# Deep Learning in Image Processing

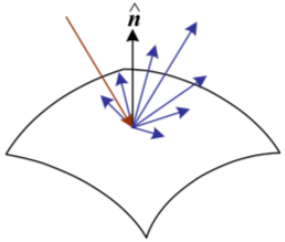
Topics:

- CNNs 101
- Image Processing Pipelines

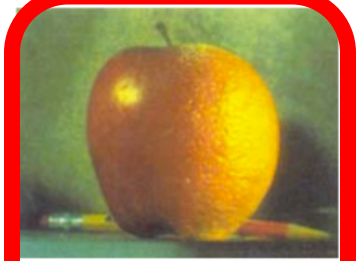
Frank Dellaert

CS 4476 Computer Vision

Many slides from Stanford's CS231N by Fei-Fei Li, Justin Johnson, Serena Yeung, as well as some slides on filtering from Devi Parikh and Kristen Grauman, who may in turn have borrowed some from others

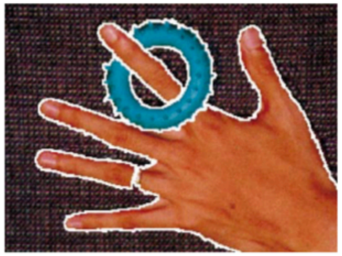


2. Image Formation



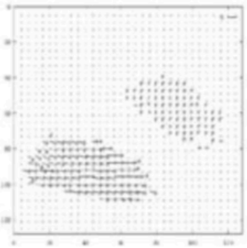
3. Image Processing

