	0	0
-1	-2	-1

Vertical

Convolutional Layers: Animation



cs231n.github.io/assets/conv-demo/index.html

Convolutional Layers: Mathematics

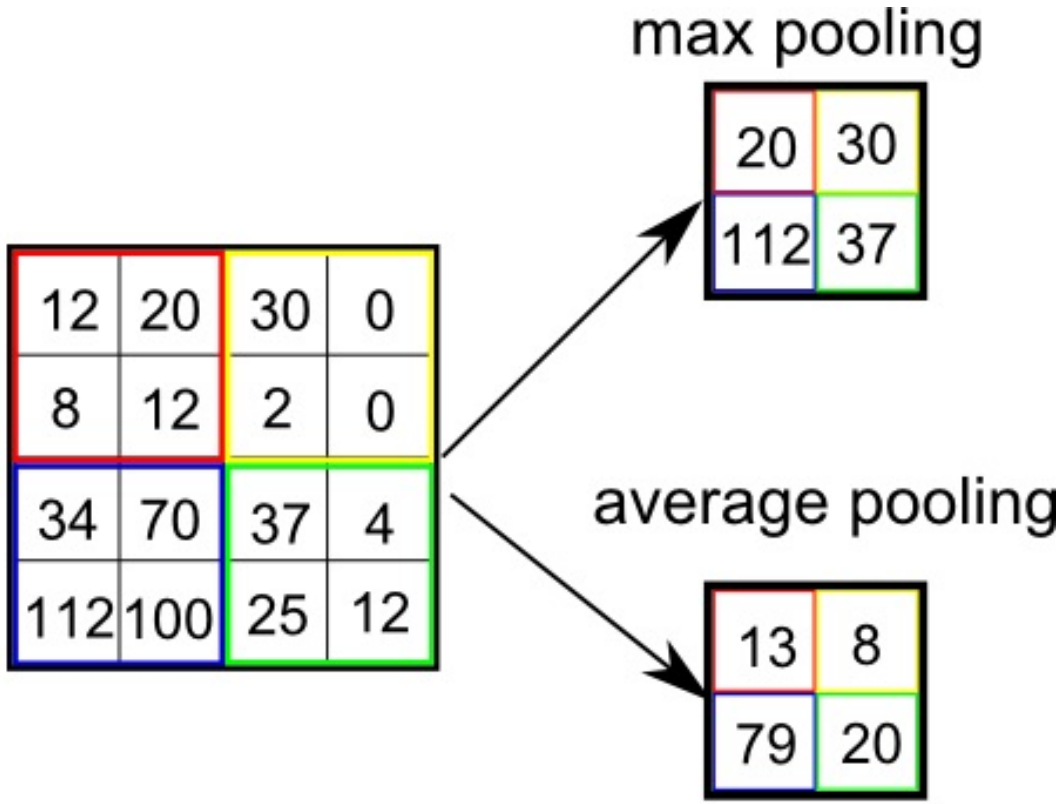
- Convolution in 2D (5x5)

$$F_{n,m}(\mathbf{x};\mathbf{w}) = \sigma\left(b + \sum_{k=-2}^{2}\sum_{l=-2}^{2} w_{k,l} \cdot x_{n+k,m+l}\right)$$

  - (n, m): center of receptive field
  - $\mathbf{x}$: image (2D pixel field)
  - $\mathbf{w}$: weights = arranged as a 2D mask
  - related to convolution in mathematics

