Computer Vision - Lecture 15

Deep Learning for Object Categorization

21.12.2016

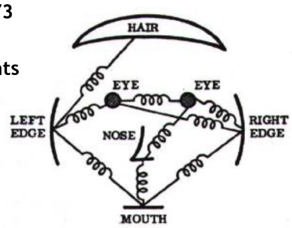
Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

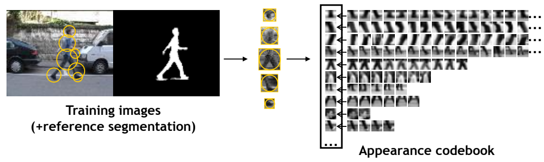
Course Outline

Recap: Part-Based Models

- Fischler & Elschlager 1973
- Model has two components
  - parts (2D image fragments)
  - structure (configuration of parts)



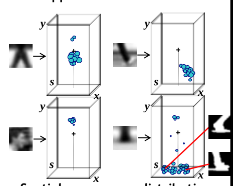
Recap: Implicit Shape Model - Representation



Training images (+reference segmentation)

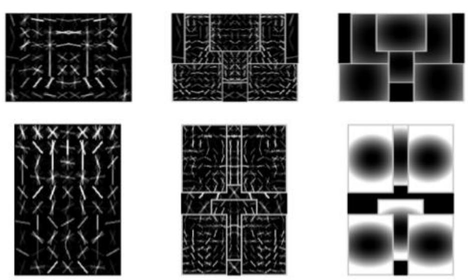
Appearance codebook

- Learn appearance codebook
  - Extract local features at interest points
  - Clustering ⇒ appearance codebook
- Learn spatial distributions
  - Match codebook to training images
  - Record matching positions on object



Spatial occurrence distributions
+ local figure-ground labels

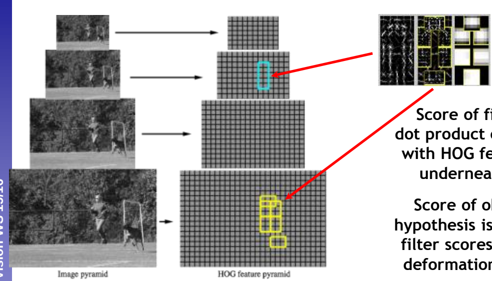
Recap: Deformable Part-Based Model



Root filters coarse resolution | Part filters finer resolution | Deformation models

Slide credit: Pedro Felzenszwalb

Recap: Object Hypothesis



Score of filter: dot product of filter with HOG features underneath it

Score of object hypothesis is sum of filter scores minus deformation costs

- Multiscale model captures features at two resolutions

Slide credit: Pedro Felzenszwalb

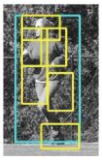
Recap: Score of a Hypothesis

$$\text{score}(p_0,\ldots,p_n) = \underbrace{\sum_{i=0}^{n} F_i \cdot \phi(H,p_i)}_{\text{“data term”}} - \underbrace{\sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)}_{\text{“spatial prior”}}$$

filters

displacements

deformation parameters



$$\text{score}(z) = \beta \cdot \Psi(H,z)$$

concatenation filters and deformation parameters

concatenation of HOG features and part displacement features

Slide credit: Pedro Felzenszwalb

Topics of This Lecture

- Deep Learning
  - Motivation
- Convolutional Neural Networks
  - Convolutional Layers
  - Pooling Layers
  - Nonlinearities
- CNN Architectures
  - LeNet
  - AlexNet
  - VGGNet
  - GoogLeNet
- Applications

We’ve finally got there!

Traditional Recognition Approach

Image/ Video Pixels ⇨ Hand-designed feature extraction ⇨ Trainable classifier ⇨ Object Class

- Characteristics
  - Features are not learned, but engineered
  - Trainable classifier is often generic (e.g., SVM)

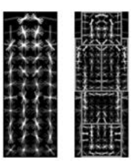
  ⇒ Many successes in 2000-2010.

Slide credit: Svetlana Lazebnik

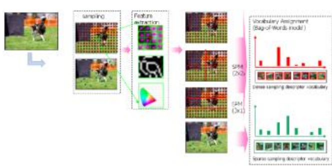
Traditional Recognition Approach

- Features are key to recent progress in recognition
  - Multitude of hand-designed features currently in use
  - SIFT, HOG, ...