4. Features

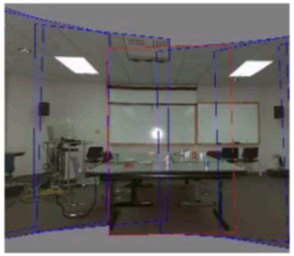


5. Segmentation

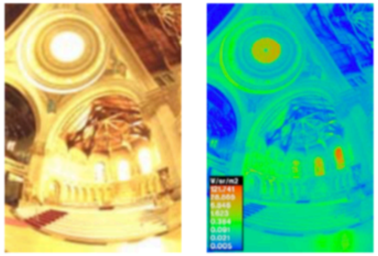
6-7. Structure from Motion



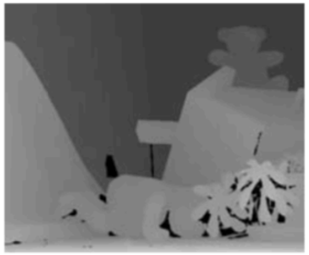
8. Motion



9. Stitching

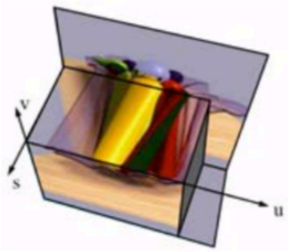


10. Computational Photography



11. Stereo

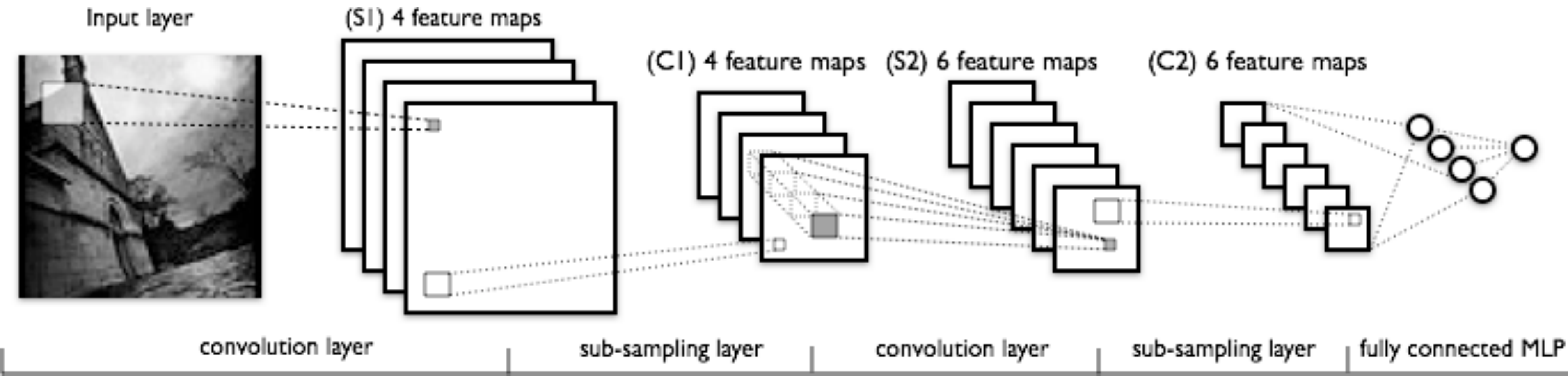
12. 3D Shape




13. Image-based Rendering

14. Recognition

# Convolutional Neural Networks



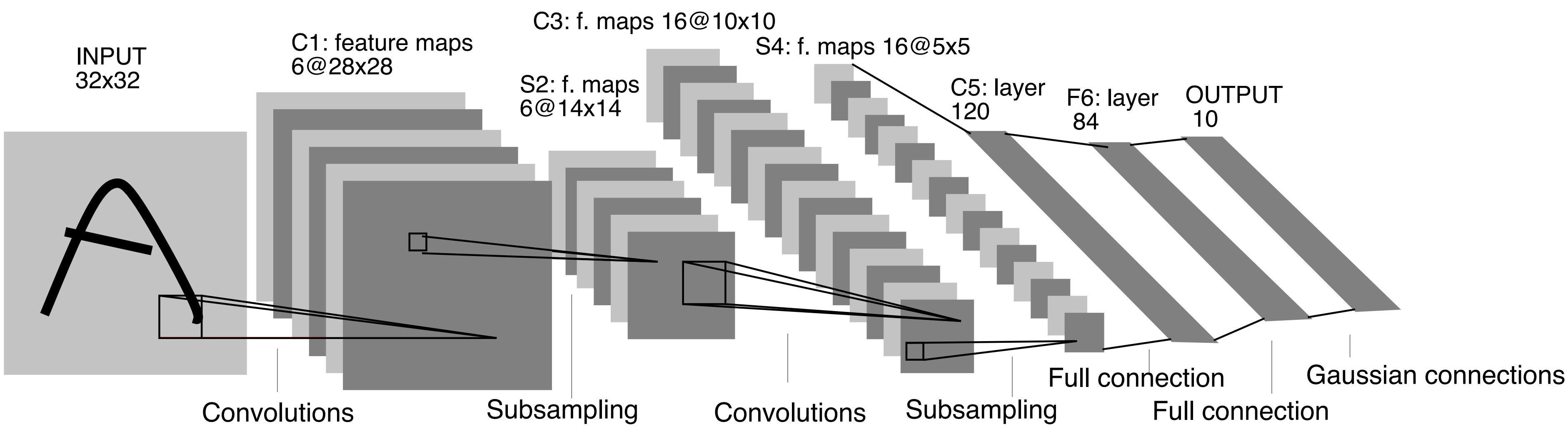
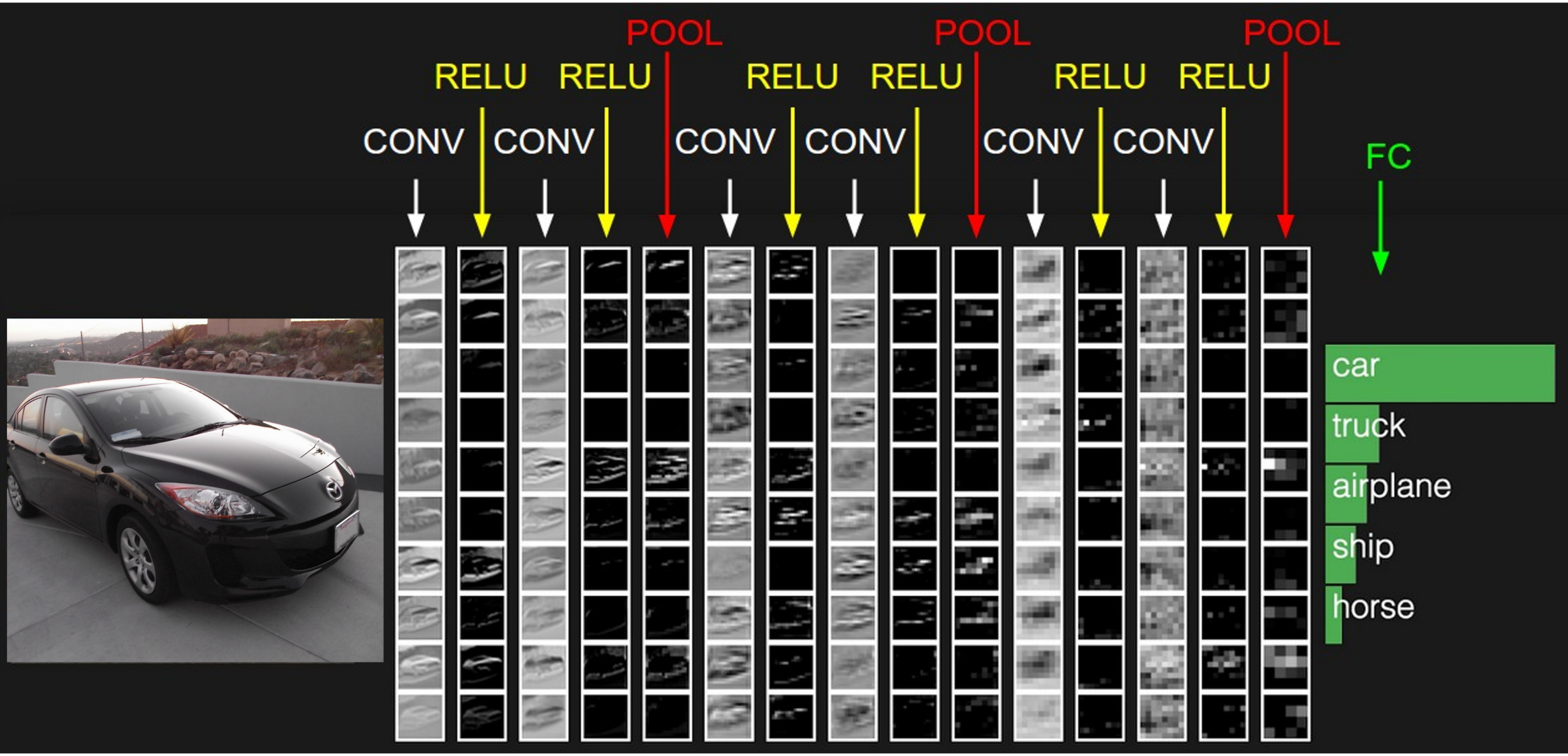