Pooling

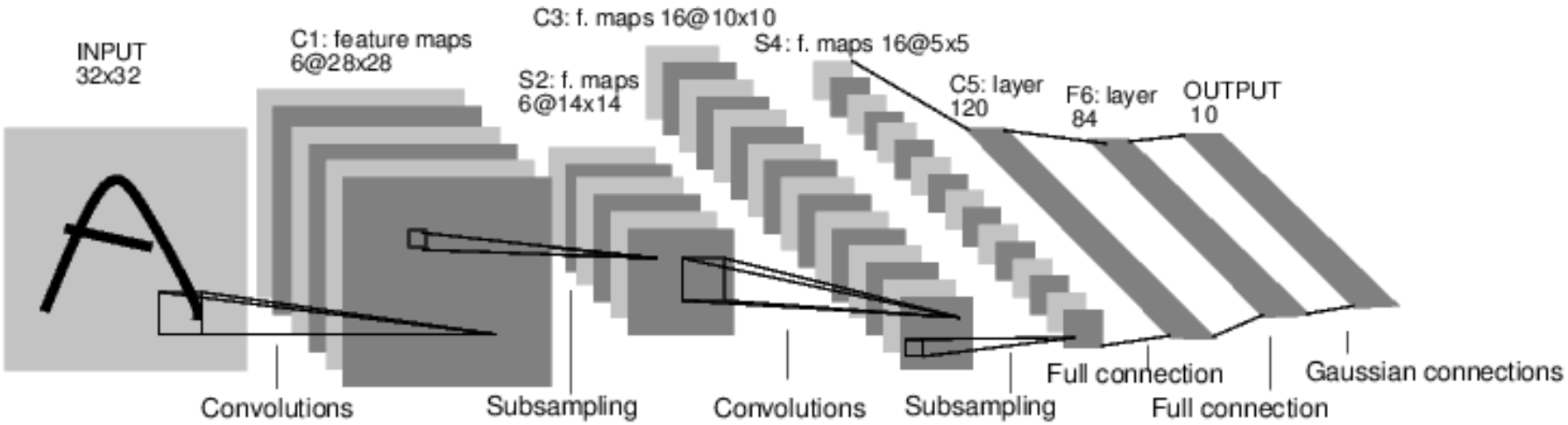
- Reduce size of convolutional layers by down-sampling
- Take average over window (e.g. 2x2)
- Common practice: max pooling = take maximum in window



Fully-connected layer

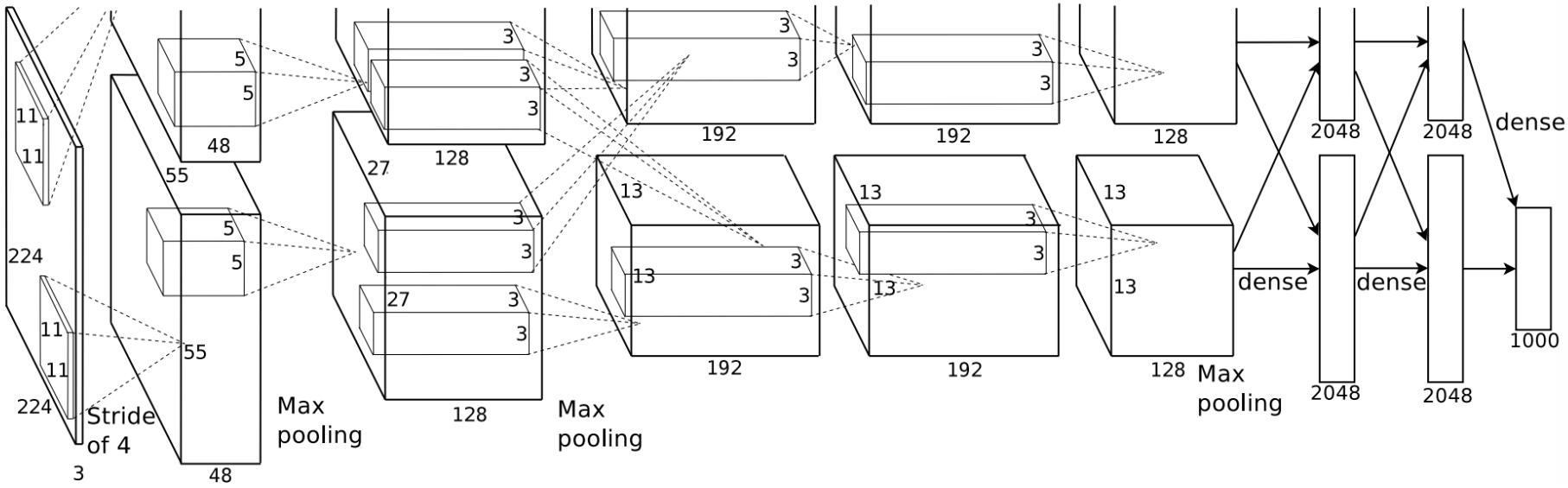
- High-level reasoning
- Connects all neurons in the previous layer to every single neuron it has
- Can be computed with a matrix multiplication