  ⇒ *Where next? Better classifiers? Or keep building more features?*



DPM [Felzenszwalb et al., PAMI’07]



Dense SIFT+LBP+HOG → BOW → Classifier [Yan & Huan ‘10] (Winner of PASCAL 2010 Challenge)

Slide credit: Svetlana Lazebnik

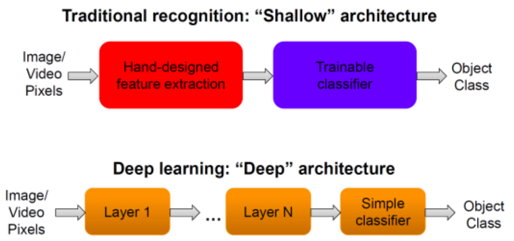
What About Learning the Features?

- Learn a *feature hierarchy* all the way from pixels to classifier
  - Each layer extracts features from the output of previous layer
  - Train all layers jointly

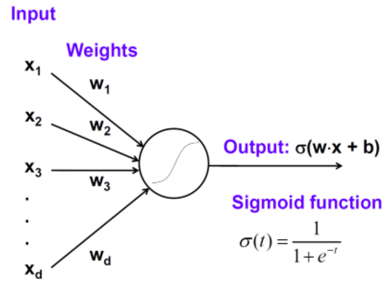
Image/ Video Pixels ⇨ Layer 1 ⇨ Layer 2 ⇨ Layer 3 ⇨ Simple Classifier

Slide credit: Svetlana Lazebnik

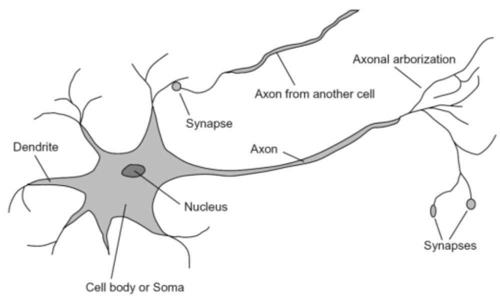
"Shallow" vs. "Deep" Architectures



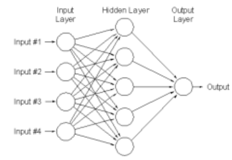
Background: Perceptrons



Inspiration: Neuron Cells



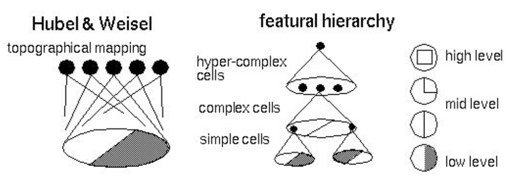
Background: Multi-Layer Neural Networks



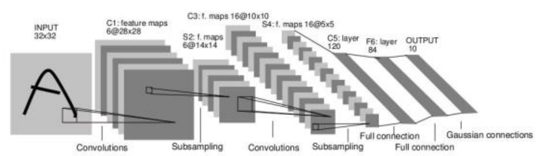
- Nonlinear classifier
  - Training: find network weights $\mathbf{w}$ to minimize the error between true training labels t_n and estimated labels $f_{\mathbf{w}}(x_n)$:
    $$E(\mathbf{W}) = \sum_n L(t_n, f(\mathbf{x}_n; \mathbf{W}))$$
  - Minimization can be done by gradient descent, provided f is differentiable
    - Training method: error backpropagation.

Hubel/Wiesel Architecture

- D. Hubel, T. Wiesel (1959, 1962, Nobel Prize 1981)
  - Visual cortex consists of a hierarchy of *simple*, *complex*, and *hyper-complex* cells



Convolutional Neural Networks (CNN, ConvNet)



- Neural network with specialized connectivity structure
  - Stack multiple stages of feature extractors
  - Higher stages compute more global, more invariant features
  - Classification layer at the end

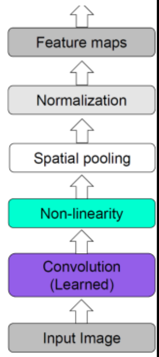
Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11): 2278-2324, 1998.