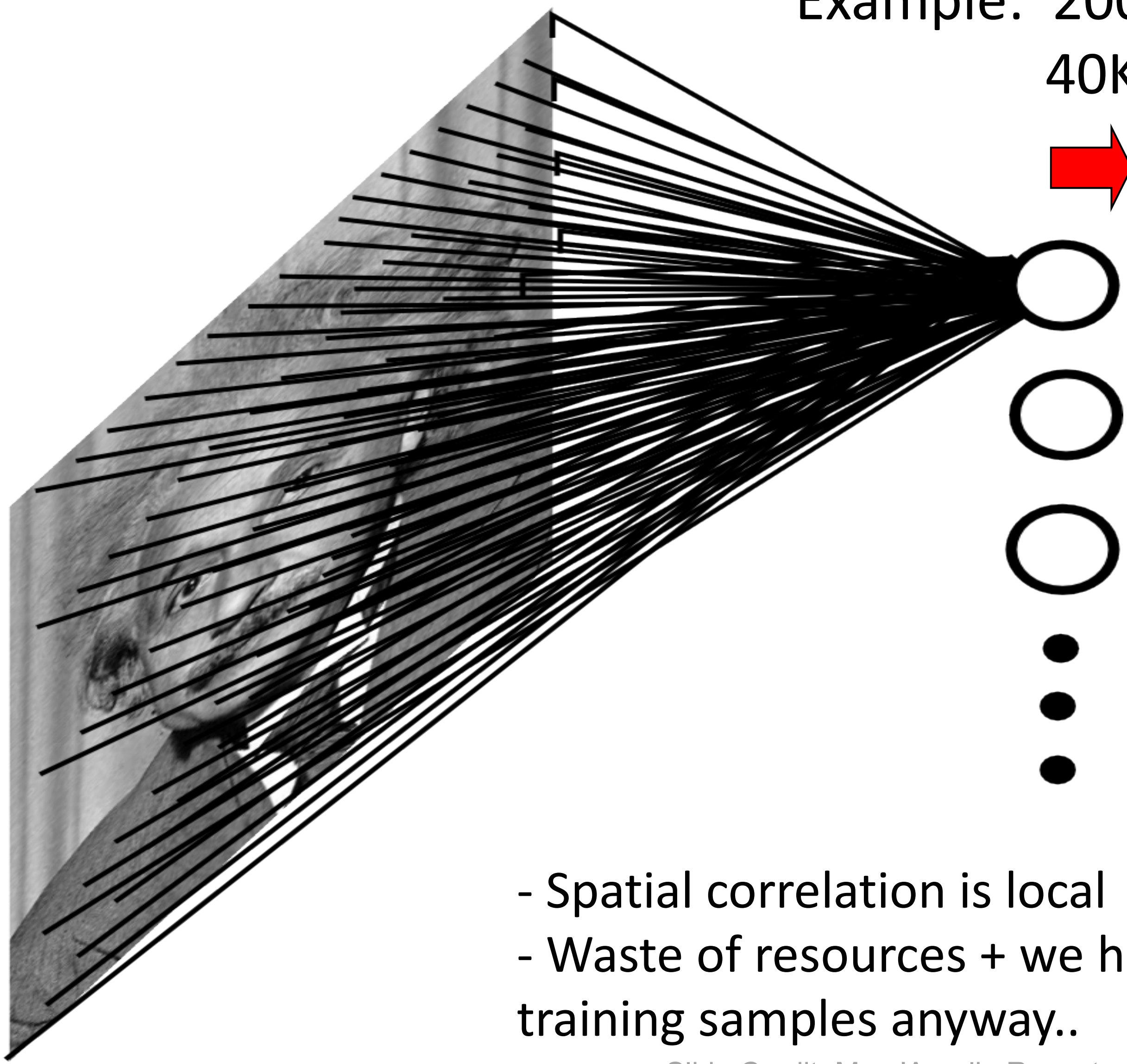

Image Credit: Yann LeCun, Kevin Murphy

preview:


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

# Fully Connected Layer

Example: 200x200 image
40K hidden units
**~2B parameters**!!!

