Nuts and Bolts of Computer Vision

CSCI 8360

LECTURE 7

What is the goal of classification?

- Learn a function f
- Maps: continuous space (data) to discrete space (label)

$$f(X) \rightarrow Y$$

What is needed for classification?

- Objective function — Decision tree, Logistic Regression, Naïve Bayes, etc
- Labeled training data — $$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$$
- ***Features*** — ???

Feature engineering

Documents

- Bag-of-words
- Term frequencies
- TF-IDF
- word2vec

Time series

- Fourier coefficients
- Cepstrum
- KL-divergence

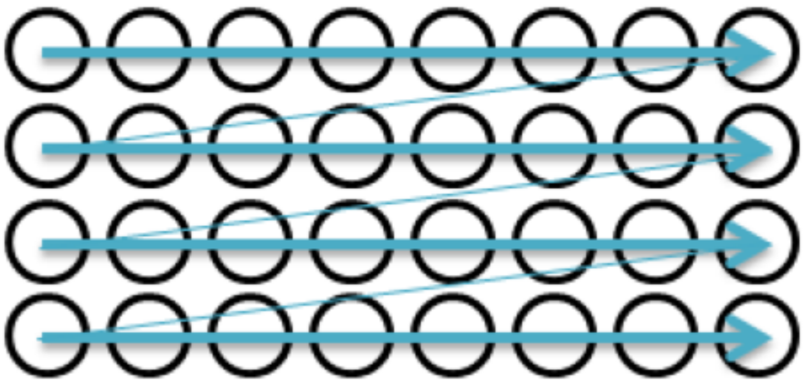
Recommendation

- SVD
- Alternating Least Squares
- Dimensionality reduction

What about images?

Level 0: Pixels

- Flatten image into a feature vector
  - Raster scan
- Only works at all if ALL images have the same dimensions
- Rarely works regardless
  - Natural images
  - Same object, different viewpoints
  - Image registration
  - Etc.



Level 1: Pixel features

- Normalization
- Equalization
- Filters

Normalization

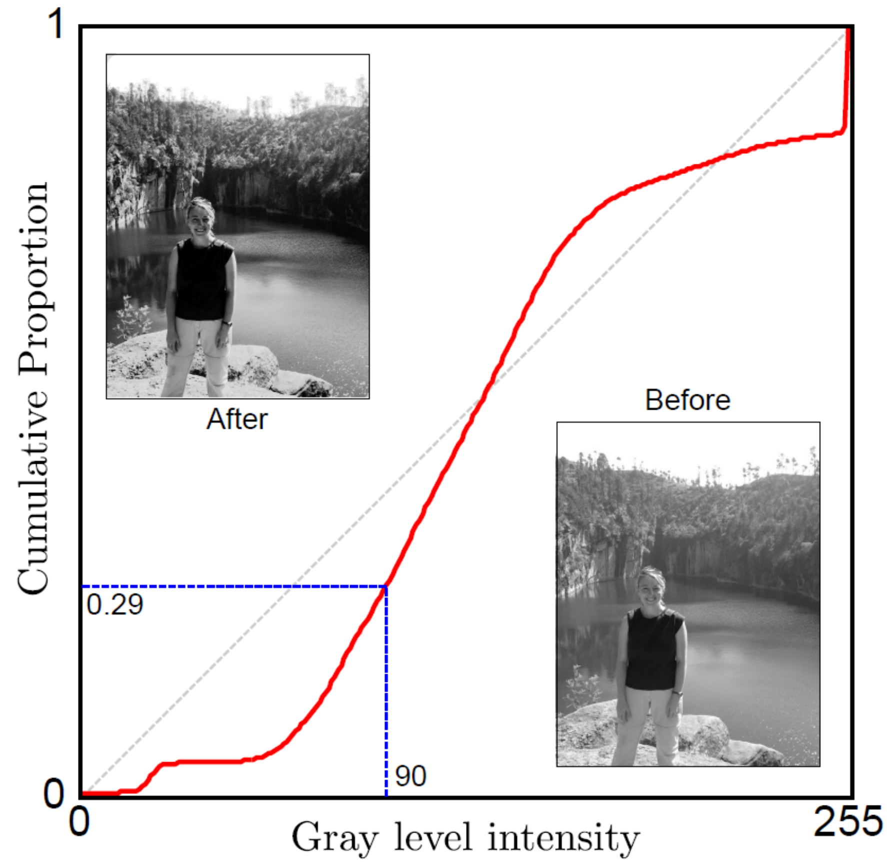
- Normalize 1^{st} and 2^{nd} moments
- Removes contrast and additive luminescence variations