Topics of This Lecture

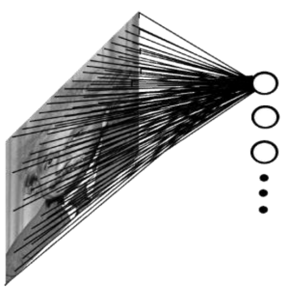
- Deep Learning
  - Motivation
- Convolutional Neural Networks
  - Convolutional Layers
  - Pooling Layers
  - Nonlinearities
- CNN Architectures
  - LeNet
  - AlexNet
  - VGGNet
  - GoogLeNet
- Applications

Convolutional Networks: Structure

- Feed-forward feature extraction
  1. Convolve input with learned filters
  2. Non-linearity
  3. Spatial pooling
  4. (Normalization)
- Supervised training of convolutional filters by back-propagating classification error

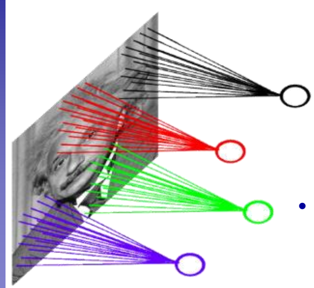


Convolutional Networks: Intuition



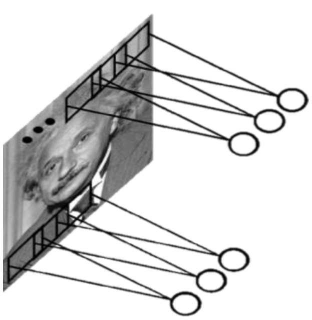
- Fully connected network
  - E.g. 1000×1000 image
    1M hidden units
  - ⇒ 1T parameters!
- Ideas to improve this
  - Spatial correlation is local

Convolutional Networks: Intuition



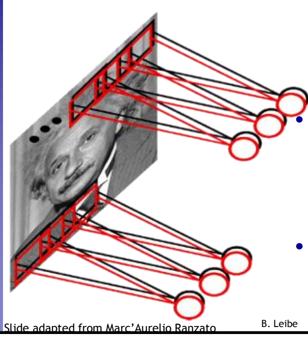
- Locally connected net
  - E.g. 1000×1000 image
    1M hidden units
    10×10 receptive fields
  - ⇒ 100M parameters!
- Ideas to improve this
  - Spatial correlation is local
  - Want translation invariance

Convolutional Networks: Intuition



- Convolutional net
  - Share the same parameters across different locations
  - Convolutions with learned kernels

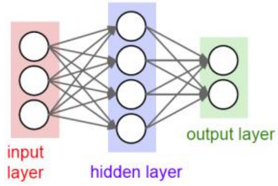
Convolutional Networks: Intuition



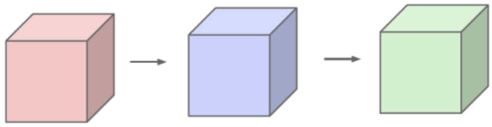
- Convolutional net
  - Share the same parameters across different locations
  - Convolutions with learned kernels
- Learn *multiple* filters
  - E.g. 1000×1000 image
    100 filters
    10×10 filter size
  - ⇒ 10k parameters
- Result: Response map
  - size: 1000×1000×100
  - Only memory, not params!

Important Conceptual Shift

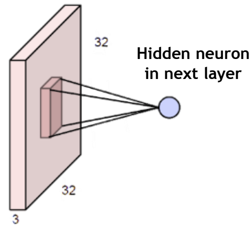
- Before



- Now:



Convolution Layers



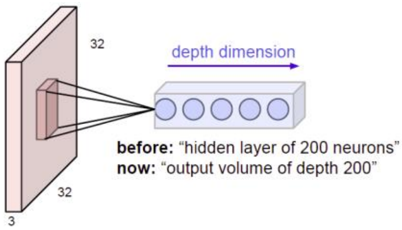
Example
image: 32×32×3 volume

Before: Full connectivity
32×32×3 weights

Now: Local connectivity
One neuron connects to, e.g., 5×5×3 region.
⇒ Only 5×5×3 shared weights.