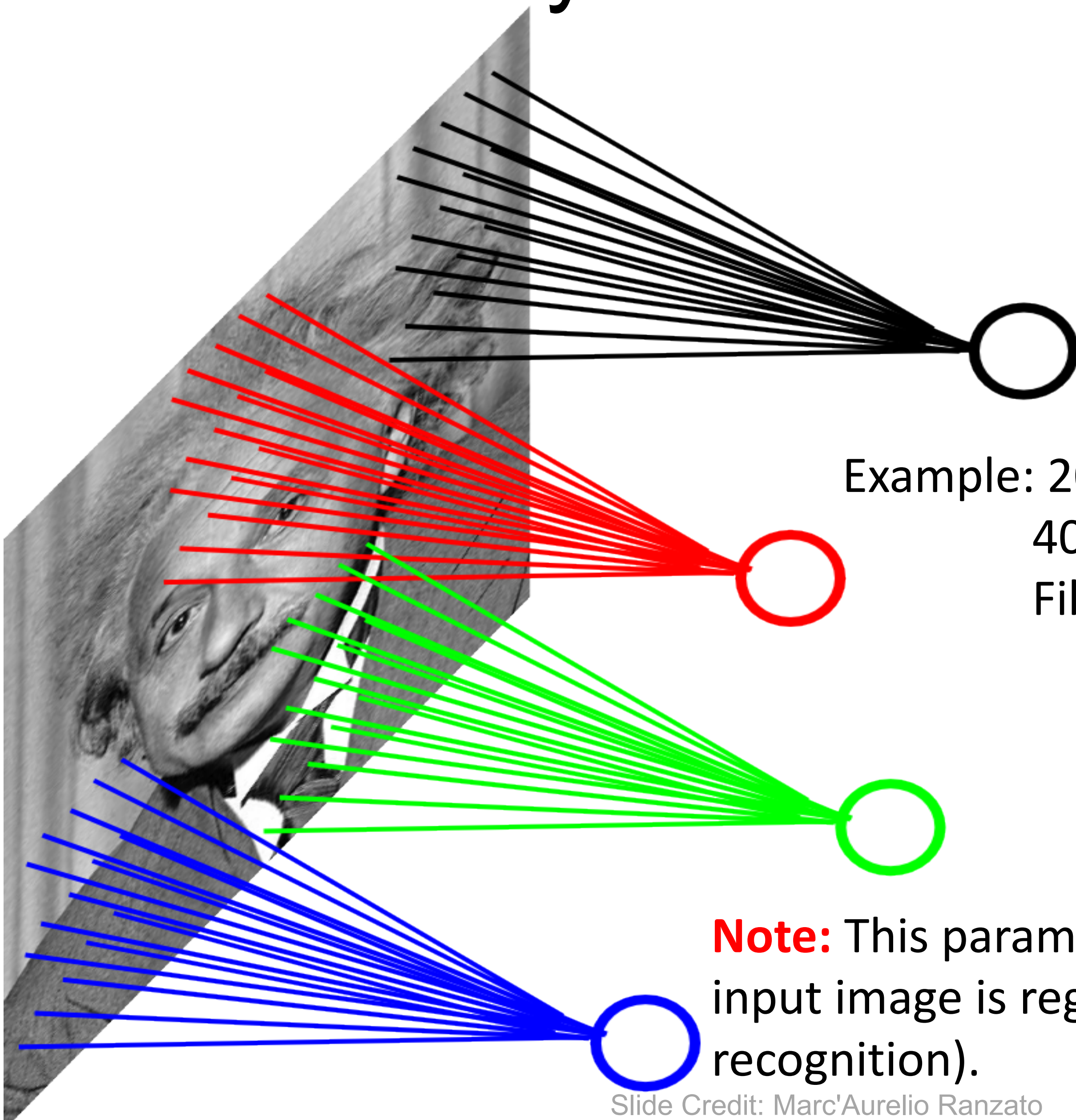


- Spatial correlation is local
- Waste of resources + we have not enough training samples anyway..



Slide Credit: Marc'Aurelio Ranzato

# Locally Connected Layer



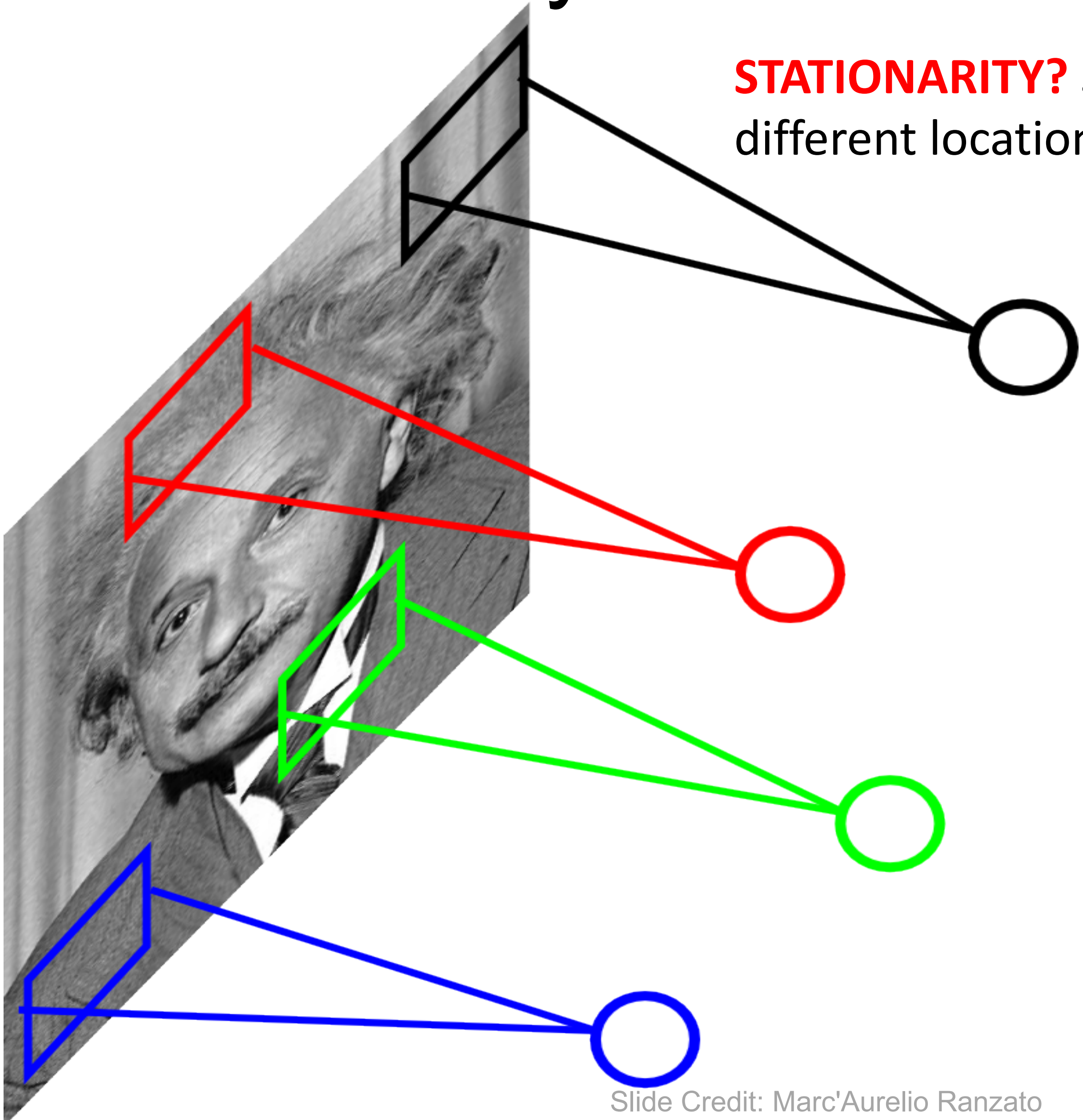
Example: 200x200 image
40K hidden units
Filter size: 10x10
4M parameters