Before

After

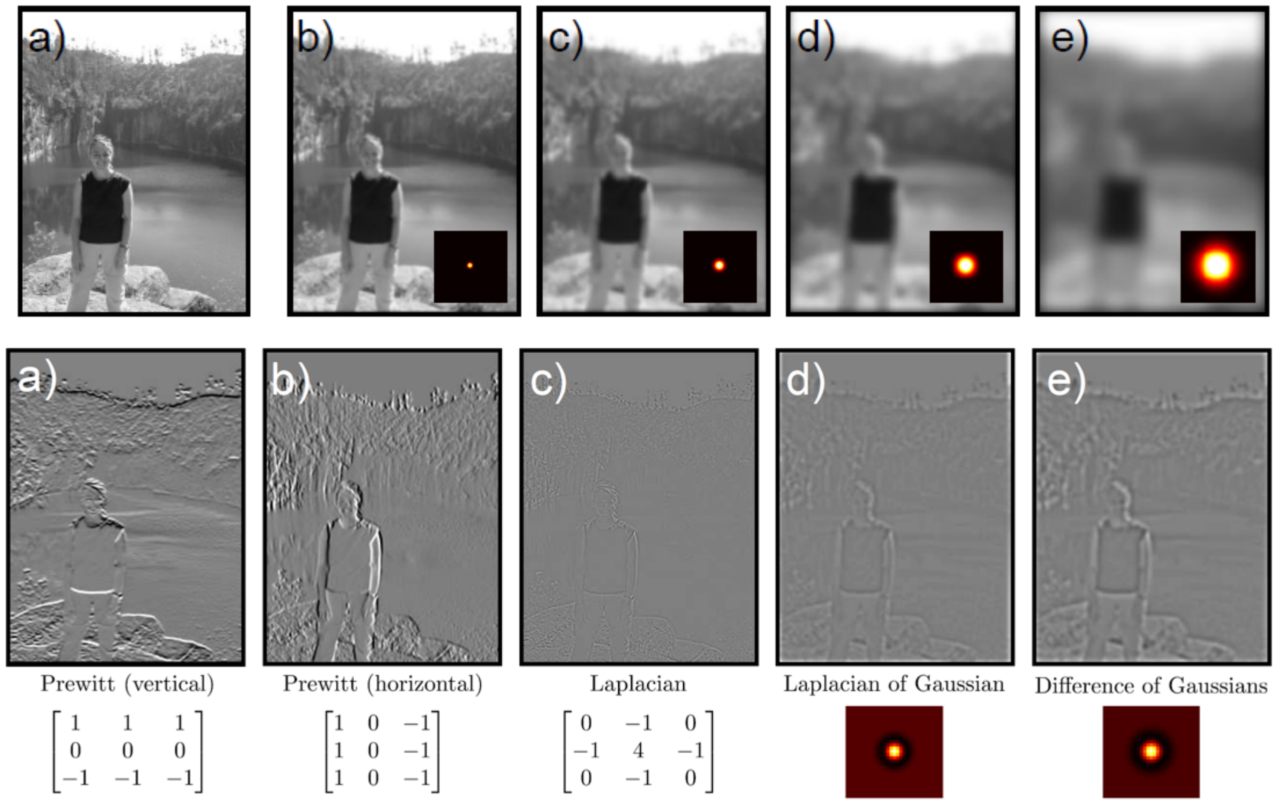
Histogram Equalization

- Consider luminescence as a cumulative distribution function (CDF)
- Normalize the CDF to be linear
- Enhances global contrast (potentially by magnifying noise)



Filters

- Blurring
- Image gradients



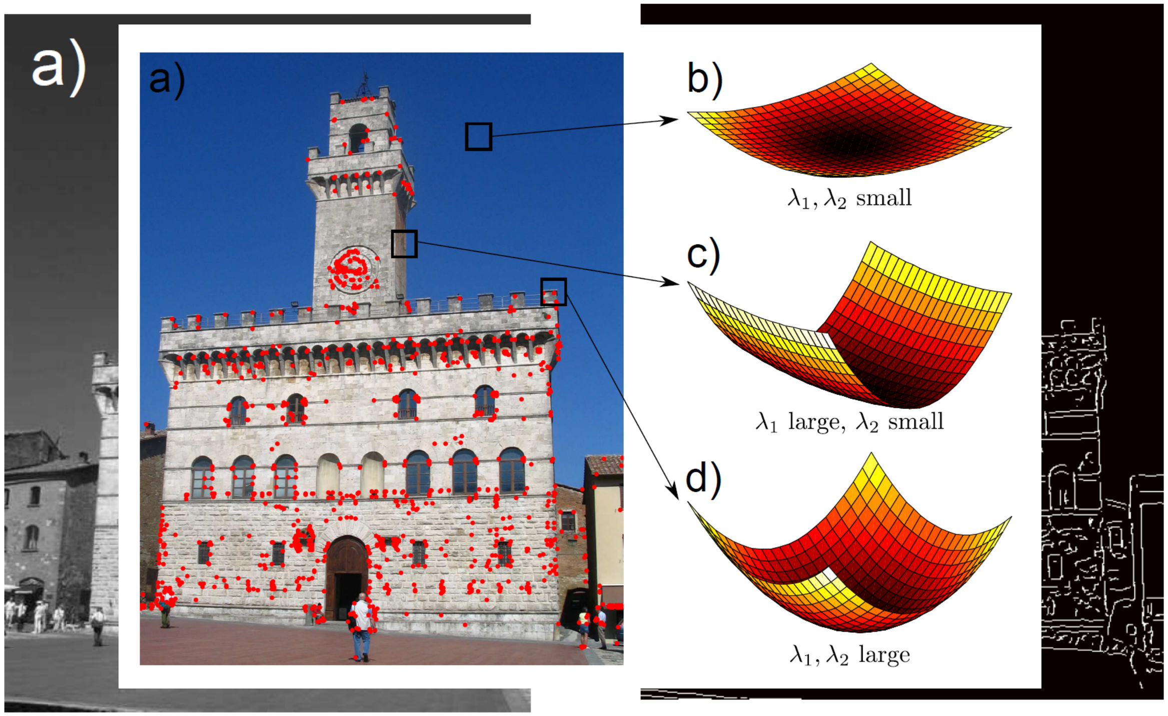
$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Level 2A: Pixel functions