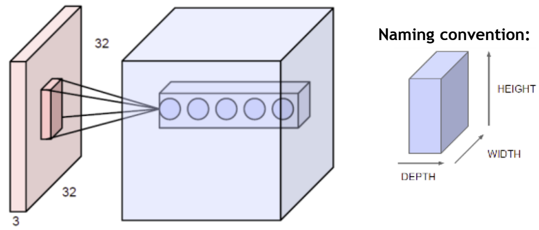
- Note: Connectivity is
  - Local in space (5×5 inside 32×32)
  - But full in depth (all 3 depth channels)

Convolution Layers



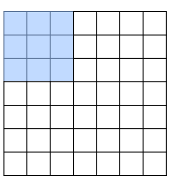
- All Neural Net activations arranged in 3 dimensions
  - Multiple neurons all looking at the same input region, stacked in depth

Convolution Layers



- All Neural Net activations arranged in 3 dimensions
  - Multiple neurons all looking at the same input region, stacked in depth
  - Form a single [1×1×depth] depth column in output volume.

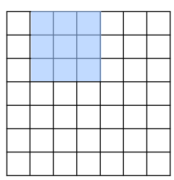
Convolution Layers



Example:
7×7 input
assume 3×3 connectivity
stride 1

- Replicate this column of hidden neurons across space, with some stride.

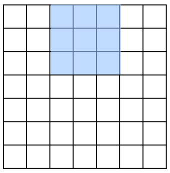
Convolution Layers



Example:
7×7 input
assume 3×3 connectivity
stride 1

- Replicate this column of hidden neurons across space, with some stride.

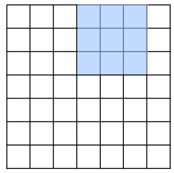
Convolution Layers



Example:
7×7 input
assume 3×3 connectivity
stride 1

- Replicate this column of hidden neurons across space, with some stride.

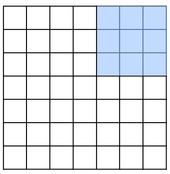
Convolution Layers



Example:
7×7 input
assume 3×3 connectivity
stride 1

- Replicate this column of hidden neurons across space, with some stride.

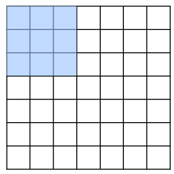
Convolution Layers



Example:
7×7 input
assume 3×3 connectivity
stride 1
$\Rightarrow 5 \times 5$ output

- Replicate this column of hidden neurons across space, with some stride.

Convolution Layers

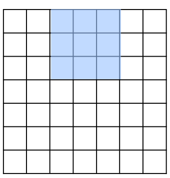


Example:
7×7 input
assume 3×3 connectivity
stride 1
$\Rightarrow 5 \times 5$ output

What about stride 2?

- Replicate this column of hidden neurons across space, with some stride.

Convolution Layers

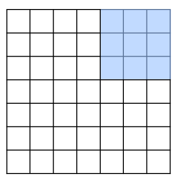


Example:
7×7 input
assume 3×3 connectivity
stride 1
$\Rightarrow 5 \times 5$ output

What about stride 2?

- Replicate this column of hidden neurons across space, with some stride.

Convolution Layers

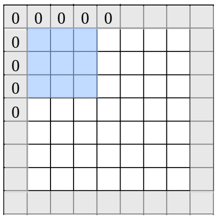


Example:
7×7 input
assume 3×3 connectivity
stride 1
$\Rightarrow 5 \times 5$ output

What about stride 2?
$\Rightarrow 3 \times 3$ output

- Replicate this column of hidden neurons across space, with some stride.

Convolution Layers



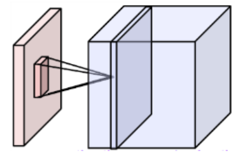
Example:
7×7 input
assume 3×3 connectivity
stride 1
$\Rightarrow 5\times5$ output

What about stride 2?
$\Rightarrow 3\times3$ output

- Replicate this column of hidden neurons across space, with some stride.
- In practice, common to zero-pad the border.
  - Preserves the size of the input spatially.

Activation Maps of Convolutional Filters

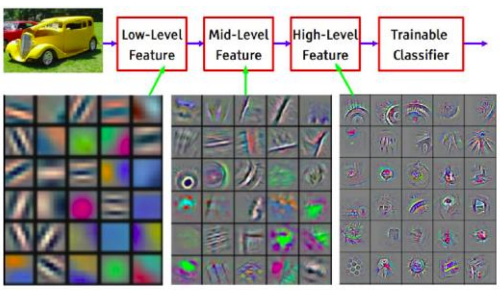
5×5 filters



Activation maps

Each activation map is a depth slice through the output volume.