LeNet5



- Architecture LeNet5
  - layers C1/S2: 6 channels, cutting at border, 2x subsampling (4704 neurons)
  - layers C3/S4: 16 channels, cutting at border, 2x subsampling (1600 neurons)
  - layers F5/F6: fully-connected
  - output: Gaussian noise model (squared loss)

AlexNet



- AlexNet
  - Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, 2012 ImageNet Classification with Deep Convolutional NN
  - 60 million parameters and 500,000 neurons
  - 5 convolutional layers, some followed by max-pooling
  - 2 globally connected layers with a final 1000-way softmax

Learning the Filters

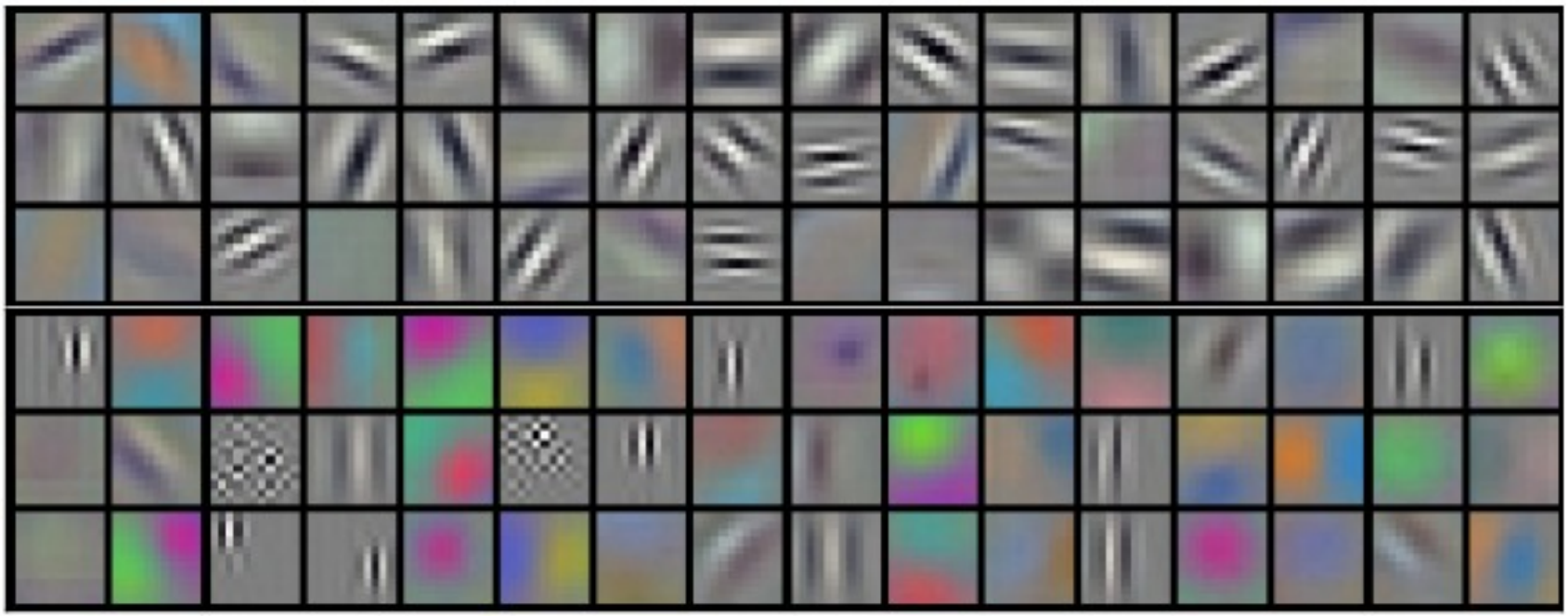
- Recall from last week: Optimize using stochastic gradient descent

$$\theta \leftarrow (1 - \eta\lambda)\theta - \eta \nabla_\theta L(y_t^*; y(\mathbf{x}_t; \theta))$$

- What do the filters look like then?