- Edges and corners
- SIFT & SURF
- Textures

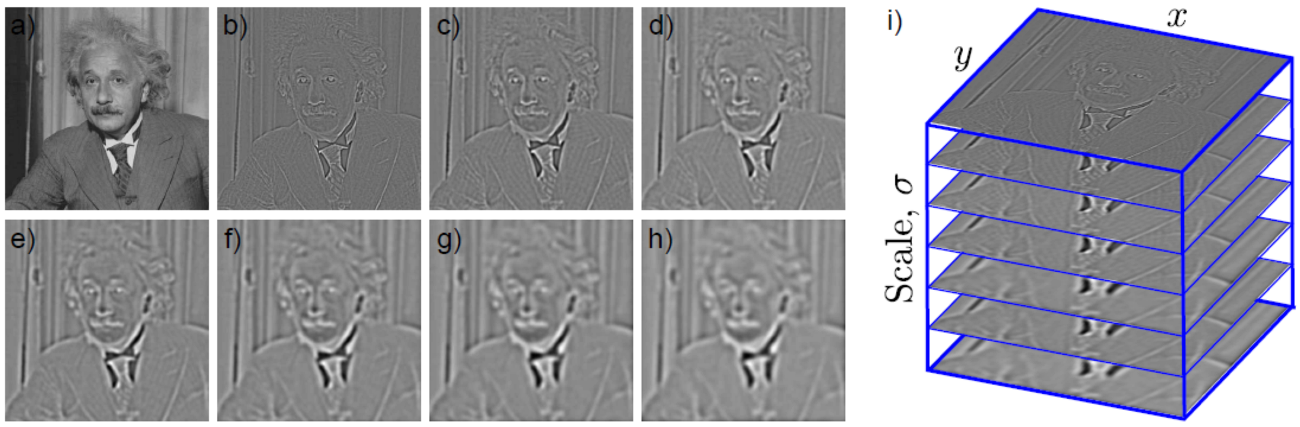
Edges and Corners

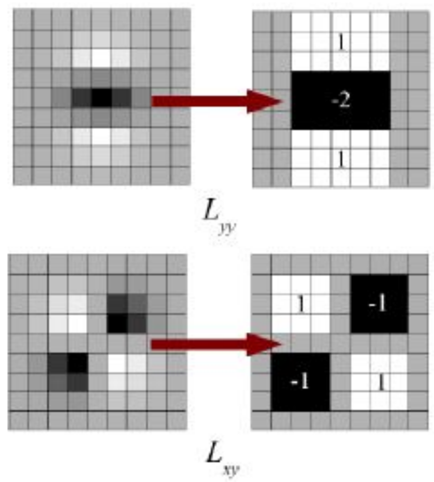
- Canny Edge Detector
  - Combination of horizontal and vertical image gradients
- Harris Corner Detector



SIFT and SURF

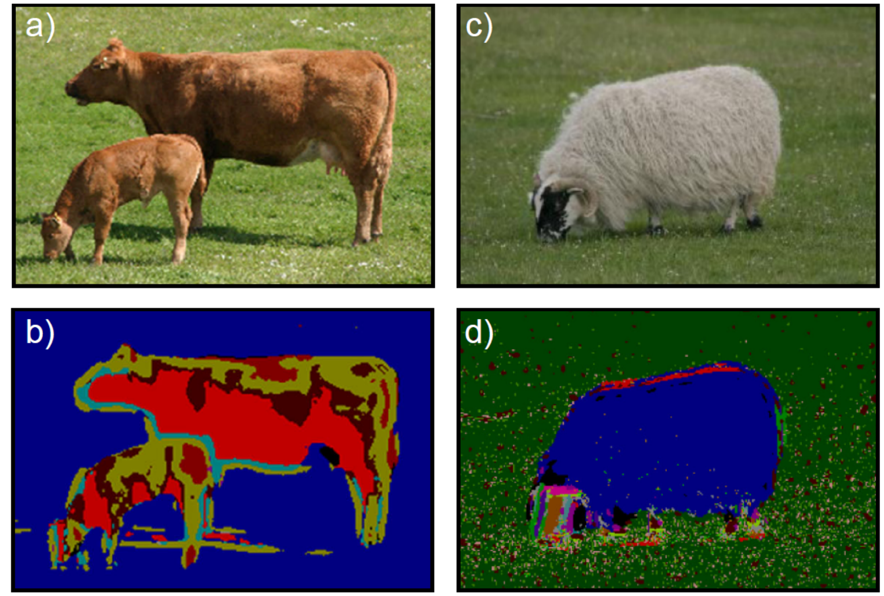
- SIFT (Scale-Invariant Feature Transform)
- SURF (Speeded-Up Robust Features)



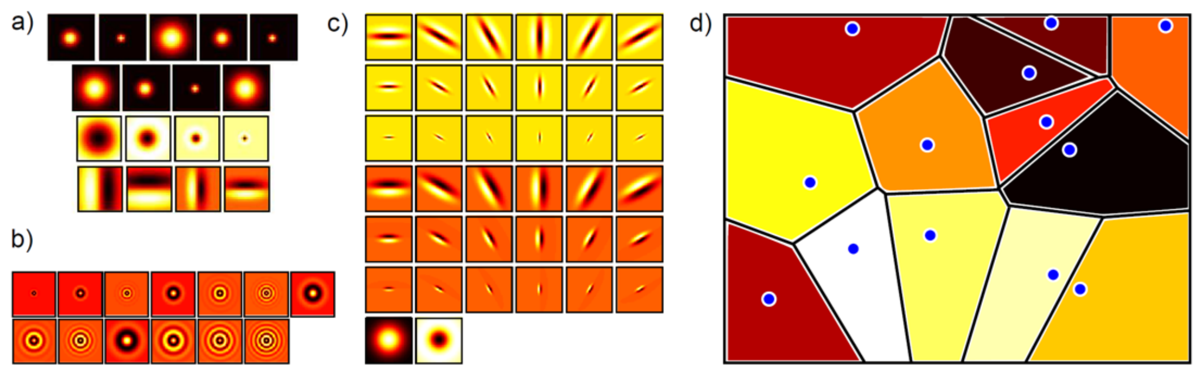


Textures

- Textons
  - Replace each pixel with integer representing "pixel type"