Effect of Multiple Convolution Layers



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

Commonly Used Nonlinearities

- Sigmoid

$$g(a) = \sigma(a) = \frac{1}{1+\exp\{-a\}}$$

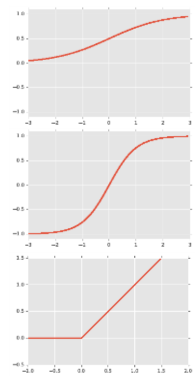
- Hyperbolic tangent

$$g(a) = tanh(a) = 2\sigma(2a) - 1$$

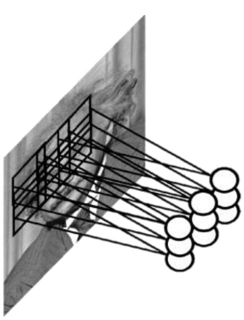
- Rectified linear unit (ReLU)

$$g(a) = \max\{0, a\}$$

Preferred option for deep networks

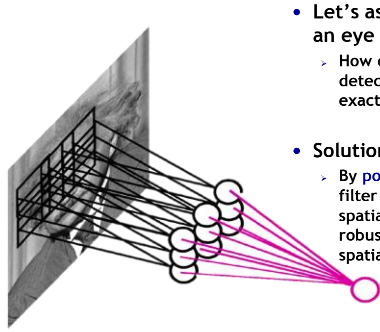


Convolutional Networks: Intuition



- Let's assume the filter is an eye detector
  - How can we make the detection robust to the exact location of the eye?

Convolutional Networks: Intuition



- Let's assume the filter is an eye detector
  - How can we make the detection robust to the exact location of the eye?
- Solution:
  - By pooling (e.g., max or avg) filter responses at different spatial locations, we gain robustness to the exact spatial location of features.

Max Pooling

Single depth slice

1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

x, y

max pool with 2x2 filters and stride 2

6	8
3	4

- **Effect:**
  - **Make the representation smaller without losing too much information**
  - **Achieve robustness to translations**

Max Pooling

Single depth slice

1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

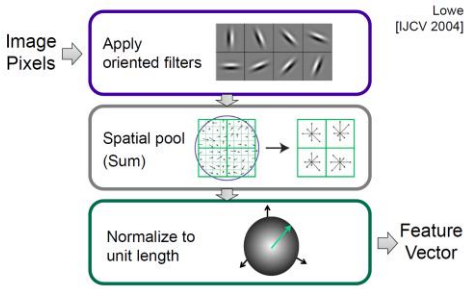
x, y

max pool with 2x2 filters and stride 2

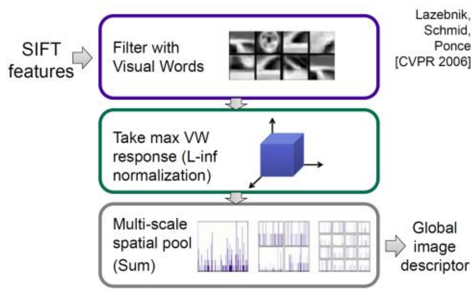
6	8
3	4

- **Note**
  - **Pooling happens independently across each slice, preserving the number of slices.**

Compare: SIFT Descriptor



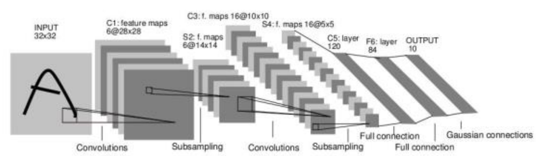
Compare: Spatial Pyramid Matching



Topics of This Lecture

- Deep Learning
  - Motivation
- Convolutional Neural Networks
  - Convolutional Layers
  - Pooling Layers
  - Nonlinearities
- **CNN Architectures**
  - **LeNet**
  - **AlexNet**
  - **VGGNet**
  - **GoogLeNet**
- Applications

CNN Architectures: LeNet (1998)



- Early convolutional architecture
  - 2 Convolutional layers, 2 pooling layers
  - Fully-connected NN layers for classification
  - Successfully used for handwritten digit recognition (MNIST)

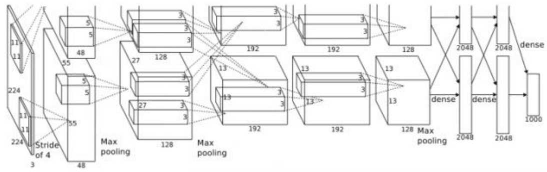
Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11): 2278-2324, 1998.