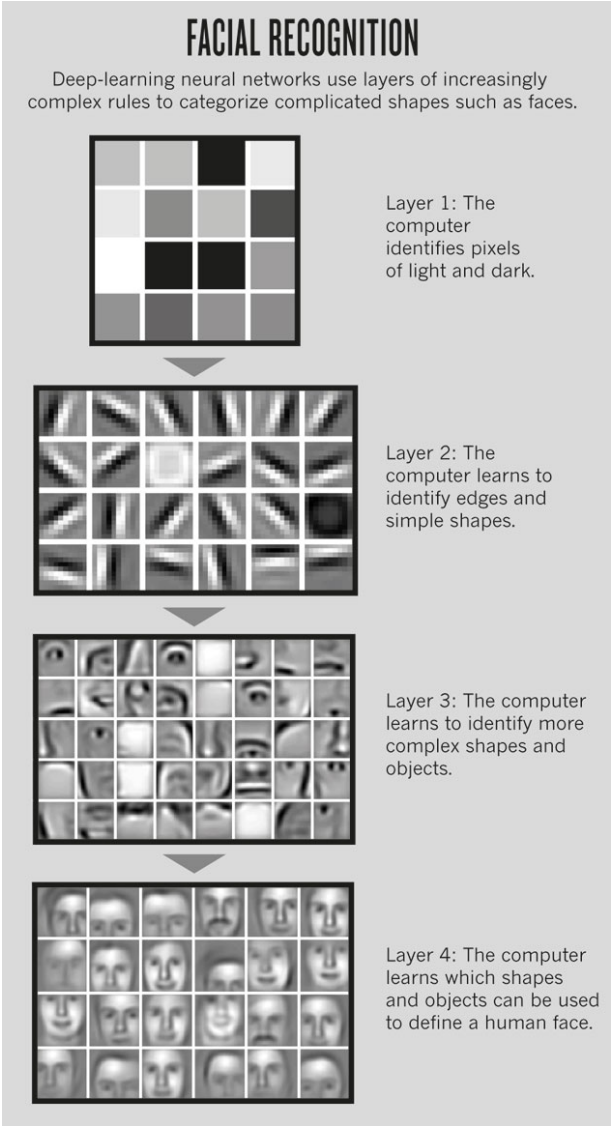
Learning Local Image Features

- Example: filters learned at first layer
- cf. Krizhevsky et al.: 96 filters of size 11x11x3