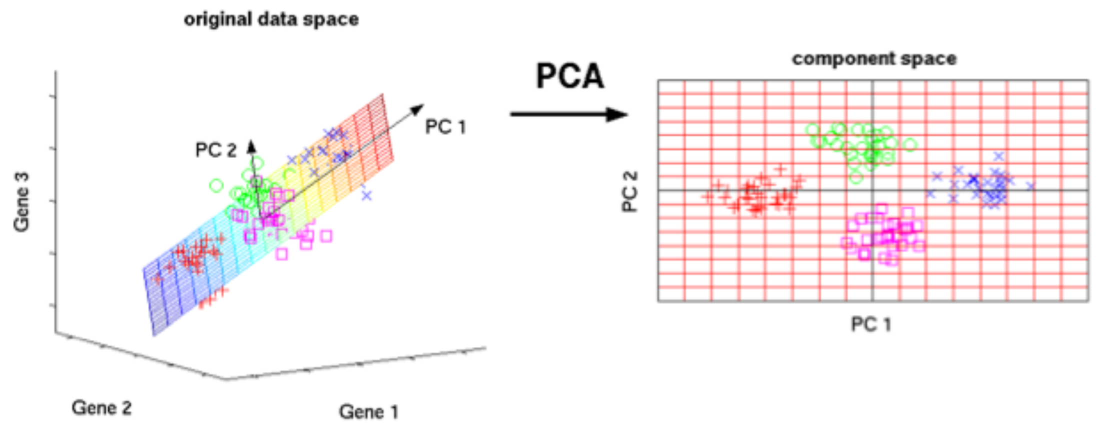


- Take a bank of filters (gaussian, laplacian, etc) and apply to images
- Cluster pixels in filter space
- For new pixel, filter surrounding region with filter bank, and assign to nearest cluster



Level 2B: Dimensionality Reduction

- Images are extremely high-dimensional
- …but you're probably only ever interested in a few things in the images
- Most of the image [data] is therefore extraneous, so our goal is to **reduce the dimensions**
  - Principal Components Analysis (PCA)
  - Singular Value Decomposition (SVD)
  - Non-negative Matrix Factorization (NMF)



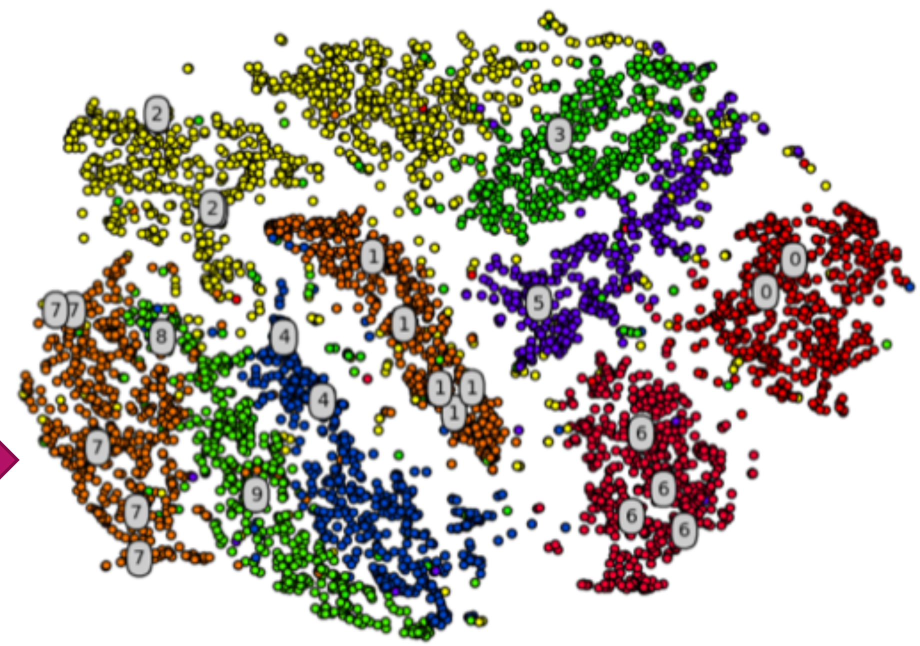
Embeddings

$$f : X \to Y$$

- What is an *embedding*?

- **Mapping**
- **Transformation**
- **Reveals / preserves “structure”**

Embedding



Embeddings

- “Degrees of freedom” versus “intrinsic dimensionality”

- Despite 64x64 pixels, only so many ways to draw a 9
- **Use projections in low-dimensional manifold as features**

Level 3: Hierarchical pixel functions

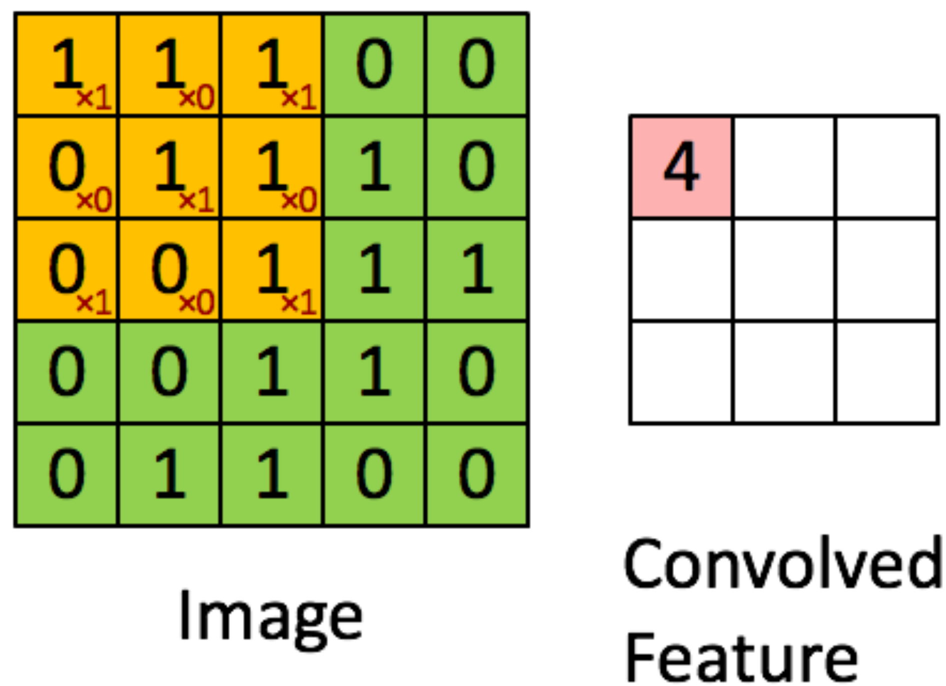
- Convolutional neural networks (CNNs) and Deep Learning