ImageNet Challenge 2012

- ImageNet
  - ~14M labeled internet images
  - 20k classes
  - Human labels via Amazon Mechanical Turk
- Challenge (ILSVRC)
  - 1.2 million training images
  - 1000 classes
  - Goal: Predict ground-truth class within top-5 responses
  - Currently one of the top benchmarks in Computer Vision

[Deng et al., CVPR'09]

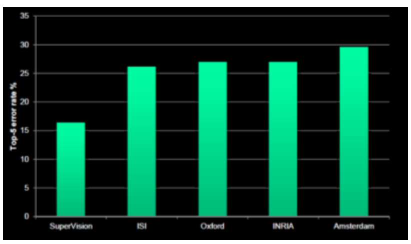
CNN Architectures: AlexNet (2012)



- Similar framework as LeNet, but
  - Bigger model (7 hidden layers, 650k units, 60M parameters)
  - More data (10^6 images instead of 10^3)
  - GPU implementation
  - Better regularization and up-to-date tricks for training (Dropout)

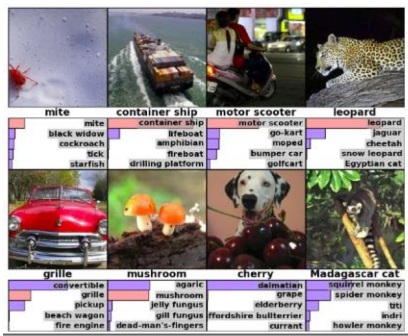
A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012.

ILSVRC 2012 Results

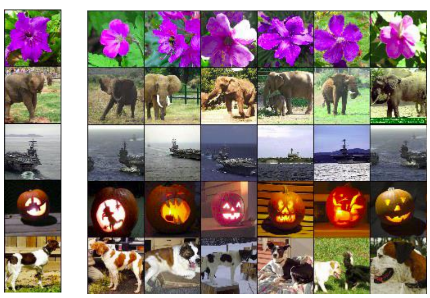


- AlexNet almost halved the error rate
  - 16.4% error (top-5) vs. 26.2% for the next best approach
  - ⇒ A revolution in Computer Vision
  - Acquired by Google in Jan '13, deployed in Google+ in May '13

AlexNet Results

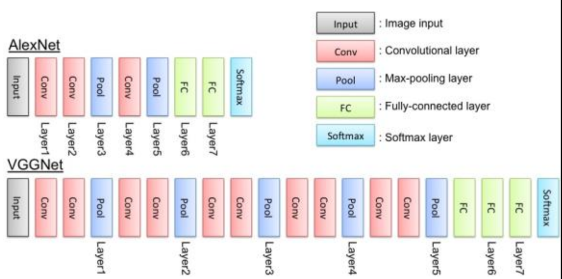


AlexNet Results



Test image Retrieved images

CNN Architectures: VGGNet (2014/15)



K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015

CNN Architectures: VGGNet (2014/15)

- **Main ideas**
  - Deeper network
  - Stacked convolutional layers with smaller filters (+ nonlinearity)
  - Detailed evaluation of all components
- **Results**
  - Improved ILSVRC top-5 error rate to 6.7%.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

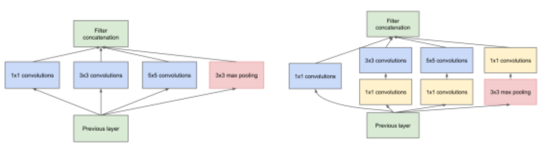
Mainly used

Image source: Simonyan & Zisserman

Comparison: AlexNet vs. VGGNet

- **Receptive fields in the first layer**
  - AlexNet: 11×11, stride 4
  - Zeiler & Fergus: 7×7, stride 2
  - VGGNet: 3×3, stride 1
- **Why that?**
  - If you stack three 3×3 on top of another 3×3 layer, you effectively get a 5×5 receptive field.
  - With three 3×3 layers, the receptive field is already 7×7.
  - But much fewer parameters: $3 \cdot 3^2 = 27$ instead of $7^2 = 49$.
  - In addition, non-linearities in-between 3×3 layers for additional discriminativity.