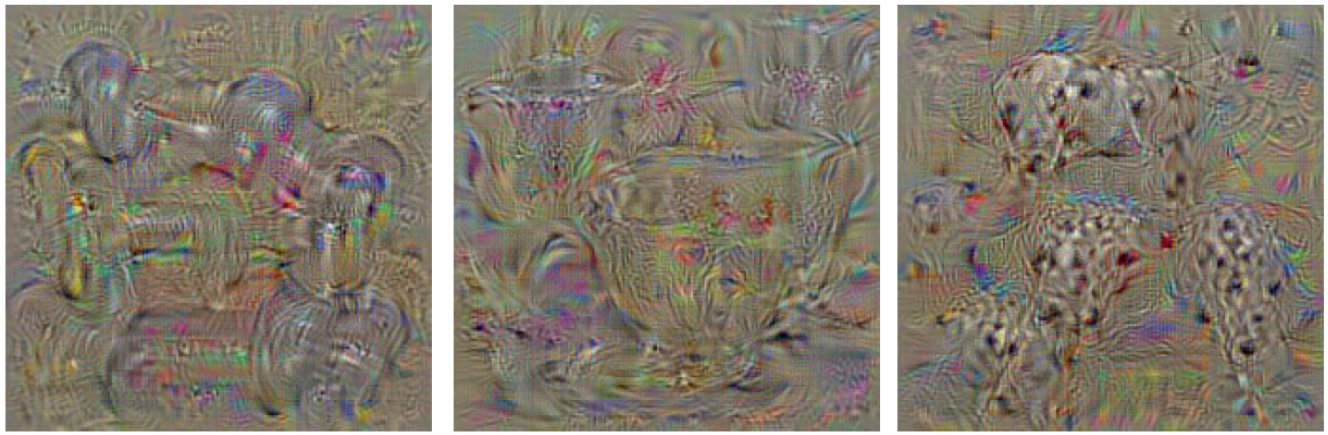
Learning Higher Level Features

- (c) Andrew Ng,
  trained on face images



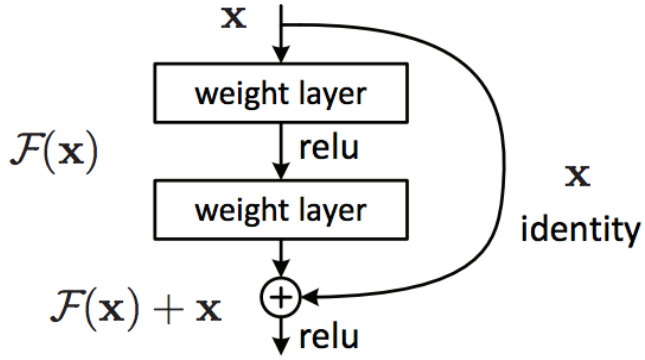
Saliency Maps

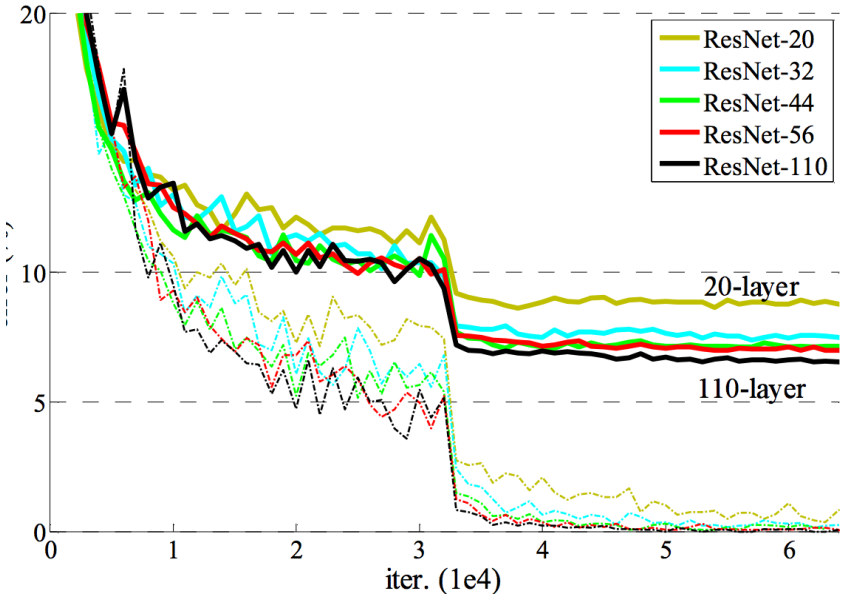
Per-class saliency maps for a CNN trained for visual classification (cf. Simonyan et al, 2015)