**Note:** This parameterization is good when input image is registered (e.g., face recognition).



Slide Credit: Marc'Aurelio Ranzato

# Locally Connected Layer

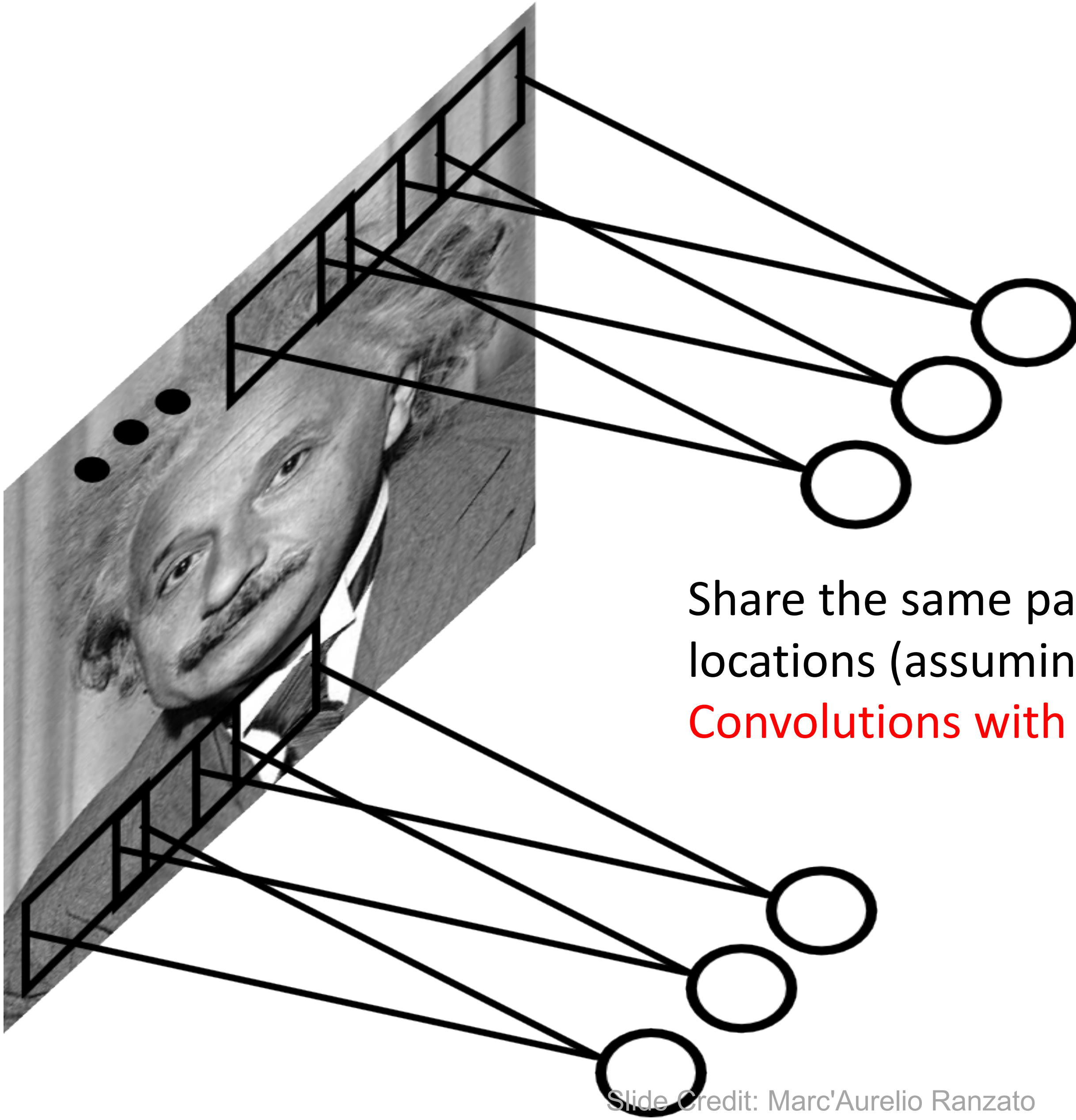


**STATIONARITY?** Statistics is similar at different locations



Slide Credit: Marc'Aurelio Ranzato

# Convolutional Layer



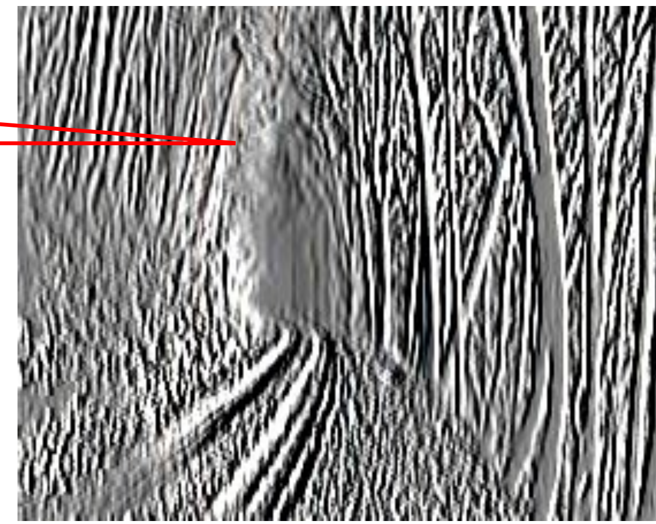
Share the same parameters across different locations (assuming input is stationary):
Convolutions with learned kernels



