CS440/ECE448 Lecture 21
Computer Vision: Object Recognition

Mark Hasegawa-Johnson, 3/202

Including Slides by Svetlana Lazebnik

License: CC-BY 4.0

Computer Vision: Object Recognition

- Review: Gaussian and difference-of-Gaussians filtering
- Object detection using trainable filters
- Convolutional Neural Net
- The ImageNet task, and trained ImageNet filters
- Image generation using a generative adversarial network

Gaussian blur

Weighted averaging is a little better than unweighted averaging. We just need to make sure that the weights add up to one. For example:

$$S(x', y') = \sum_{m=-5}^{5} \sum_{n=-5}^{5} h(m,n) Y(x'-m, y'-n)$$

$$h(m,n) = \frac{1}{2\pi\sigma^2} e^{-\left(\left(\frac{m}{\sigma}\right)^2 + \left(\frac{n}{\sigma}\right)^2\right)}$$

The Gaussian blur filter $h(m,n)$ results in different degrees of smoothness of $S(x', y')$, depending on how we set the hyperparameter σ (the StDev):

By IkamusumeFan - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=41790217

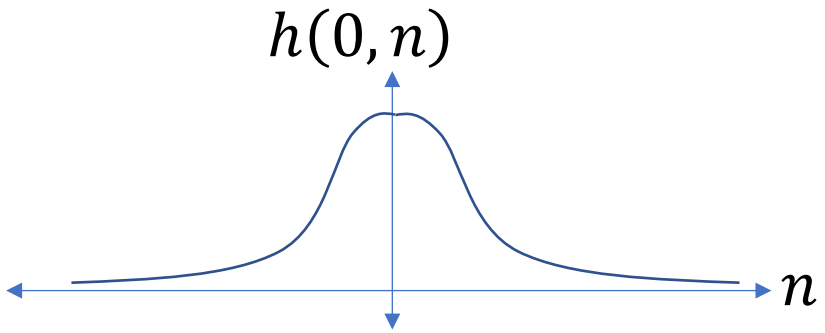
Gaussian blur

Weighted averaging is a little better than unweighted averaging. We just need to make sure that the weights add up to one. For example:

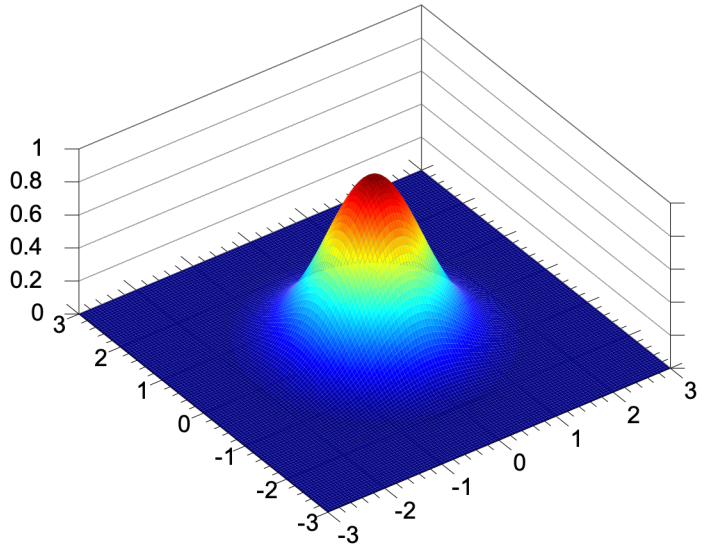
$$S(x', y') = \sum_{m=-5}^{5} \sum_{n=-5}^{5} h(m,n) Y(x'-m, y'-n)$$

$$h(m,n) = \frac{1}{2\pi\sigma^2} e^{-\left(\left(\frac{m}{\sigma}\right)^2 + \left(\frac{n}{\sigma}\right)^2\right)}$$

The Gaussian blur filter $h(m,n)$ looks kind of like this (plotted as a function of just n, for the coordinate $m = 0$):



Plotted in 2D, it looks kind of like this:



By Krishnavedala - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=35701251

Difference of Gaussians

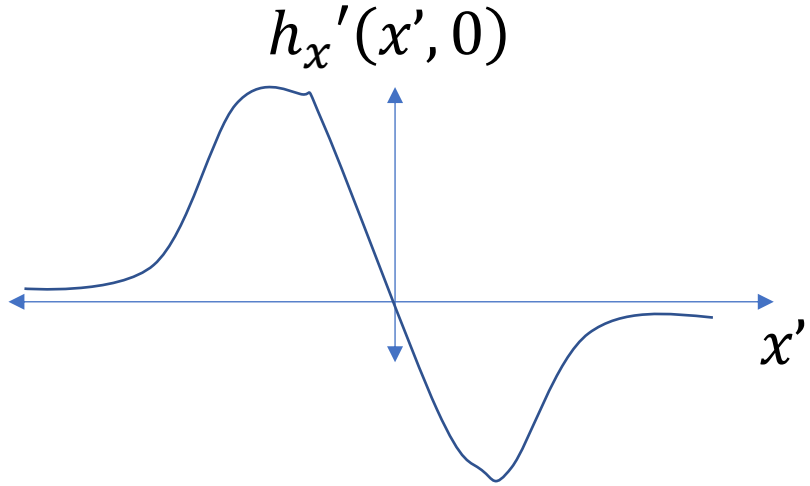
A "difference-of-Gaussians" filter is created by subtracting two Gaussian-blur filters, like this:

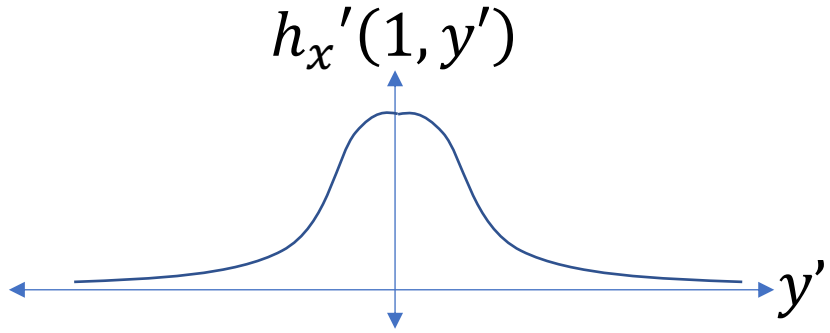
$$h(m,n) = \frac{1}{2\pi\sigma^2} e^{-\left(\left(\frac{m}{\sigma}\right)^2 + \left(\frac{n}{\sigma}\right)^2\right)}$$

$$h_x'(x',y') = \frac{(h(x'+1,y') - h(x'-1,y'))}{2}$$

$$h_y'(x',y') = \frac{(h(x',y'+1) - h(x',y'-1))}{2}$$

A "difference-of-Gaussians" filter looks kind of like this:





Difference of Gaussians

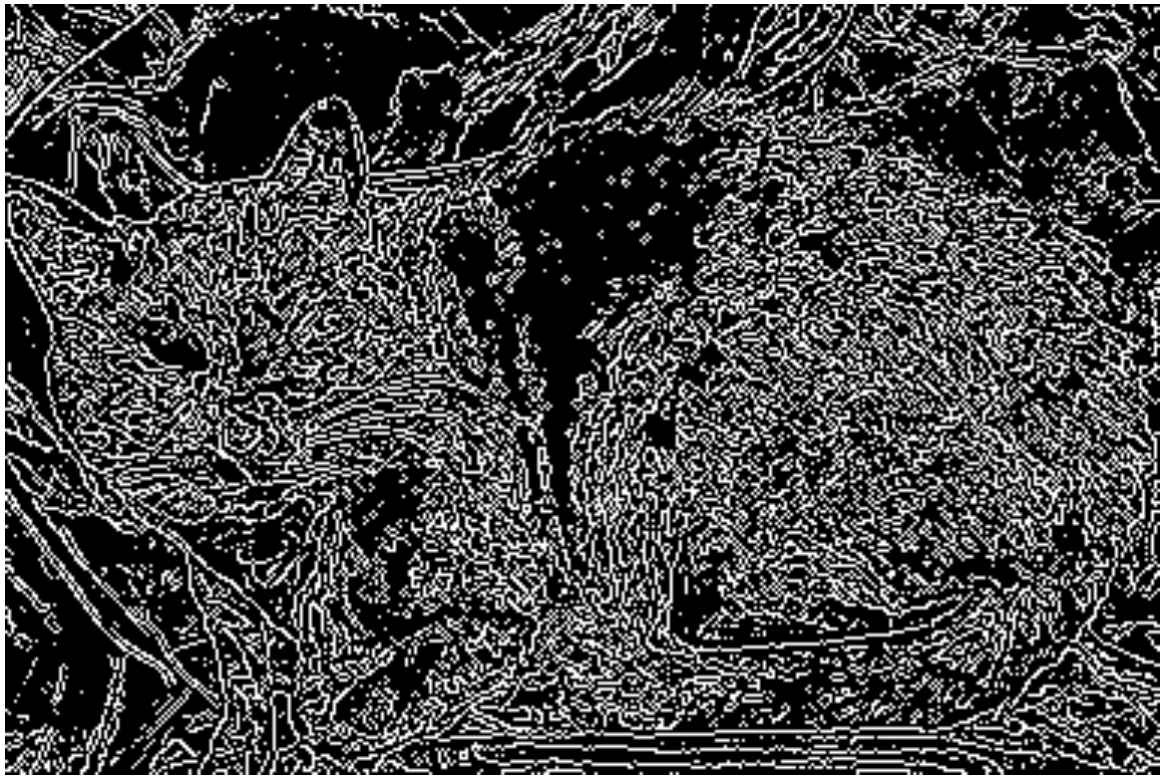
If we pre-compute the difference-of-Gaussians filters, then we can combine the weighted-average and the subtraction into just one operation, to save computation:

$$G_x(x', y') = \sum_{m=-5}^{5} \sum_{n=-5}^{5} h_x'(m,n) Y(x'-m, y'-n)$$

$$G_y(x', y') = \sum_{m=-5}^{5} \sum_{n=-5}^{5} h_y'(m,n) Y(x'-m, y'-n)$$

… so we never need to compute the smoothed image, $S(x', y')$, at all. We just go directly from the luminance, $Y(x'-m, y'-n)$, to the edge detection.