Deeper Nets

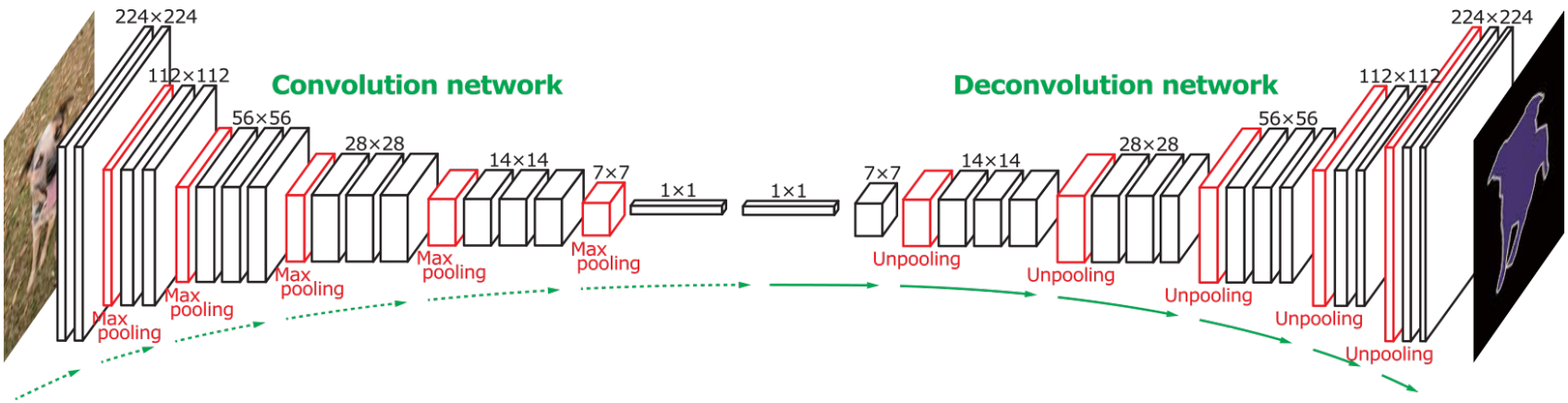
- ImageNet 2015 (Dec):
  - winner: residual networks
  - more than 100 layers deep





Semantic Segmentation

- CNNs can also be used for semantic segmentation.
- Typical architecture of a de-convolutional network (from Noh et al. 2015)



Section 2

Generative Models

Generative Models

- Goal: Building generative models over space of images.
- Consider a probability distribution $p(\mathbf{x})$ with a large number of random variables $\mathbf{x} = \{x_1, \ldots, x_i, \ldots, x_n\}$
  - E.g. $\mathbf{x}$ is an image and x_i is the i-th pixel in the image
- Two extremes to model $p(\mathbf{x})$:
  - Consider the full joint distribution:
    $p(\mathbf{x}) = p(x_1, x_2, \ldots, x_n) = p(x_1|x_2, \ldots, x_n)p(x_2, \ldots, x_n) \ldots$
  - Consider all the variables as independent:
    $p(\mathbf{x}) = p(x_1)p(x_2) \ldots p(x_n)$

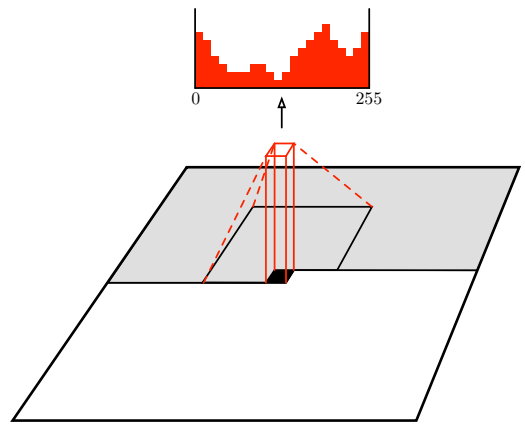
Approaches to Learn a Generative Model

- Popular approach: Autoregressive models such as PixelCNN (A. van den Oord et al. 2016) train a network that models the conditional distribution of every individual pixel given previous pixels (to the left and to the top).
- Other approaches (out of scope): Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs)

Pixel CNN

- Model joint distribution of pixels over image $\mathbf{x}$ as product of conditional distributions, where x_i is a single pixel:

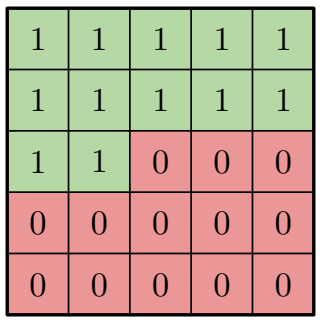
$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p(x_i | x_1, ..., x_{i-1})$$



- Visualization on the left: generate pixel x_i by conditioning on previously generated pixels $x_1, \ldots x_{i-1}$
- Ordering of the pixel dependencies is in raster scan order: row by row and pixel by pixel within every row

Pixel CNN

- Need to make sure the CNN can only use information about pixels above and to the left of the current pixel
- Used to mask the 5x5 filters to make sure the model cannot read pixels below (or strictly to the right) of the current pixel to make its predictions

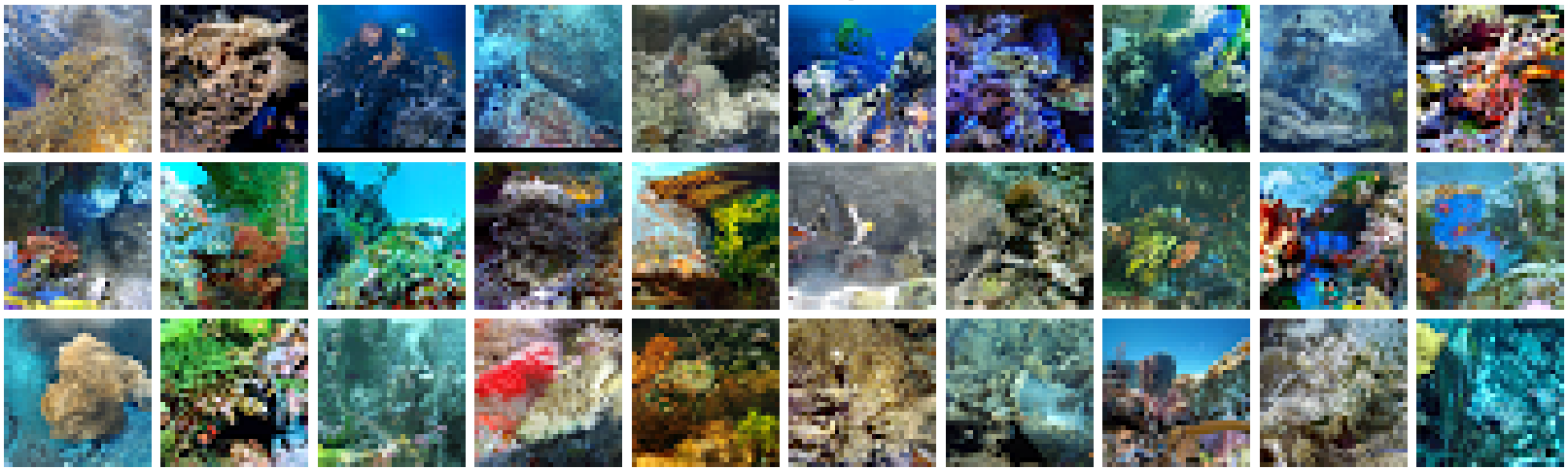


Prediction with Pixel CNN

- During sampling the predictions are sequential: every time a pixel is predicted, it is fed back into the network to predict the next pixel
- Drawback: Slow process

Prediction with Pixel CNN

African elephant



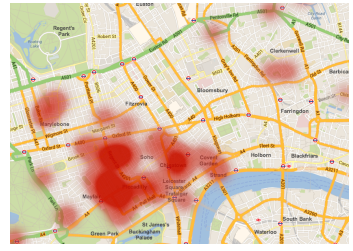
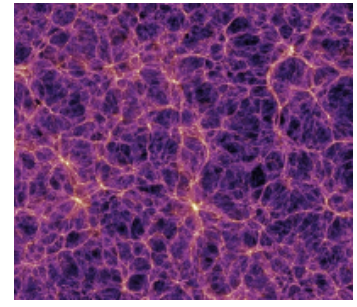
Coral Reef

Section 3

Projects

Project Overview

- Real world data sets and challenges – you pick one!