Slide Credit: Marc'Aurelio Ranzato

# Convolutional Layer

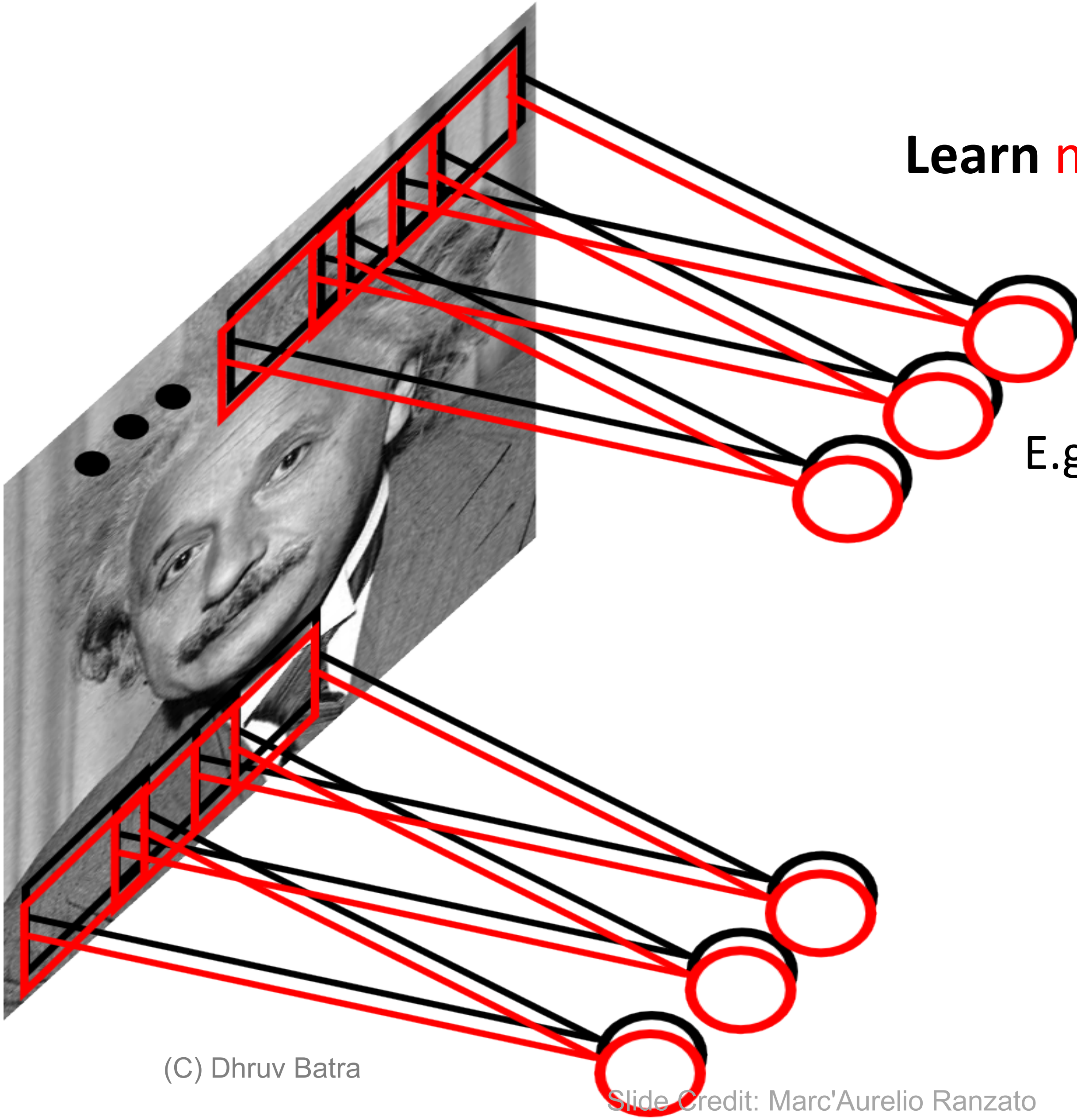
$$* \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} =$$