http://deeplearning4j.org/convolutionalnets

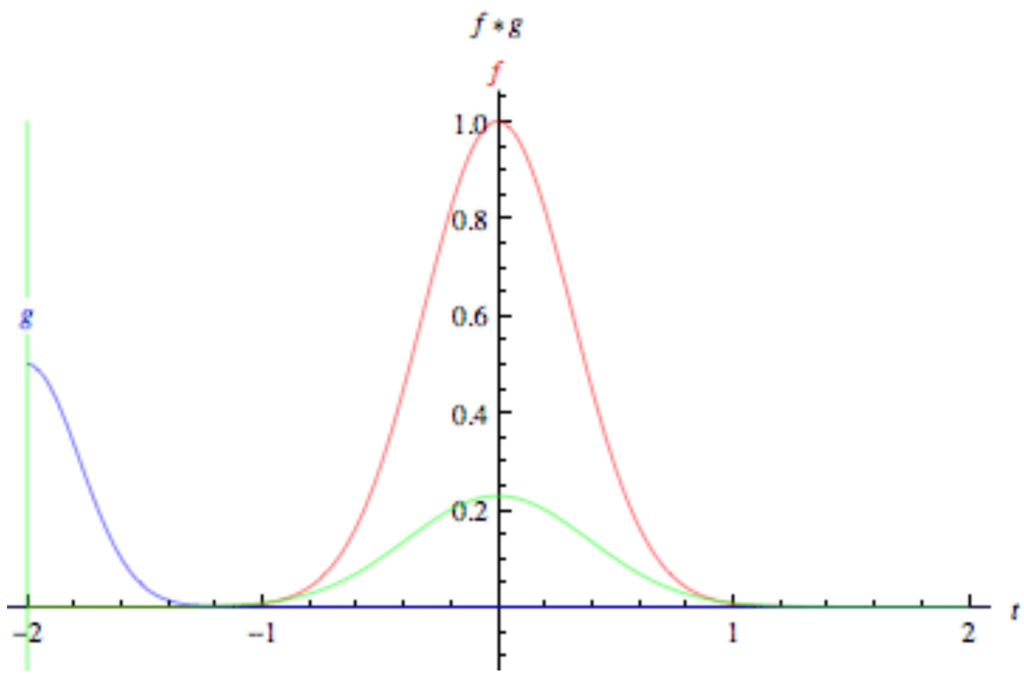
What is a “convolution”?

- Two matrices:
  - Image (of course)
  - Filter (small matrix)
- Run filter over image, taking dot products

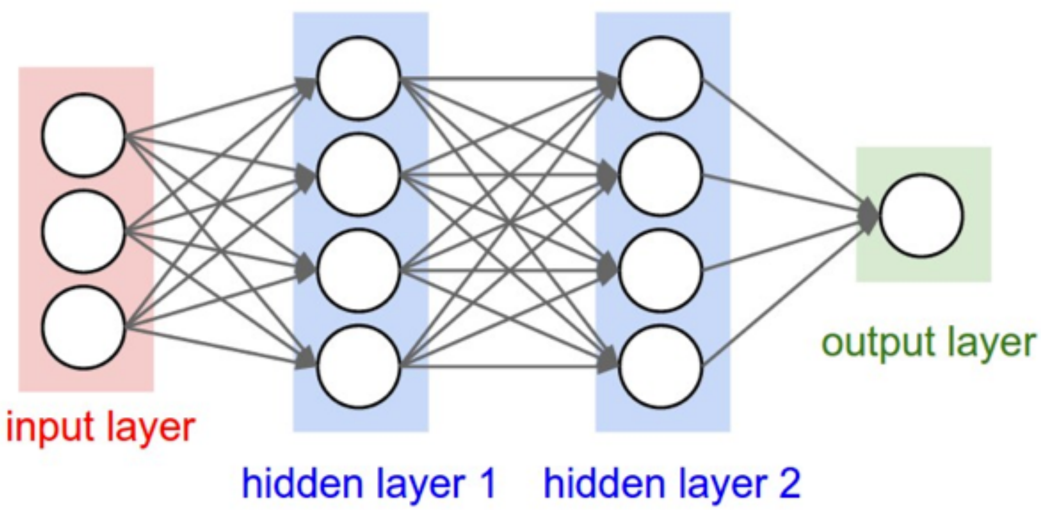
$$\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

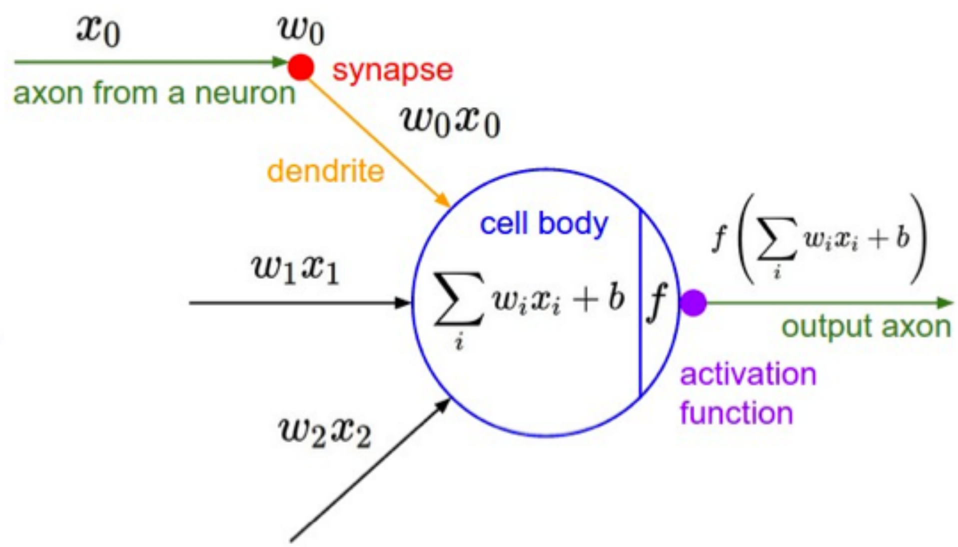


What is a “convolution”?