The difference-of-Gaussians filters, $h_x'(m,n)$ and $h_y'(m,n)$, detect more or less edges, depending on how we set the hyperparameter σ. Here is $\sqrt{G_x^2 + G_y^2}$, thresholded to make it black and white:



By Overremorto - Own work, Public Domain, https://commons.wikimedia.org/w/index.php?curid=10581259

Filtering, a.k.a. Convolution

This operation goes by two names. It can be called either “convolution,” or “filtering:”

$$D(x', y') = \sum_{m=-(M-1)/2}^{(M-1)/2} \sum_{n=-(M-1)/2}^{(M-1)/2} h(m, n) Y(x' - m, y' - n)$$

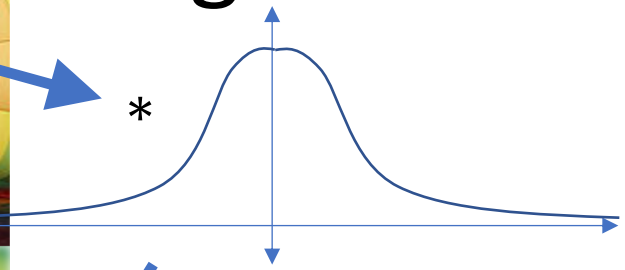
We’ll use the asterisk (*) to mean “convolved with.” In other words, we use the notation $D(x', y') = h(x', y') * Y(x', y')$ as a shorthand for the equation above.

The weights, $h(m, n)$, are called the ”filter.” The way I’ve written it here, the filter has a size of $M \times M = M^2$ pixels.

The input is the image $Y(x', y')$. The output is $D(x', y')$.

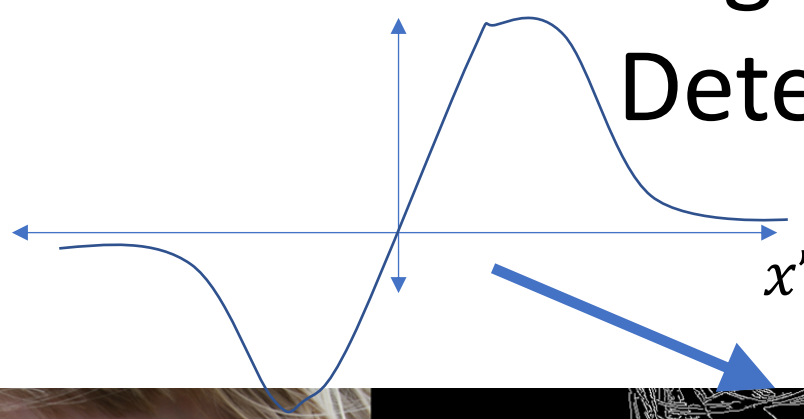
Example Convolutions: Weighted Average, Edge Detector

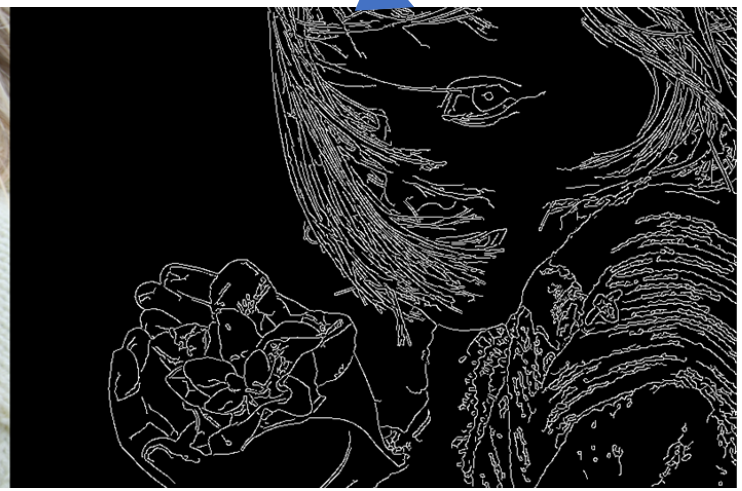
Weighted Average



$h_x'(x', 0)$

Edge Detector





By IkamusumeFan - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=41790217

Jon McLoone, CC-SA 3.0,https://commons.wikimedia.org/wiki/File:%C3%84%C3%A4retuvastuse_n%C3%A4ide.png

Computer Vision: Object Recognition

- Review: Gaussian and difference-of-Gaussians filtering
- Object detection using trainable filters
- Convolutional Neural Net
- The ImageNet task, and trained ImageNet filters
- Image generation using a generative adversarial network

Object detection using trainable filters

Suppose you have images of two different types of objects, e.g., beetles and bicycles.

One way to tell them apart is by creating two different filters, $h_0(x', y')$ and $h_1(x', y')$. Then if you want to tell whether image $Y(x', y')$ contains a beetle or a bicycle, you compute

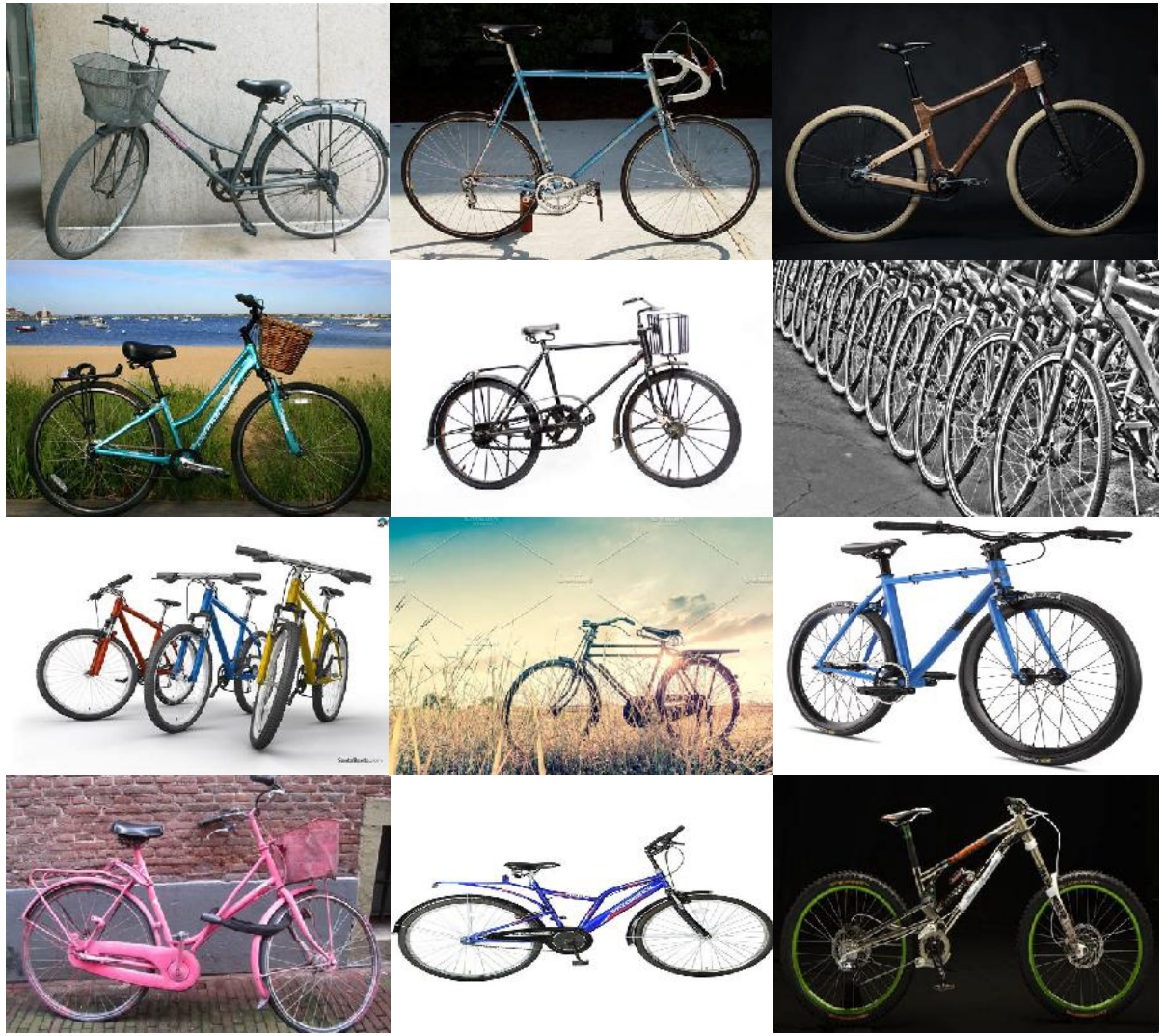
$$D_0(x', y') = h_0(x', y') * Y(x', y')$$

and

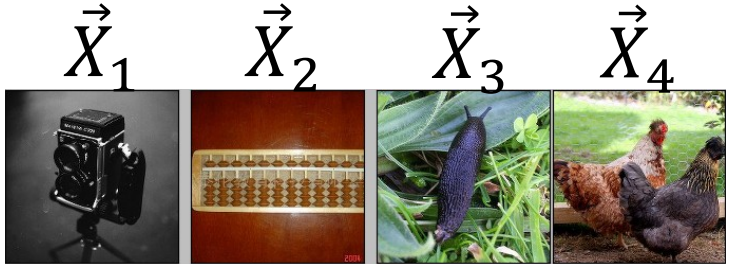
$$D_1(x', y') = h_1(x', y') * Y(x', y')$$

Then, if $\max_{x', y'} D_0(x', y') > \max_{x', y'} D_1(x', y')$, you call it a beetle. Otherwise, you call it a bicycle.

Example: beetles versus bicycles



Training the filter



$\vec{X}_1$ $\vec{X}_2$ $\vec{X}_3$ $\vec{X}_4$

Y_1 = "camera" Y_2 = "abacus" Y_3 = "slug" Y_4 = "chickens"

- A "training database" is a large set of labeled training images, each labeled with a "ground truth" or "reference" image label.
- We use this database to "train" the filters, i.e., to find the following filters:
  - $h_0(x', y')$ is something that makes $D_0(x', y')$ big for all images that are known to contain beetles.
  - $h_1(x', y')$ is something that makes $D_1(x', y')$ big for all images that are known to contain bicycles.

Designing a matched filter by averaging all of the input data

- Given: 12 example images of beetles.
- Goal: design a filter $h_0(x', y')$ that will maximize

$$D_0(x', y') = h_0(x', y') * Y(x', y')$$

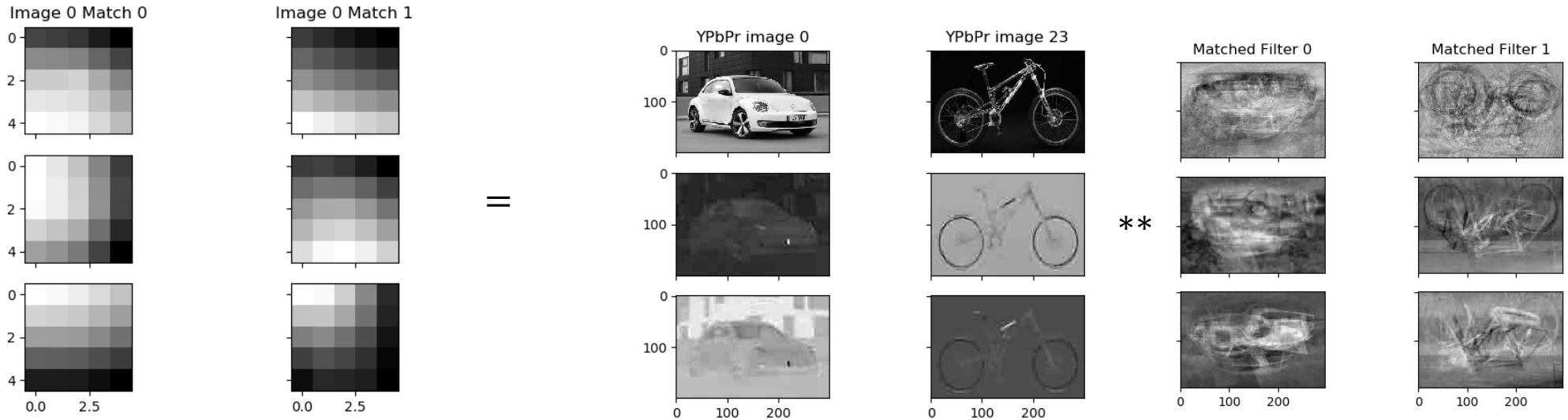
…on average, for all of the images in the training database.