1	2	3	4
			
preferences recommender	texts & tweets sentiment	aerial imagery segmentation	cosmology galaxies

- Combine and extend techniques ⇒ novel solution
  - compare to baselines developed during the course
- Produce a write up of your findings ⇒ scientific short paper
  - emphasize experimental protocol, metrics, and reproducibility

Project 1: Collaborative Filtering

Viewers were asked to rate some movies (items):

	Ben	Tom	John	Fred	Jack
Star Wars	?	?	1	?	4
WallE	5	?	3	4	?
Avatar	3	4	?	4	4
Trainspotting	?	1	5	?	?
Shrek	5	?	?	5	?
Ice Age	5	?	4	?	1

- Not all viewers rated all movies.
- We want to predict unrated user-movie pairs (**matrix completion**)
- Should we recommend Fred to watch "Ice Age"?

Project 2: Sentiment Classification

Automatic sentiment analysis to give a machine the ability to understand text and its polarity.

- **Data**: we provide a large set of training tweets.
- **Ground-truth**: each tweet is labeled as {negative, positive}.

Positive: "i have the worlds best dad"

Negative: "pouring rain outside . wish i could go out"

- **Goal**: train classifier using word vectors to predict polarity

Project 3: Semantic Segmentation

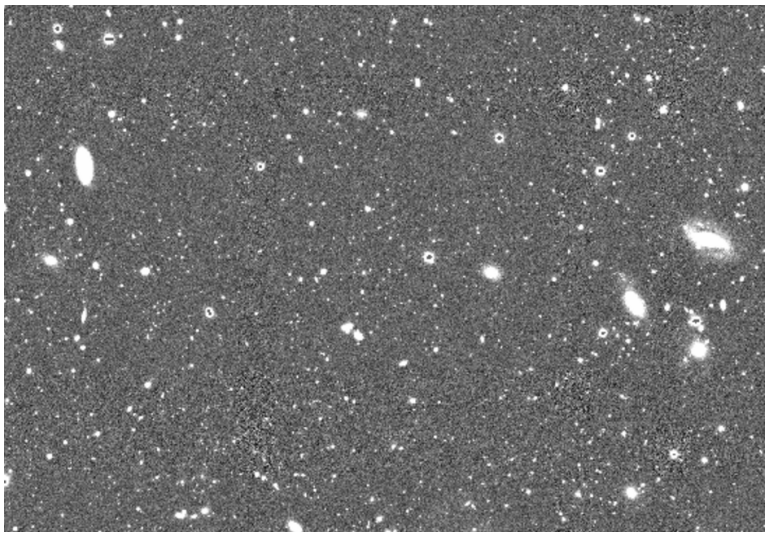
Extract roads from satellite images

- **Data**: set of satellite/aerial images acquired from GoogleMaps
- **Ground-truth**: images with pixels labeled as {road, background}.

- **Goal**: train a classifier to segment roads in these images, i.e. assign a label {road=1, background=0} to each pixel.
- **Can get familiar with the data in next exercise**.

Project 4: Galaxy Image Generation

- **Data**: astronomical images acquired from wide field imaging surveys.
- **Ground-truth**: images labeled as {cosmology, corrupted, background}



- **Goal**: train a generative model that can generate galaxy images.
- **Can get familiar with the data in next exercise**.

Computational infrastructure

- Euler, CPUs only, no GPUs
- Microsoft Azure
  - Will post detailed instructions on the web site
  - Caution: You pay a price per minute of use of the machine. A GPU machine is therefore quite expensive and cost around 1.6$ per hour, so make sure you shut down the machine in the portal.

Administrative

- Work in groups of three of four students (two **not** accepted)
- Deadline: July 1st, 2017
- Submit a 4-page report to CMT (see lecture about Scientific Writing)