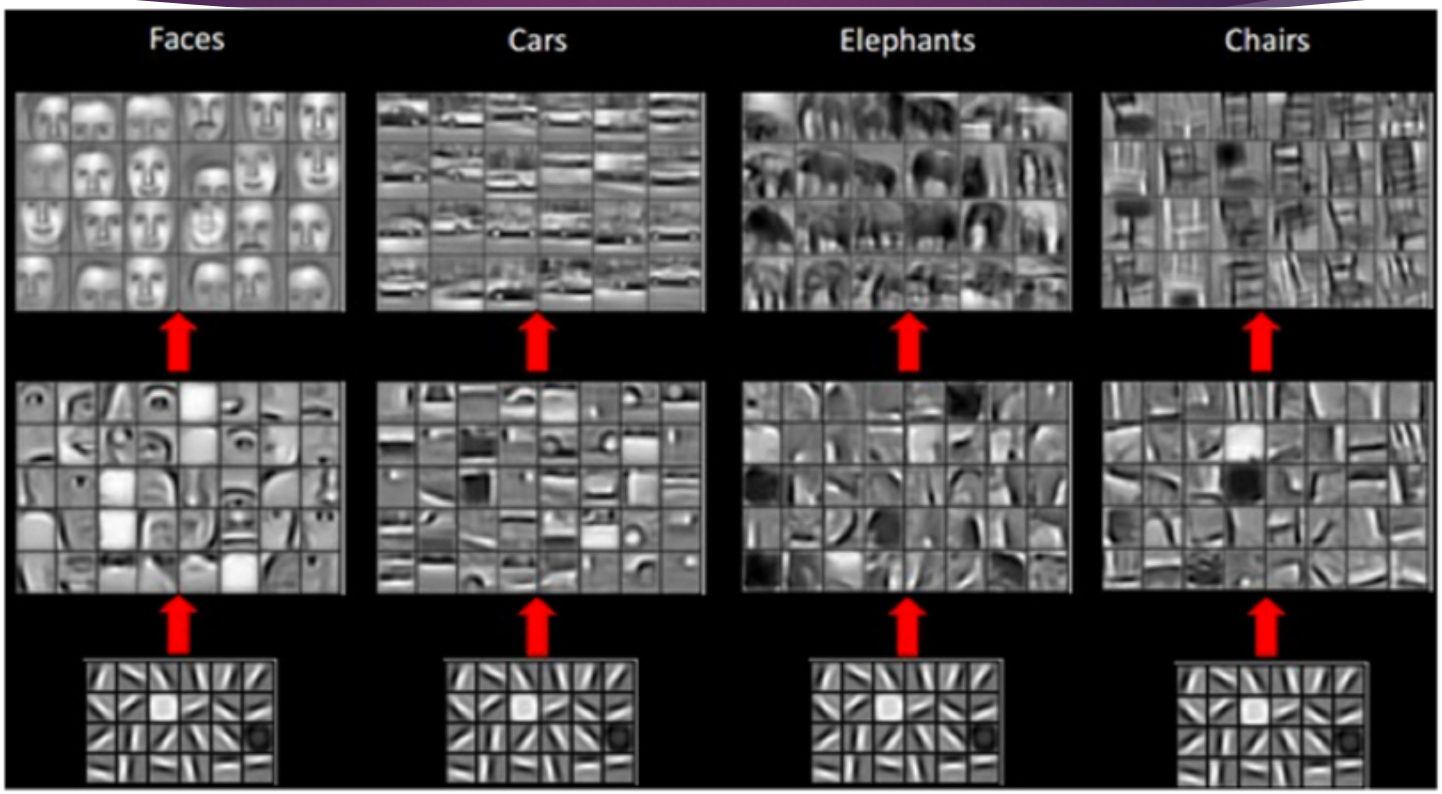


What is a “neural network”?