The surprising solution: average the training images!

Matched filters for bicycles and beetles

Match = input image, convolved with matched filter

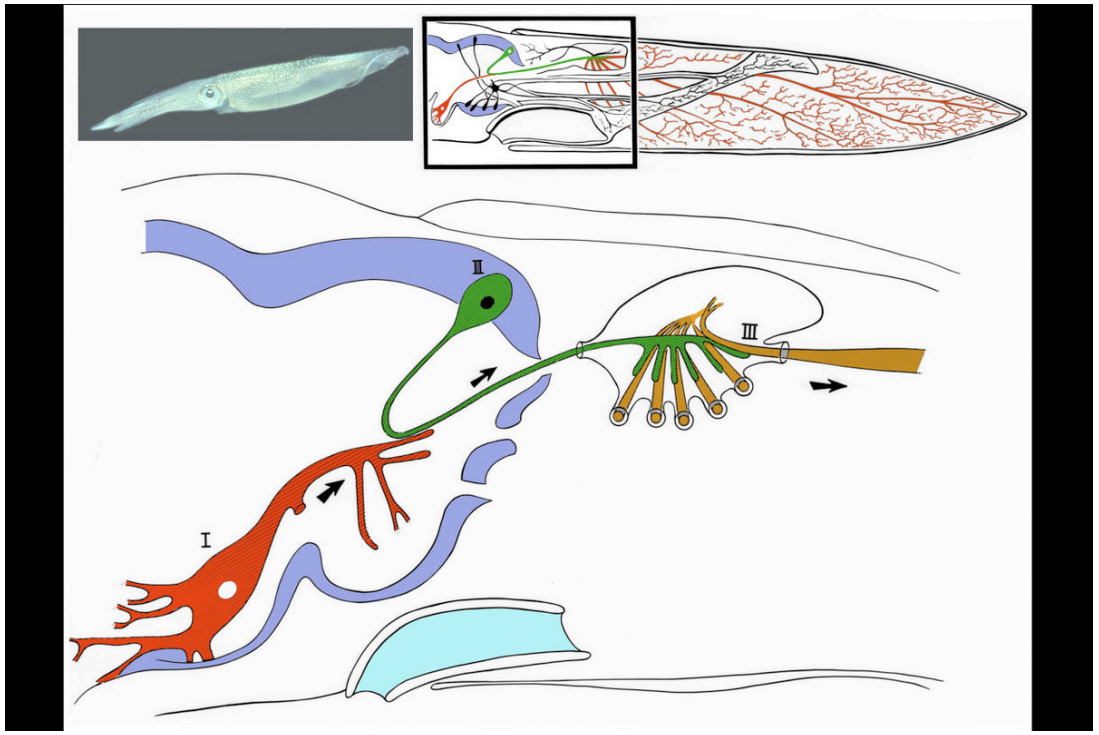
$$D_0(x', y') = h_0(x', y') * Y(x', y')$$
and
$$D_1(x', y') = h_1(x', y') * Y(x', y')$$



Computer Vision: Object Recognition

- Review: Gaussian and difference-of-Gaussians filtering
- Object detection using trainable filters
- Convolutional Neural Net
- The ImageNet task, and trained ImageNet filters
- Image generation using a generative adversarial network

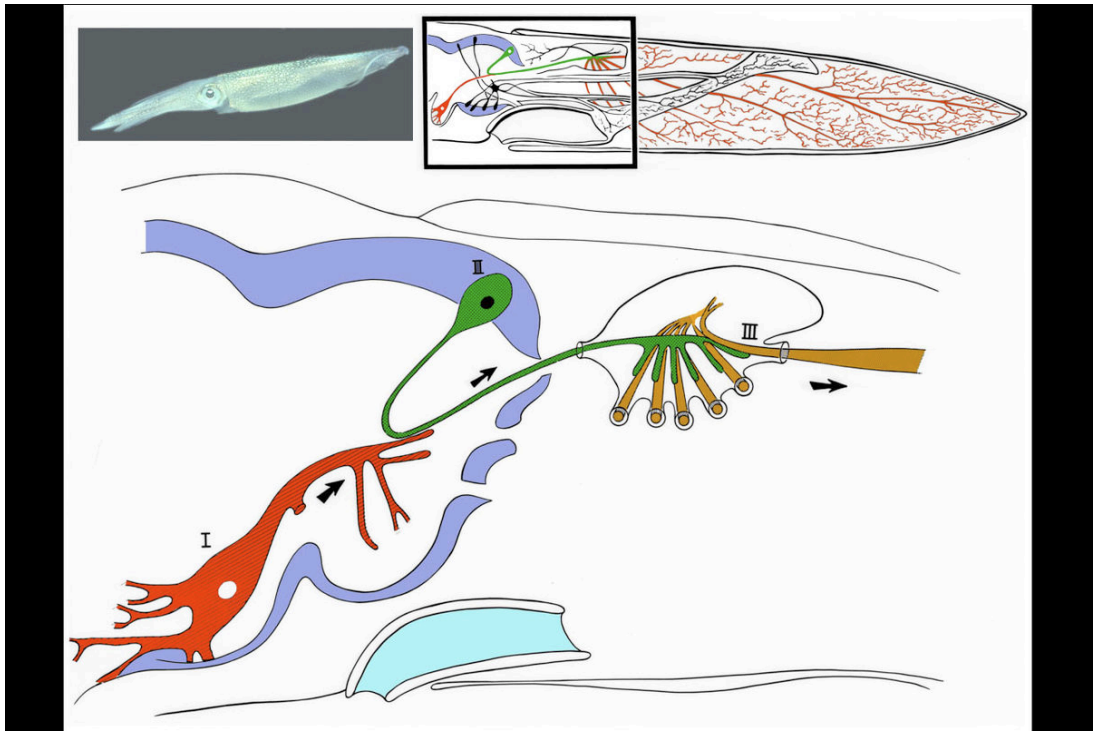
Neural net



An artificial neural network is a parameterized function, whose equations imitate the propagation of signals between biological neurons.

We will talk about how it is used, and how it is trained, later in the course. You don't need to know that today.

Notation



A deep neural net has thousands of neurons (nodes).

Each neuron (node) has two key variables:

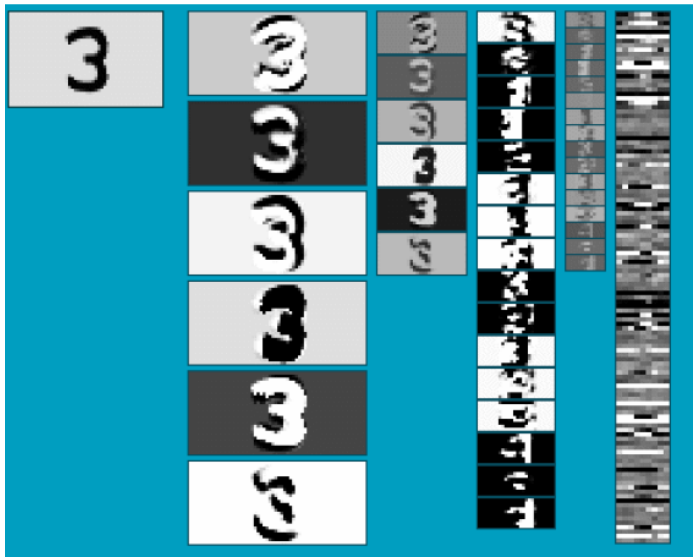
- The "excitation", D, models the synapse of a biological neuron.
- The "activation," A, models the axon of a biological neuron, i.e., it's zero when the input is negative, and nonzero when the input is positive:

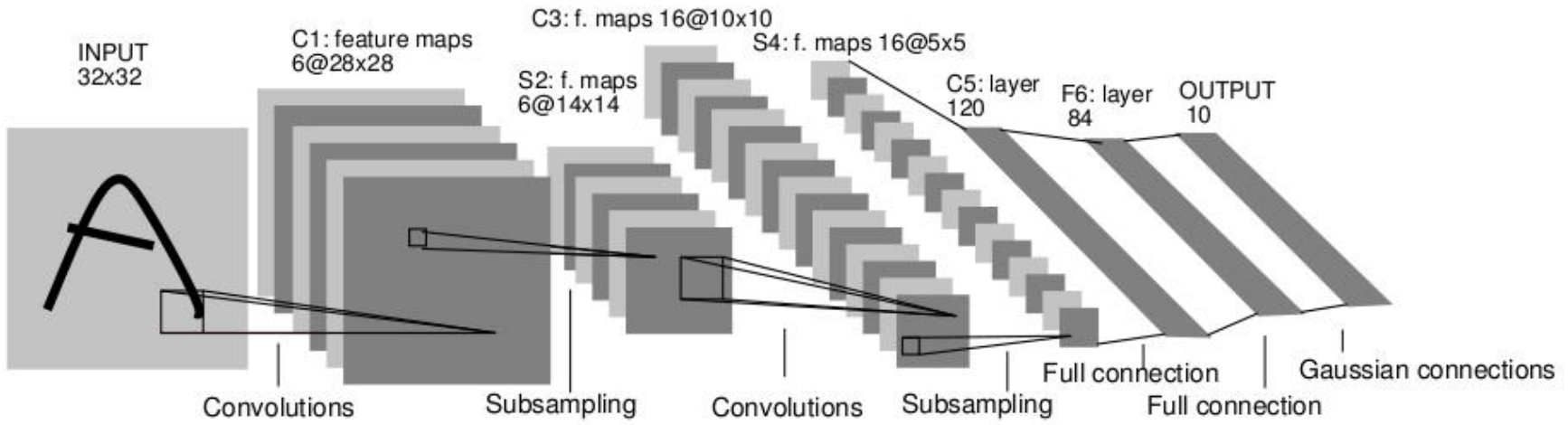
$$A = g(D)$$

…where $g(D)$ is some nonlinearity that is non-negative.