Slide Credit: Marc'Aurelio Ranzato

# Convolutional Layer



**Learn** multiple filters.

E.g.: 200x200 image
100 Filters
Filter size: 10x10
10K parameters

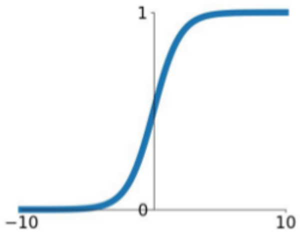


Slide Credit: Marc'Aurelio Ranzato

# Activation Functions

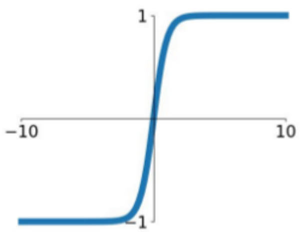
**Sigmoid**

$\sigma(x) = \frac{1}{1+e^{-x}}$



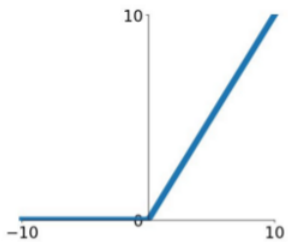

**tanh**

$\tanh(x)$



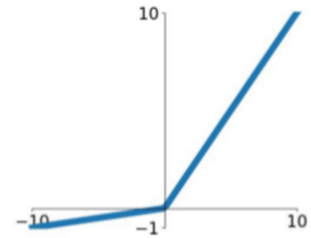

**ReLU**

$\max(0, x)$




**Leaky ReLU**

$\max(0.1x, x)$