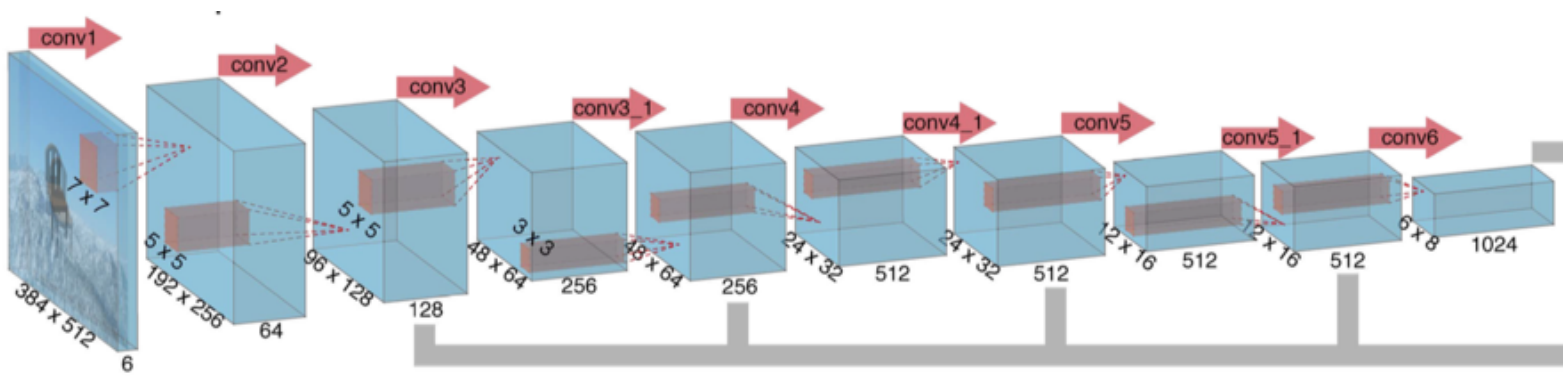




Stacked feed-forward networks

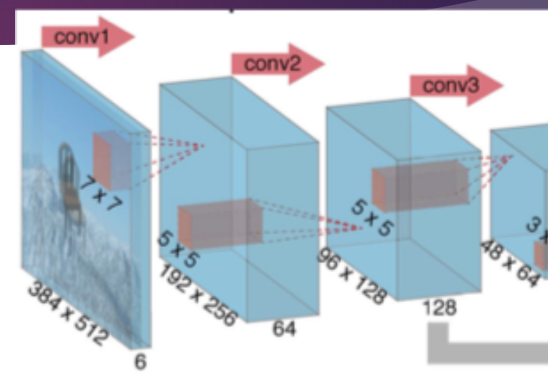


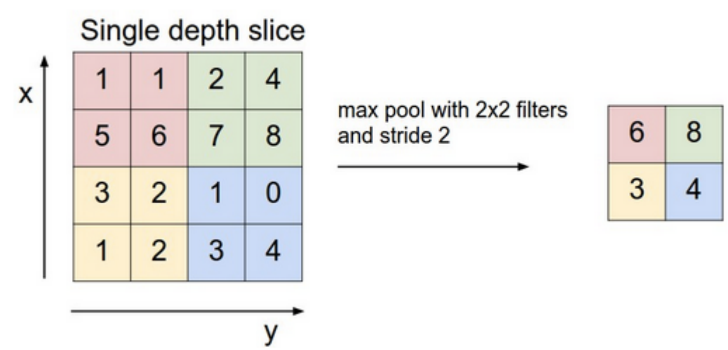
Stacked feed-forward networks



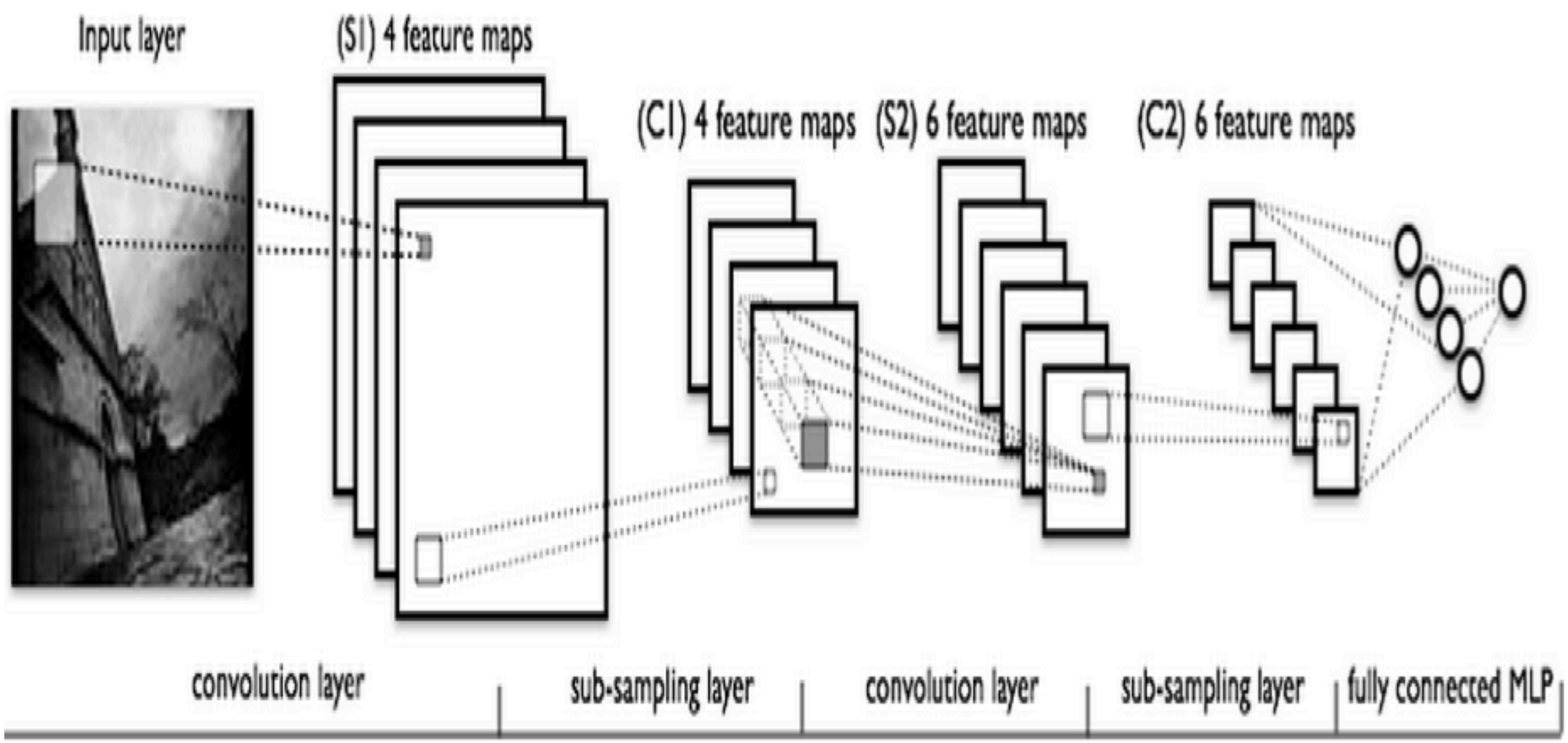
Terminology

- **Feature maps**
  - The output of one layer of a CNN
- **Stride**
  - Size (in pixels) of the move a filter makes over neighborhoods of pixels
- **Pooling**
  - "Summary" of the output of a filter
  - Popular choices: mean, max
- **Activation functions**
  - Function determining whether or not an individual neuron fires
  - Popular choices: ReLU, tanh, sigmoid