CNN Architectures: GoogLeNet (2014)



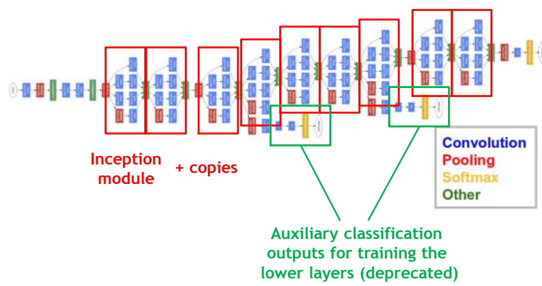
(a) Inception module, naïve version

(b) Inception module with dimension reductions

- **Main ideas**
  - "Inception" module as modular component
  - Learns filters at several scales within each module

C. Szegedy, W. Liu, Y. Jia, et al, Going Deeper with Convolutions, arXiv:1409.4842, 2014.

GoogLeNet Visualization



Results on ILSVRC

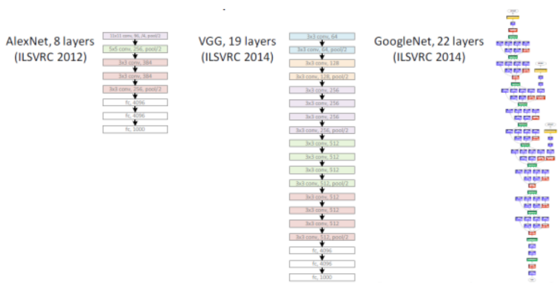
Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	**23.7**	**6.8**	**6.8**
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-	7.9	
GoogLeNet (Szegedy et al., 2014) (7 nets)	-	**6.7**	
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

- **VGGNet and GoogLeNet perform at similar level**
  - Comparison: human performance ~5% [Karpathy]

http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/

Image source: Simonyan & Zisserman

Newest Development: Residual Networks



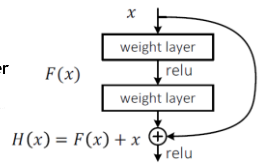
Newest Development: Residual Networks

AlexNet, 8 layers (ILSVRC 2012)

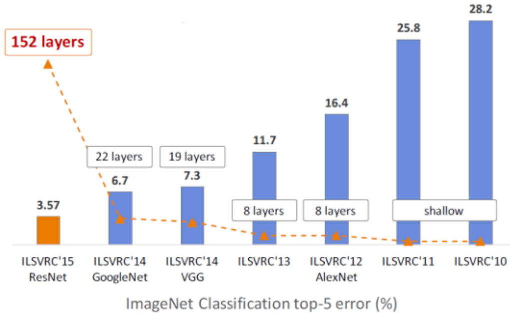
VGG, 19 layers (ILSVRC 2014)

ResNet, 152 layers (ILSVRC 2015)

- Core component
  - Skip connections bypassing each layer
  - Better propagation of gradients to the deeper layers



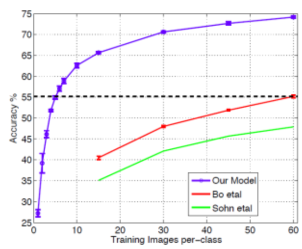
ImageNet Performance



Topics of This Lecture

- Deep Learning
  - Motivation
- Convolutional Neural Networks
  - Convolutional Layers
  - Pooling Layers
  - Nonlinearities
- CNN Architectures
  - LeNet
  - AlexNet
  - VGGNet
  - GoogLeNet
- Applications

The Learned Features are Generic



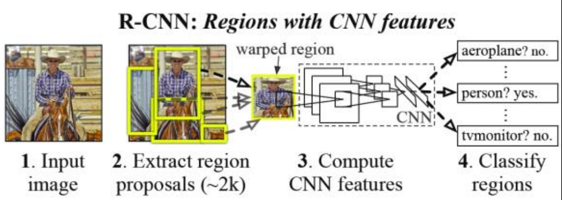
state of the art level (pre-CNN)

- Experiment: feature transfer
  - Train network on ImageNet
  - Chop off last layer and train classification layer on CalTech256