Convolutional Neural Networks

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

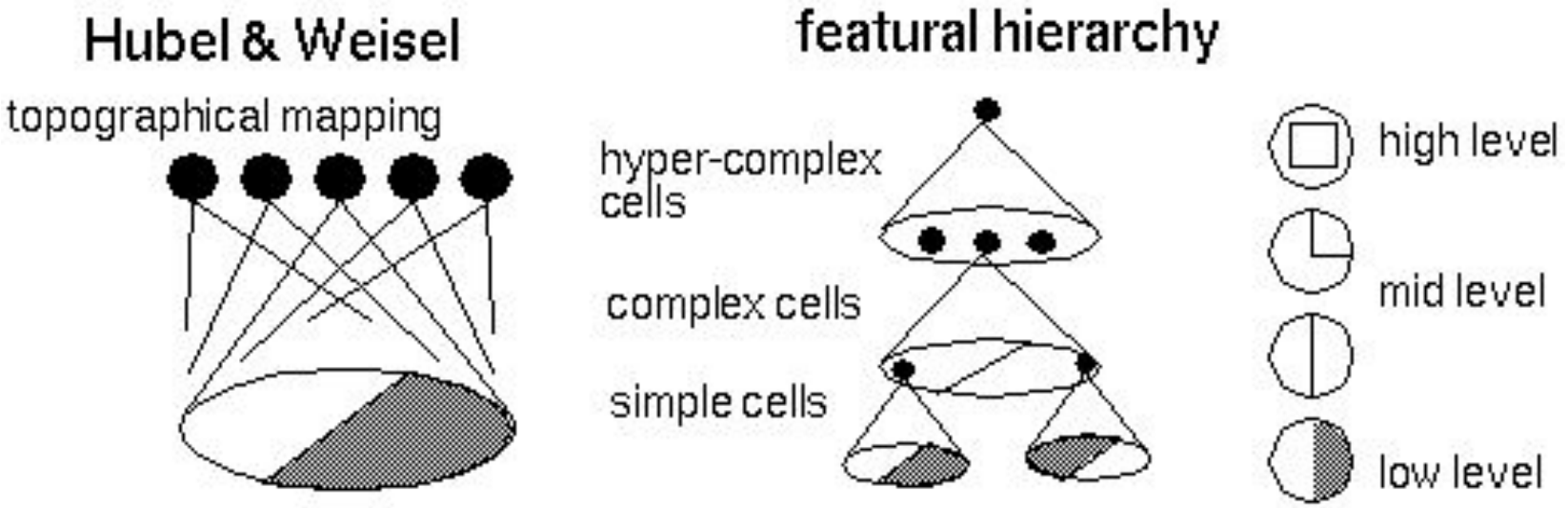




Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86(11): 2278–2324, 1998.

Biological inspiration

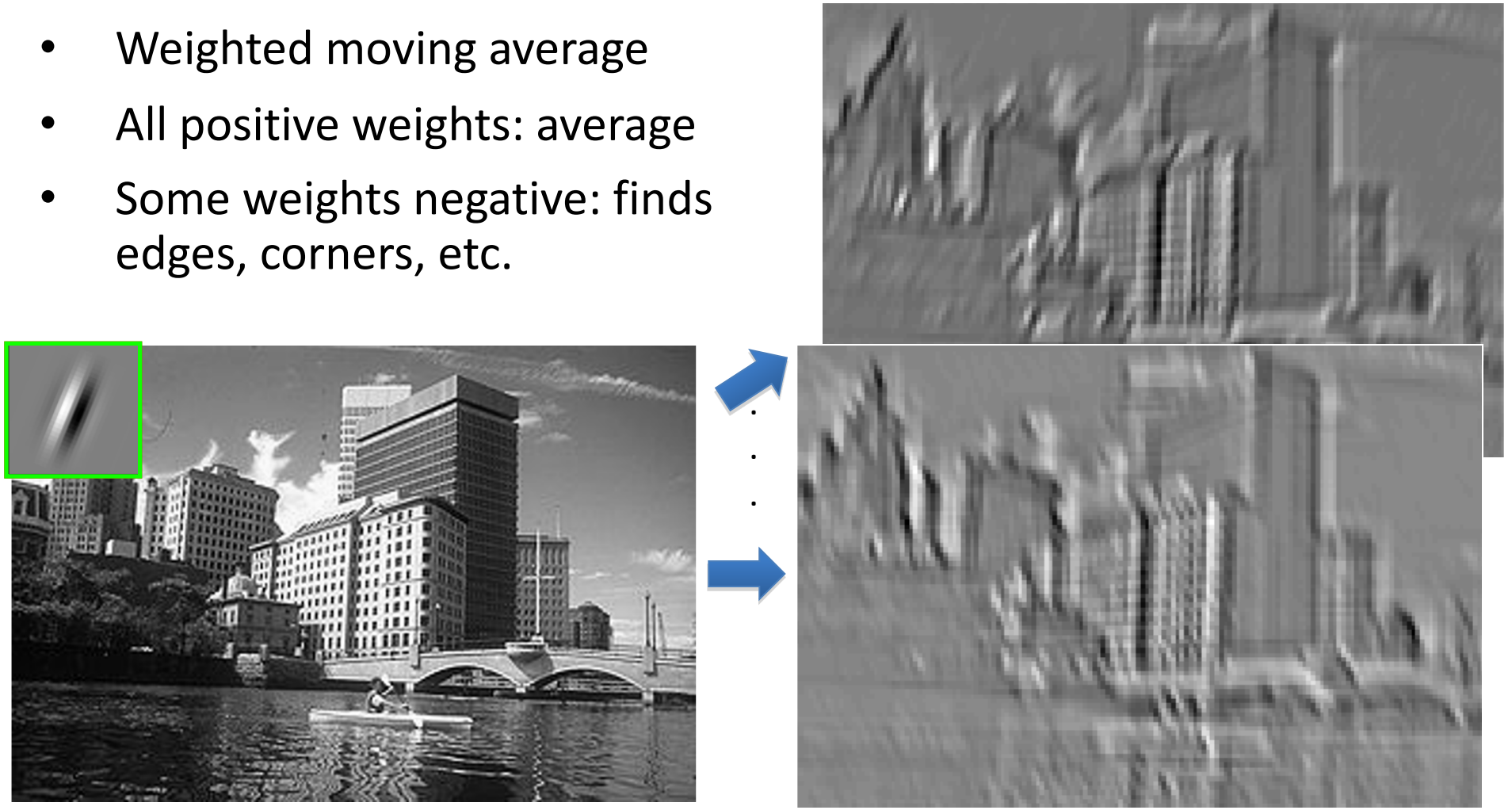
- D. Hubel and T. Wiesel (1959, 1962, Nobel Prize 1981)
  - Visual cortex consists of a hierarchy of *simple*, *complex*, and *hyper-complex* cells



Source

What is a convolution?

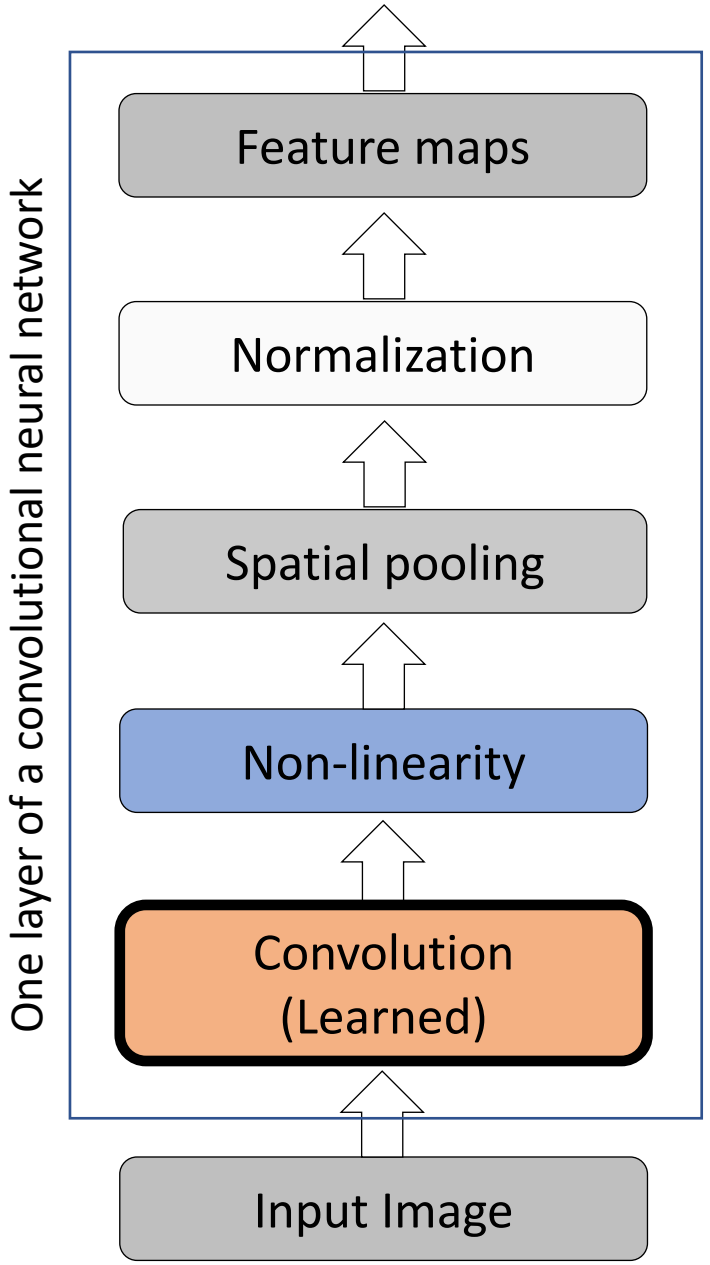
- Weighted moving average
- All positive weights: average
- Some weights negative: finds edges, corners, etc.