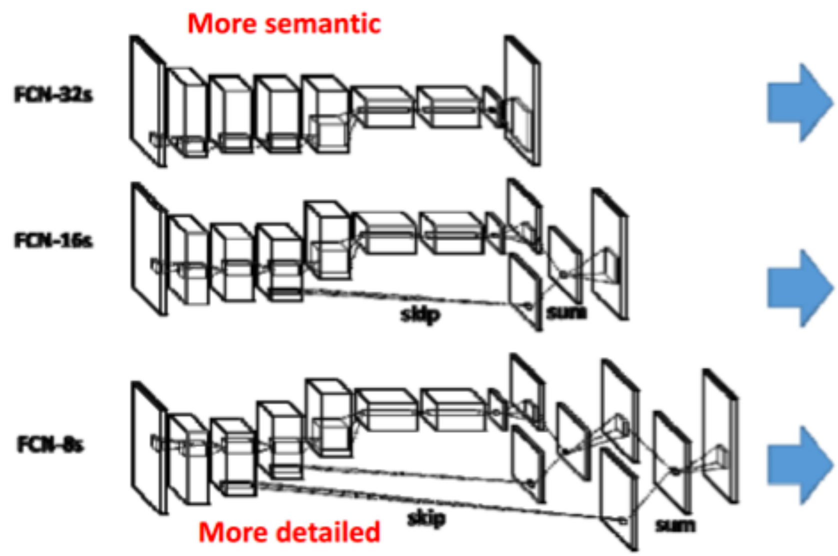


CNN Pipeline



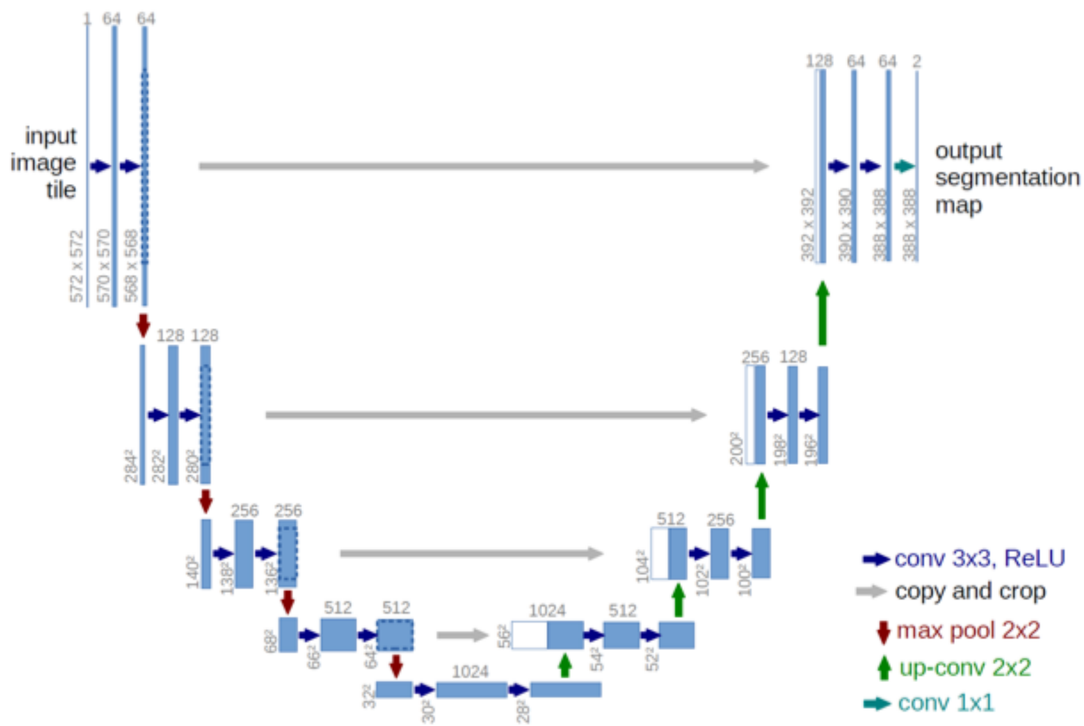
Semantic Segmentation

- Create a map of the detected object areas
- "Fully-convolutional" networks
  - Substitute fully-connected layer at end for another convolutional layer
  - Activations show object
- Resolution is lost in upsampling step
  - Skip-connections to bring in some of the "lost" resolution
- *EXTREME* Segmentation
  - Replace upsampling with a complete deconvolution stack



Semantic Segmentation

- In effect, each pixel is "classified"
- Uses ground-truth segmentation maps in conjunction with convolutional feature maps
- U-Net
  - So named because of the "U" structure of convolutions + interpolations
  - Skip-connections at each level incorporate image features to help refine up-sampling



References

- Prince, Simon JD. *Computer vision: models, learning, and inference*. Cambridge University Press, 2012. Chapter 13: Image Processing and Feature Extraction.
- Coelho, Luis Pedro, and Willi Richert. *Building machine learning systems with Python*. Packt Publishing Ltd, 2015. Chapter 10: Computer Vision – Pattern Recognition
- DL4J Documentation: http://deeplearning4j.org/convolutionalnets
- Andrej Karpathy's CNN tutorial: https://cs231n.github.io/convolutional-networks/

Course Administrivia

- Project 2 due *tomorrow at 11:59pm*
- Project 3 out tomorrow! Due in **three weeks** (March 8)
  - Teams will be announced tomorrow
- **Final Project**
  - Teams of **2-3 people** (*you* get to decide your teams!)
  - Data Science project of *your* choice
  - Each team must submit a 1-2 page proposal by **March 9** (last day before spring break) outlining the project
  - Proposal must a) be **no more** than 2 pages, b) **including** references, c) describe the question, proposed solution/methods/frameworks, dataset, and how you intend to test/verify your solution (i.e., how you can determine your finished project answers the question you think it does)

Questions?