⇒ State of the art accuracy already with only 6 training images

Image source: M. Zeiler, R. Fergus

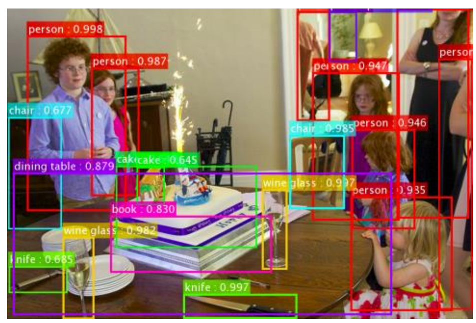
Other Tasks: Detection



- Results on PASCAL VOC Detection benchmark
  - Pre-CNN state of the art: 35.1% mAP [Uijlings et al., 2013]
    33.4% mAP DPM
  - R-CNN: 53.7% mAP

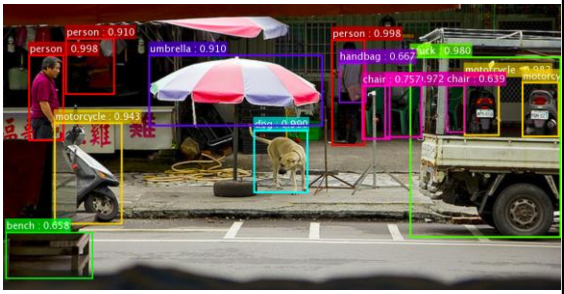
R. Girshick, J. Donahue, T. Darrell, and J. Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, CVPR 2014

Faster R-CNN (based on ResNets)



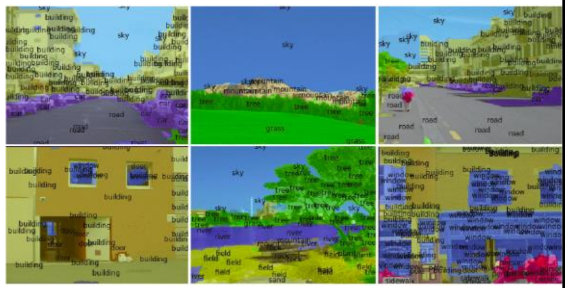
K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, CVPR 2016.

Faster R-CNN (based on ResNets)



K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, CVPR 2016.

Other Tasks: Semantic Segmentation



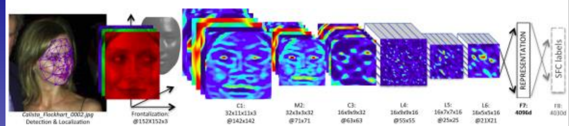
[Farabet et al. ICML 2012, PAMI 2013]

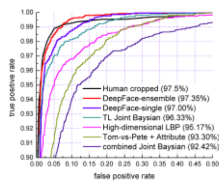
Semantic Segmentation

[Pohlen, Hermans, Mathias, Leibe, arXiv 2016]

- More recent results
  - Based on an extension of ResNets

Other Tasks: Face Verification



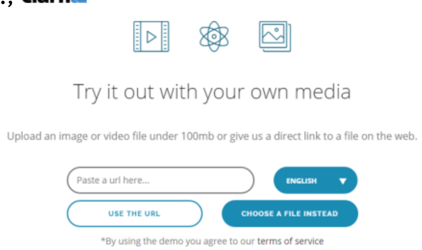


Y. Taigman, M. Yang, M. Ranzato, L. Wolf, DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CVPR 2014

Slide credit: Svetlana Lazebnik

Commercial Recognition Services

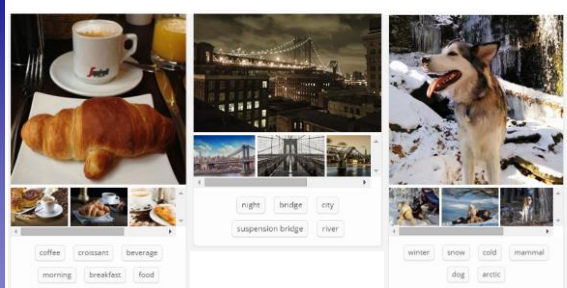
- E.g., clarifai



- Be careful when taking test images from Google Search
  - Chances are they may have been seen in the training set...

Image source: clarifai.com

Commercial Recognition Services



clarifai

Image source: clarifai.com

References and Further Reading

- **LeNet**
  - Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11): 2278–2324, 1998.
- **AlexNet**
  - A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012.
- **VGGNet**
  - K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015
- **GoogLeNet**
  - C. Szegedy, W. Liu, Y. Jia, et al, Going Deeper with Convolutions, arXiv:1409.4842, 2014.