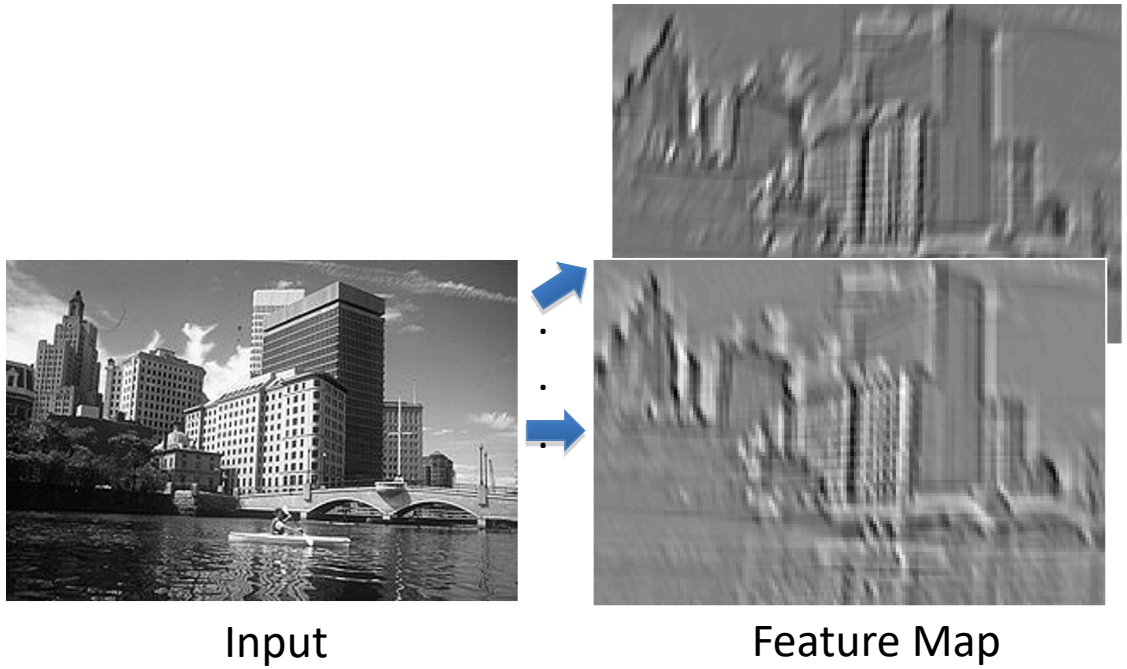
Input

Output Channels (edges with different orientations)

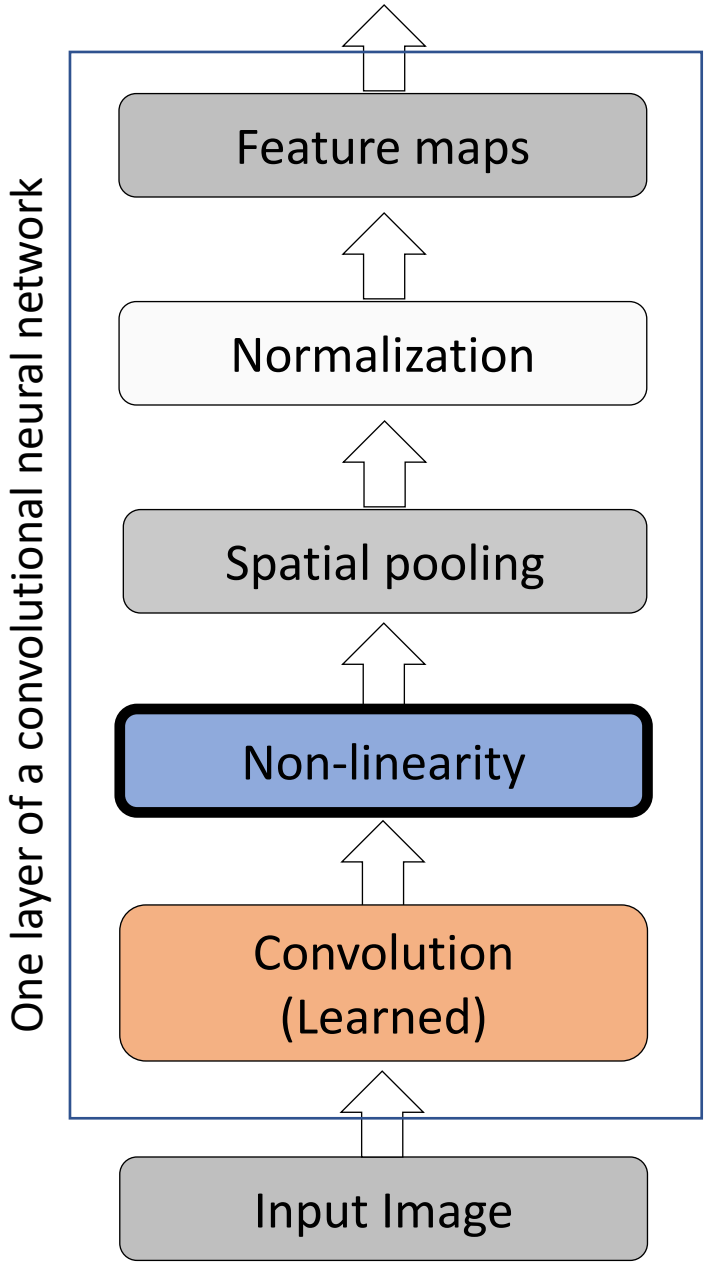
Convolutional Neural Networks



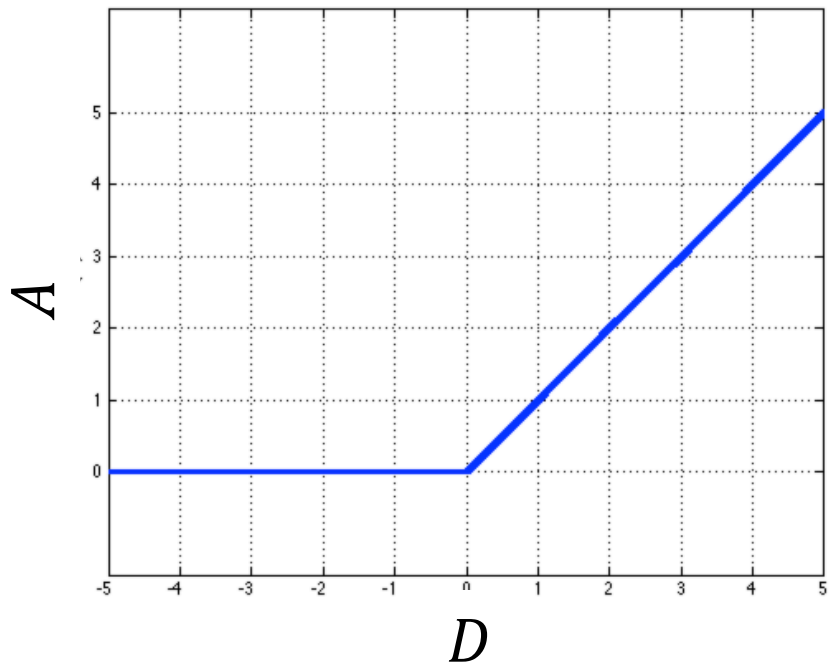
$$D(x', y') = h(x', y') * Y(x', y')$$



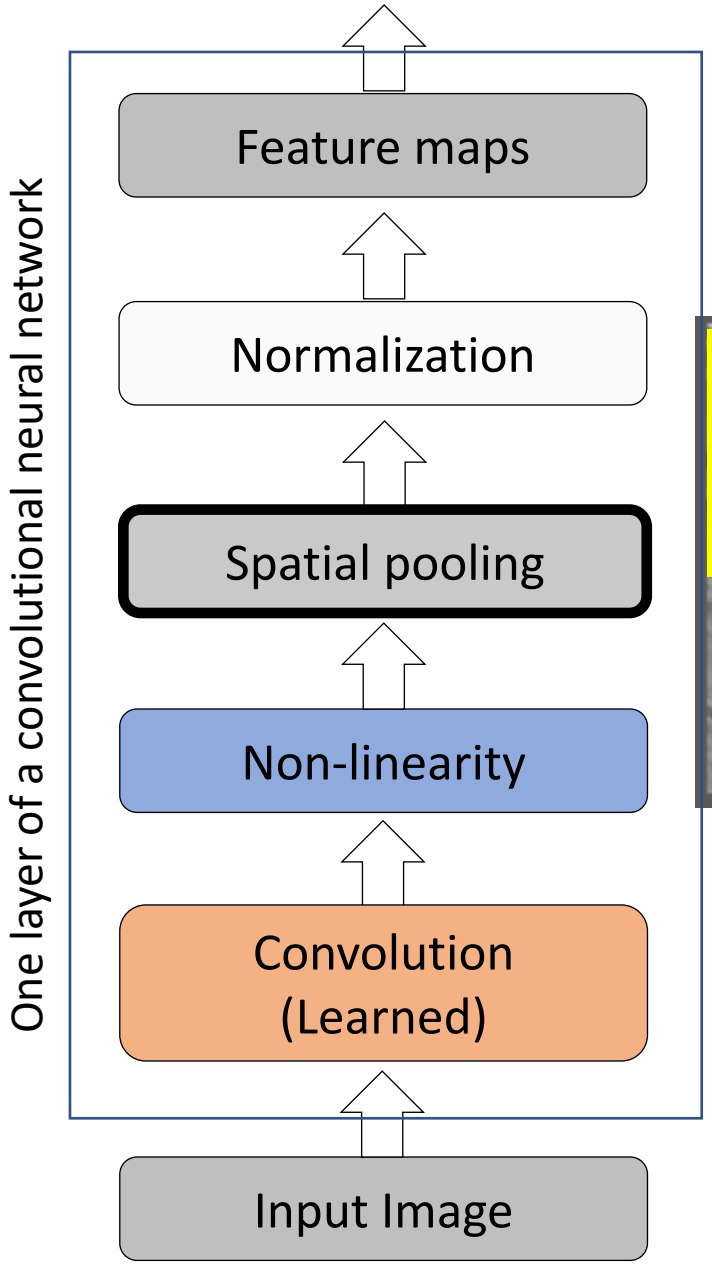
Convolutional Neural Networks



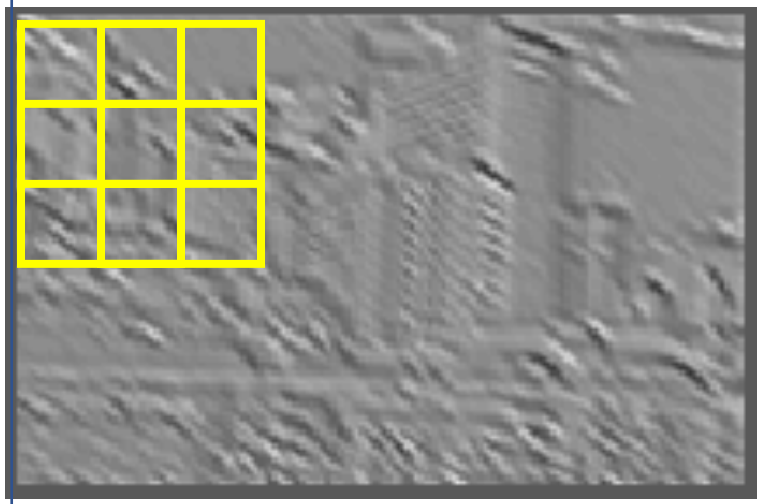
$$A = g(D)$$



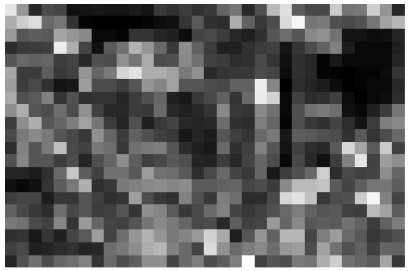
Convolutional Neural Networks



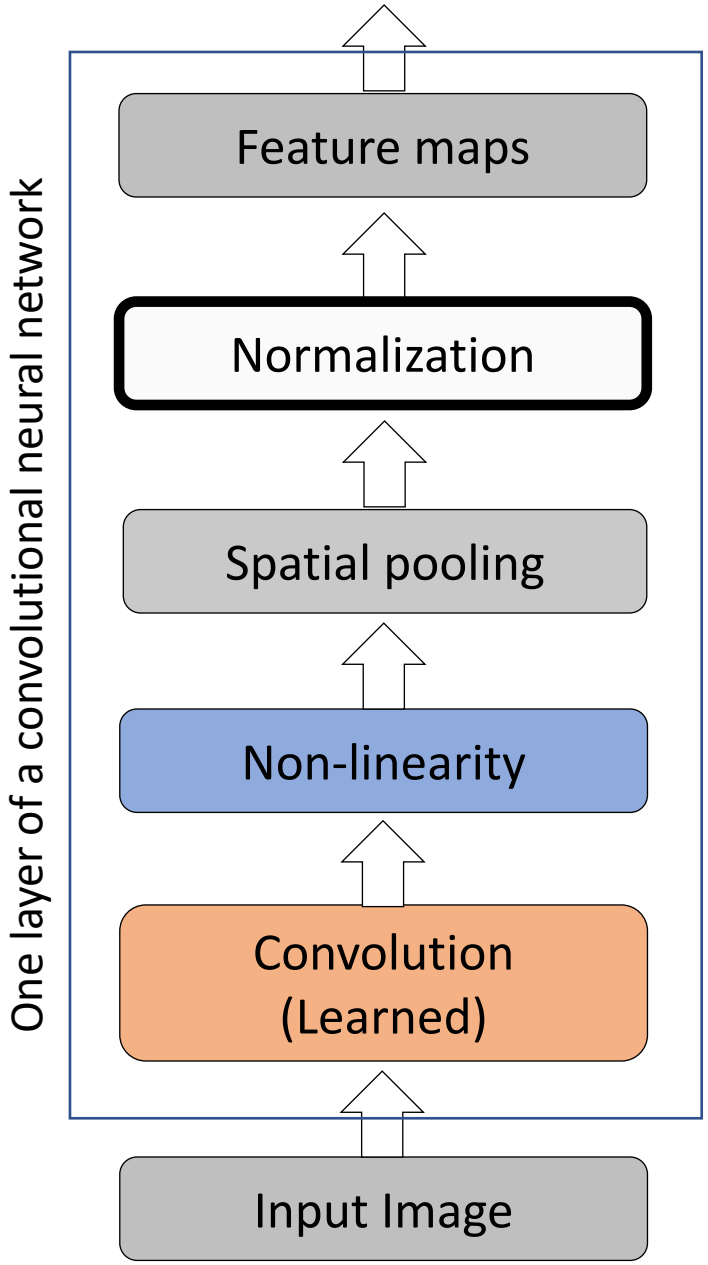
$\max_{x',y'} A(x', y')$, within some local region.

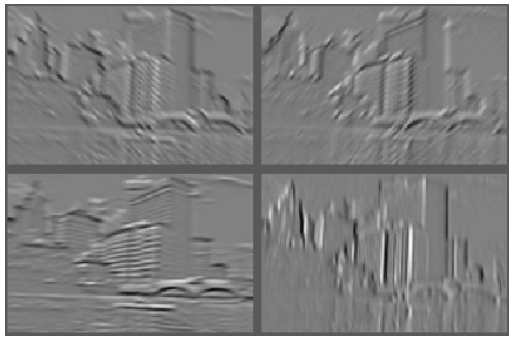


Max

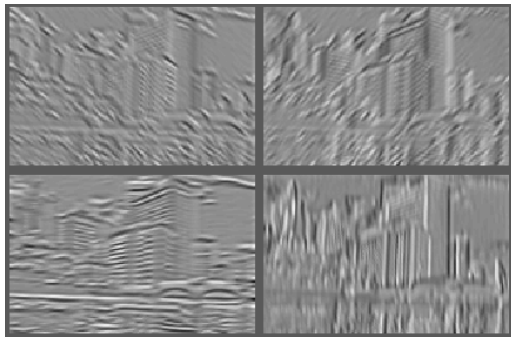


Convolutional Neural Networks



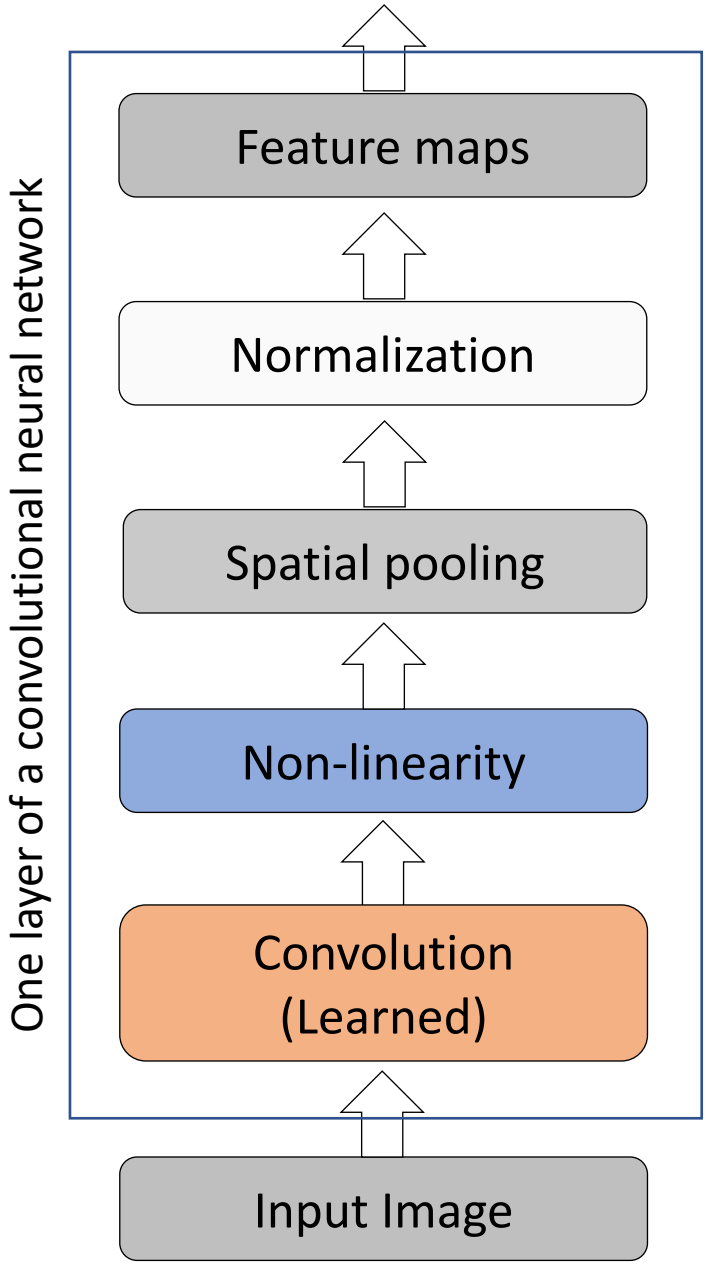


Feature Maps



Feature Maps
After Contrast
Normalization

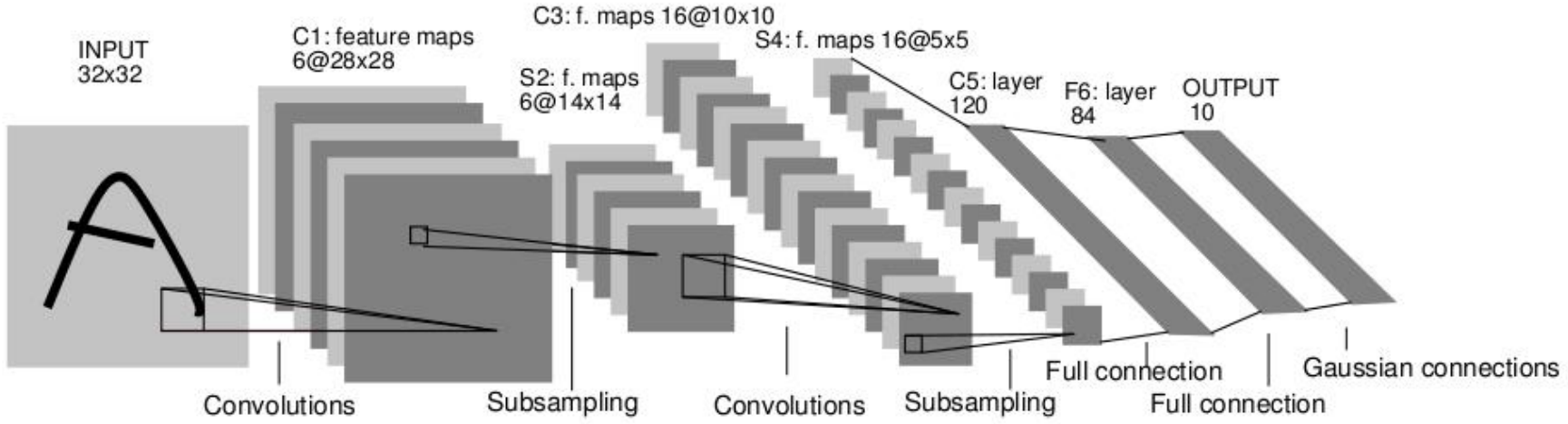
Convolutional Neural Networks



- Convolutional filters are trained using labeled training data.
- Basically, you can think of the top layer as a classifier, and the layer below it learns features. And its features are computed from the outputs of the layer below that, and so on.
- The coefficients of all of those filters, in every layer, are chosen to minimize the error rate on the training data.

LeNet, 1998

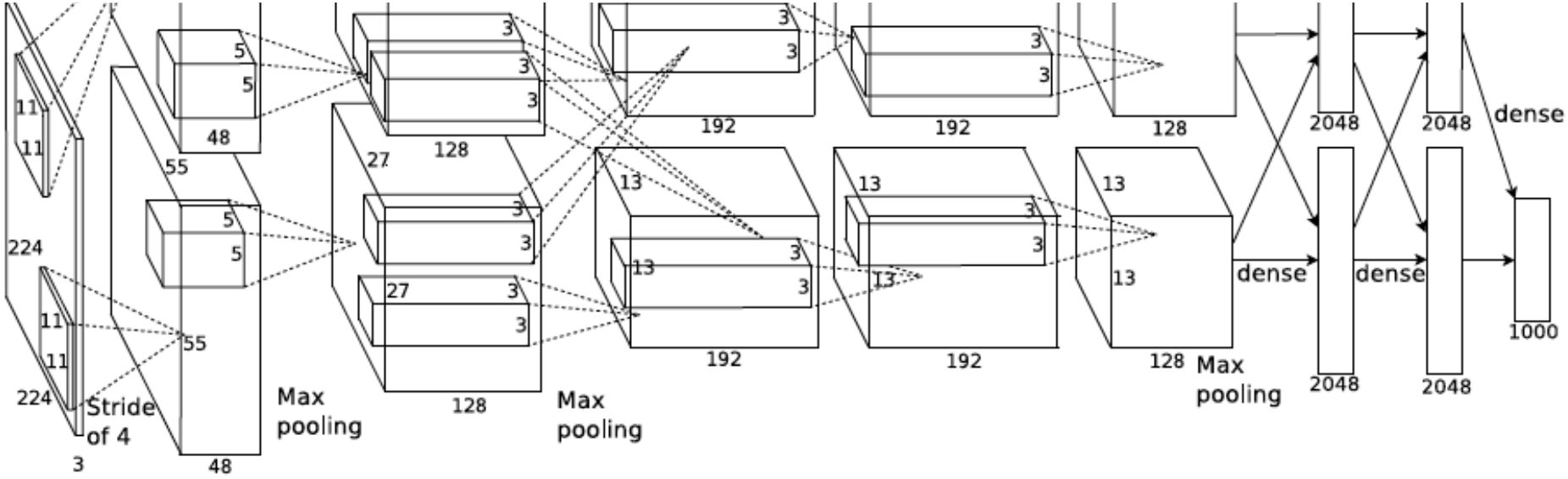
- (Convolution, Nonlinearity, Maximum-Subsampling, Normalization) X 2 layers
- After the second subsampling, the resulting features are concatenated to form a vector, which is then classified.



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

AlexNet

- Similar framework to LeCun'98 but:
  - Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
  - More data (10^6 vs. 10^3 images)
  - GPU implementation (50x speedup over CPU)
    - Trained on two GPUs for a week

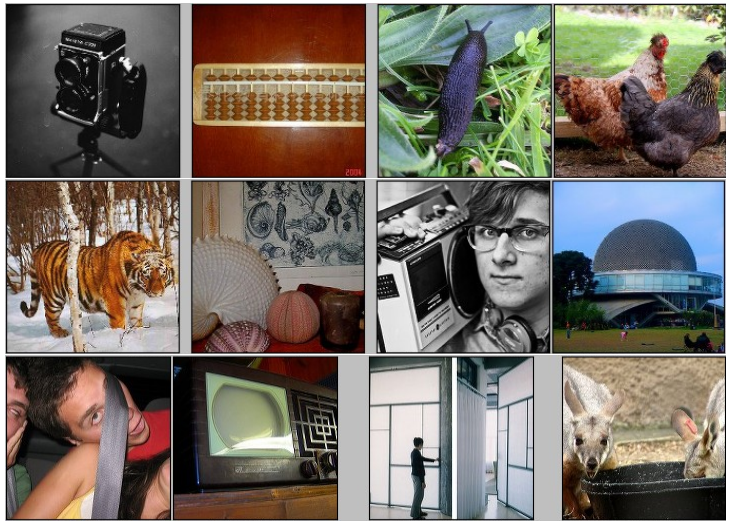


A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

Computer Vision: Object Recognition

- Review: Gaussian and difference-of-Gaussians filtering
- Object detection using trainable filters
- Convolutional Neural Net
- The ImageNet task, and trained ImageNet filters
- Image generation using a generative adversarial network

ImageNet Challenge

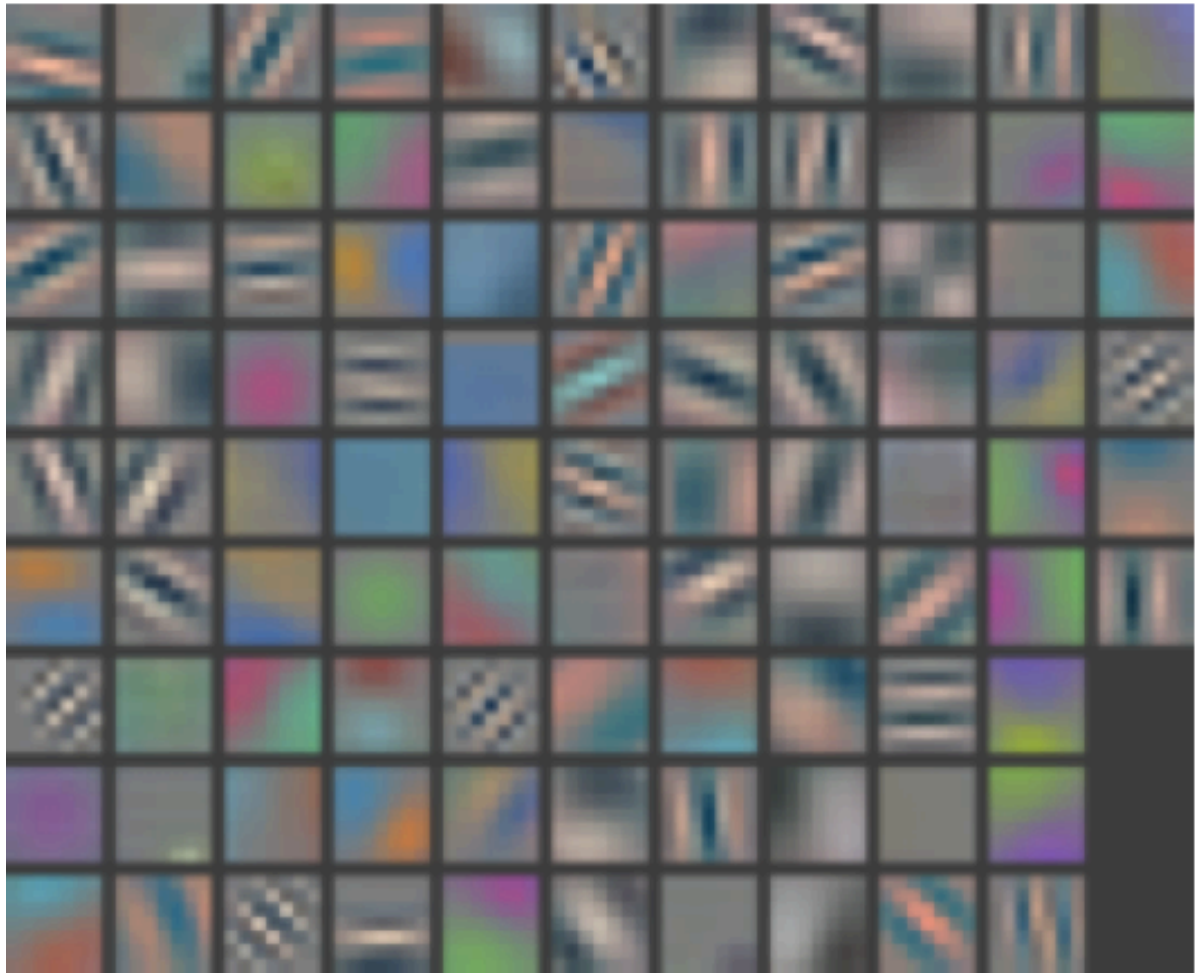


[Deng et al. CVPR 2009]

- ~14 million labeled images, 20k classes
- Images gathered from Internet
- Human labels via Amazon MTurk
- Challenge: 1.2 million training images, 1000 classes

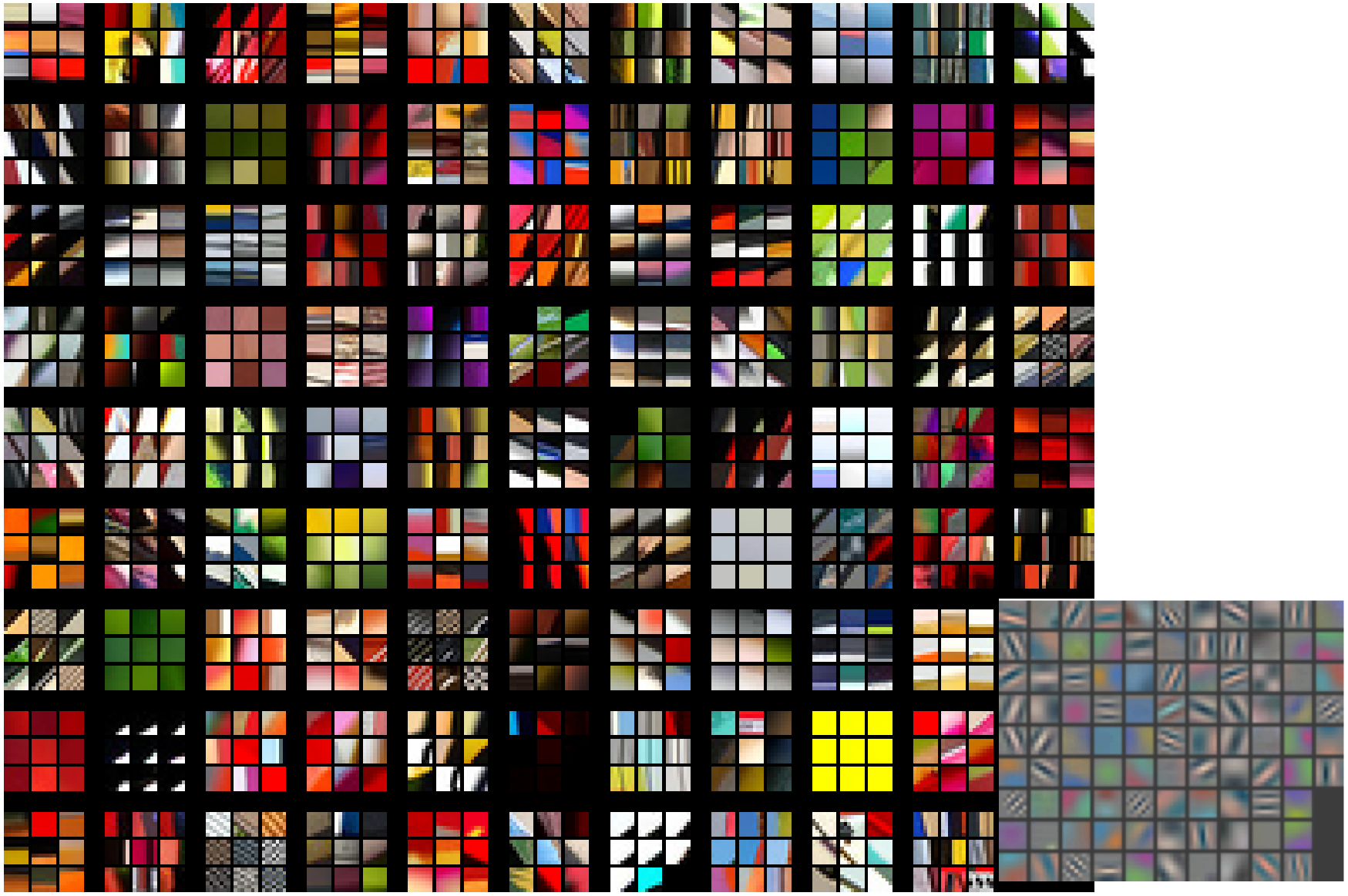
A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

Layer 1 Filters



M. Zeiler and R. Fergus, Visualizing and Understanding Convolutional Networks, arXiv preprint, 2013

Layer 1: Top-9 Patches



Layer 2: Top-9 Patches

- Patches from validation images that give maximal activation of a given feature map

Layer 3: Top-9 Patches

Layer 4: Top-9 Patches

Layer 5: Top-9 Patches

Computer Vision: Object Recognition

- Review: Gaussian and difference-of-Gaussians filtering
- Object detection using trainable filters
- Convolutional Neural Net
- The ImageNet task, and trained ImageNet filters
- Image generation using a generative adversarial network

Semantic Image Inpainting with Deep Generative Models

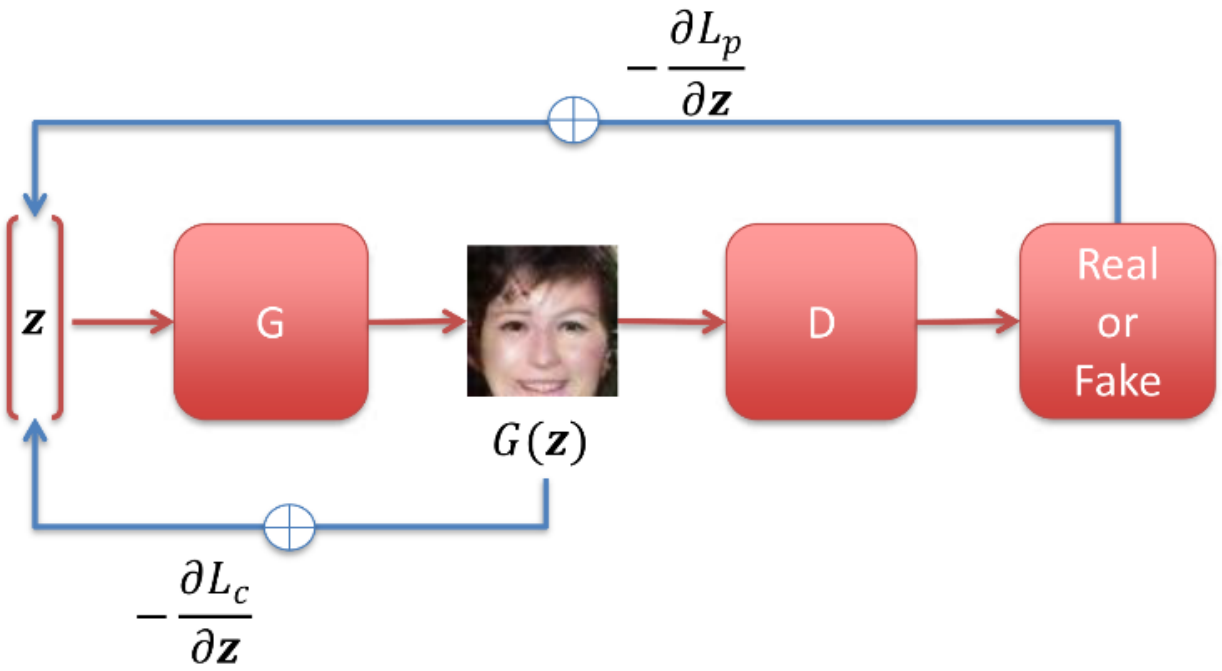
Raymond Yeh, Chen Chen, Teck Yian Lim, Alexander G. Schwing, Mark Hasegawa-Johnson and Minh Do

The problem:

Input

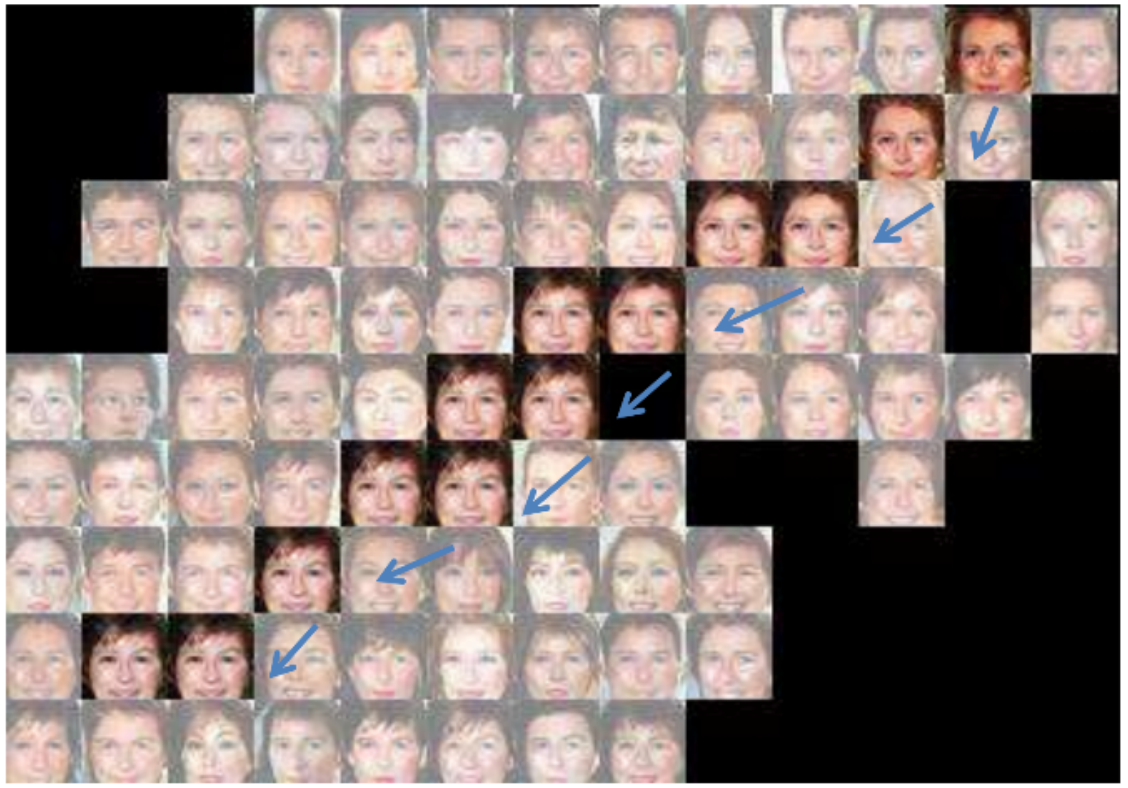
Semantic Image Inpainting

The solution:



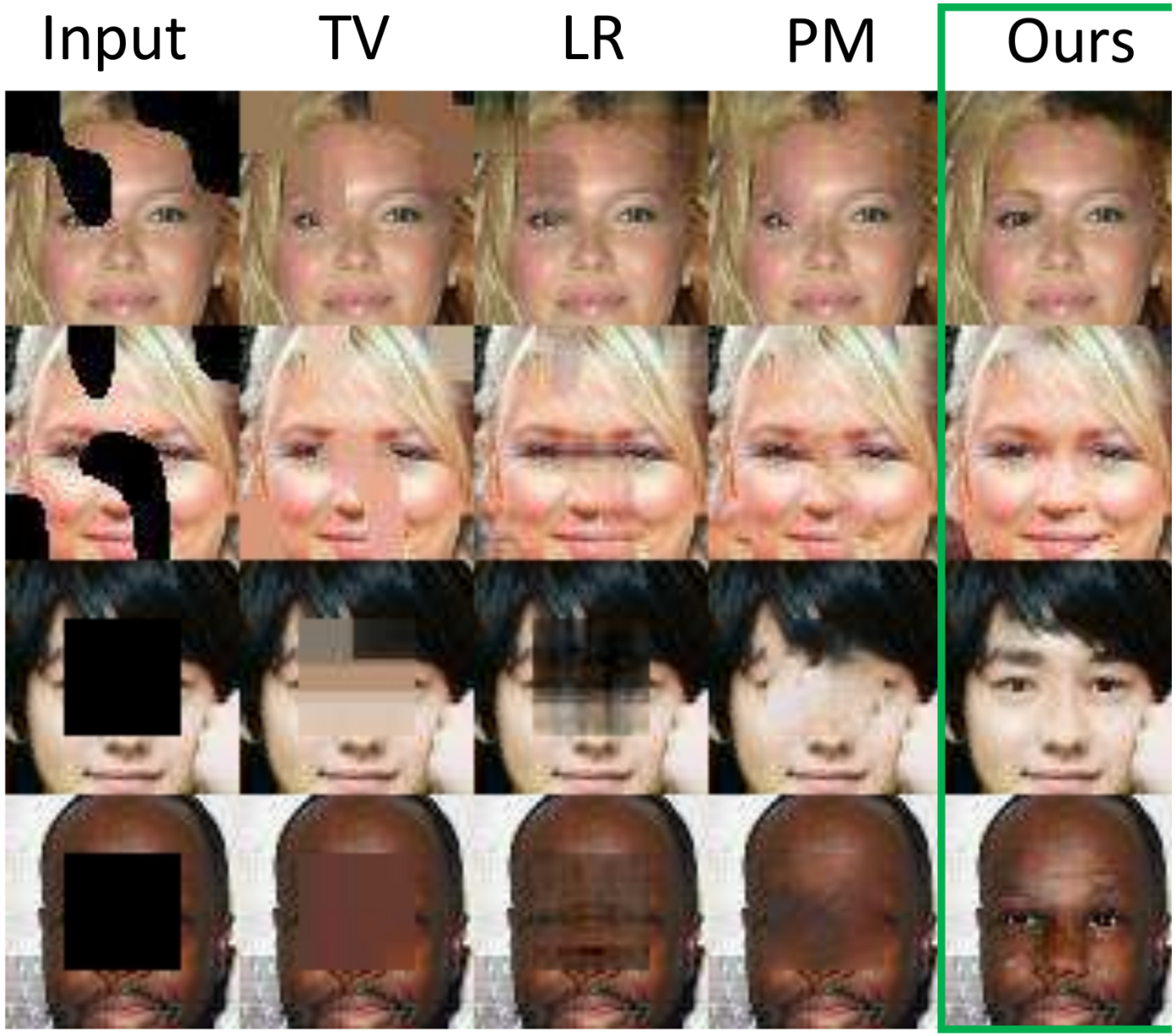
$$Loss = L_p(z) + L_c(z|$$,)

y $\mathbf{M}$



Semantic Image Inpainting

The results:



Conclusions

- Weighted averaging and edge detection are two examples of an operation called “convolution.”
- It’s possible to detect objects, in an image, by convolving with a trained filter, and looking for the maximum.
- A convolutional neural network is a series of layers, each of which contains a convolution, followed by a nonlinearity, followed by a local maximum.
- ImageNet is a database with millions of images, showing examples of over 1000 different objects. It has permitted people to train extremely complicated and highly accurate neural nets.