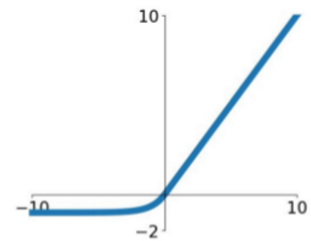



**Maxout**

$\max(w_1^T x + b_1, w_2^T x + b_2)$

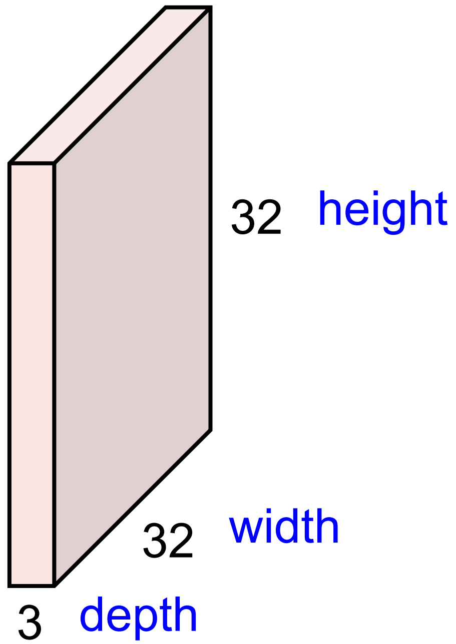
**ELU**

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

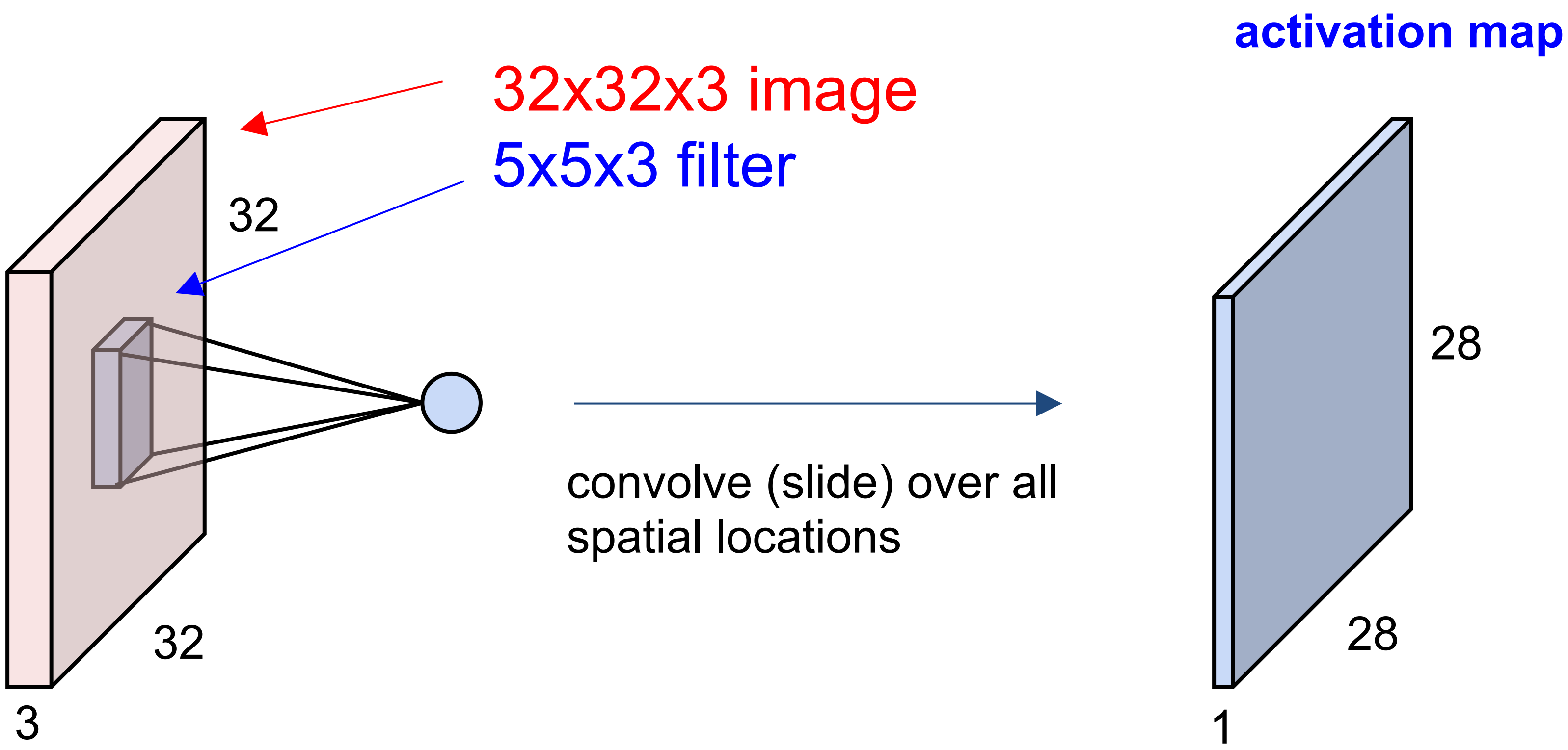


# Convolution Layer

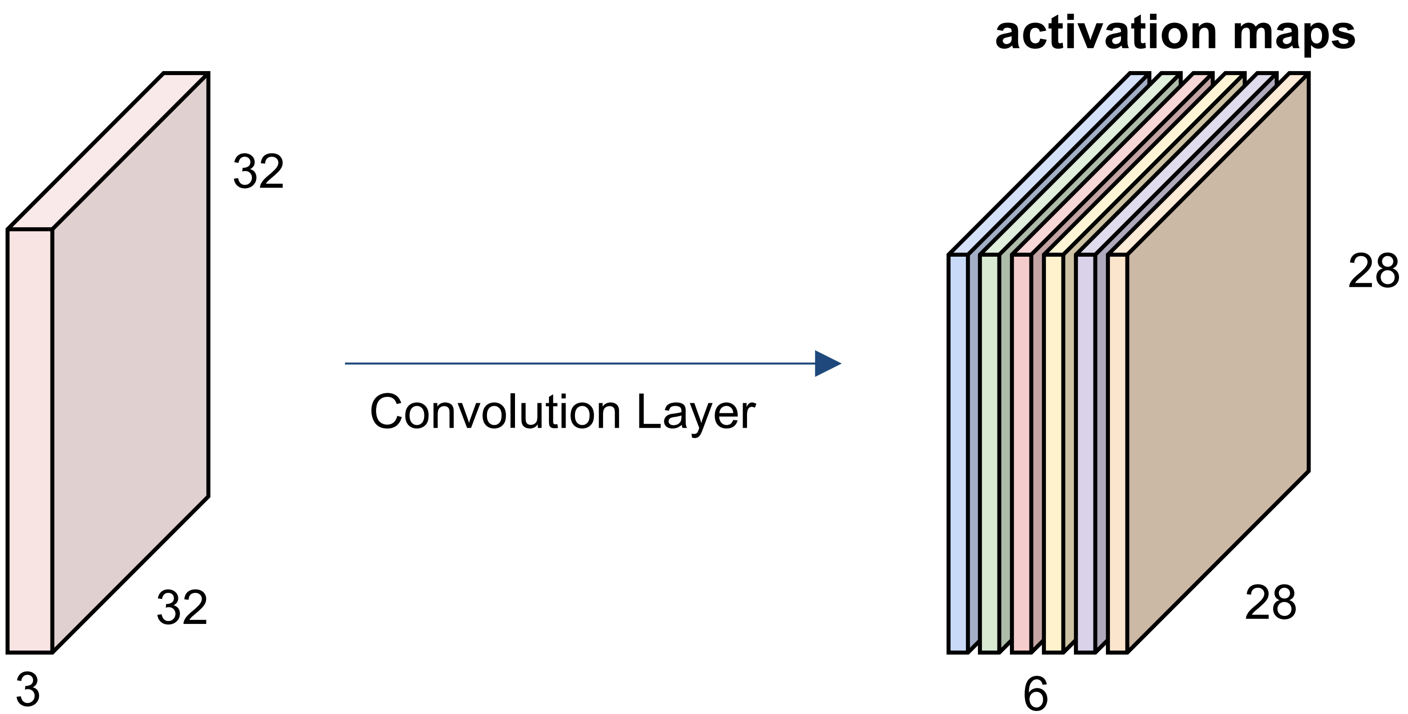
32x32x3 image -> preserve spatial structure



# Convolution Layer

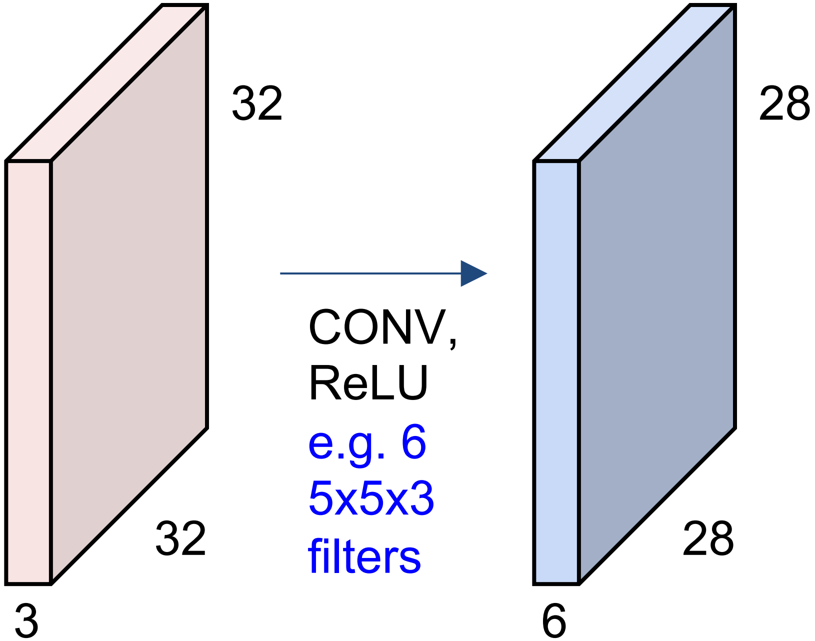


Multiple filters: if we have 6 5x5 filters, we'll get 6 separate activation maps:

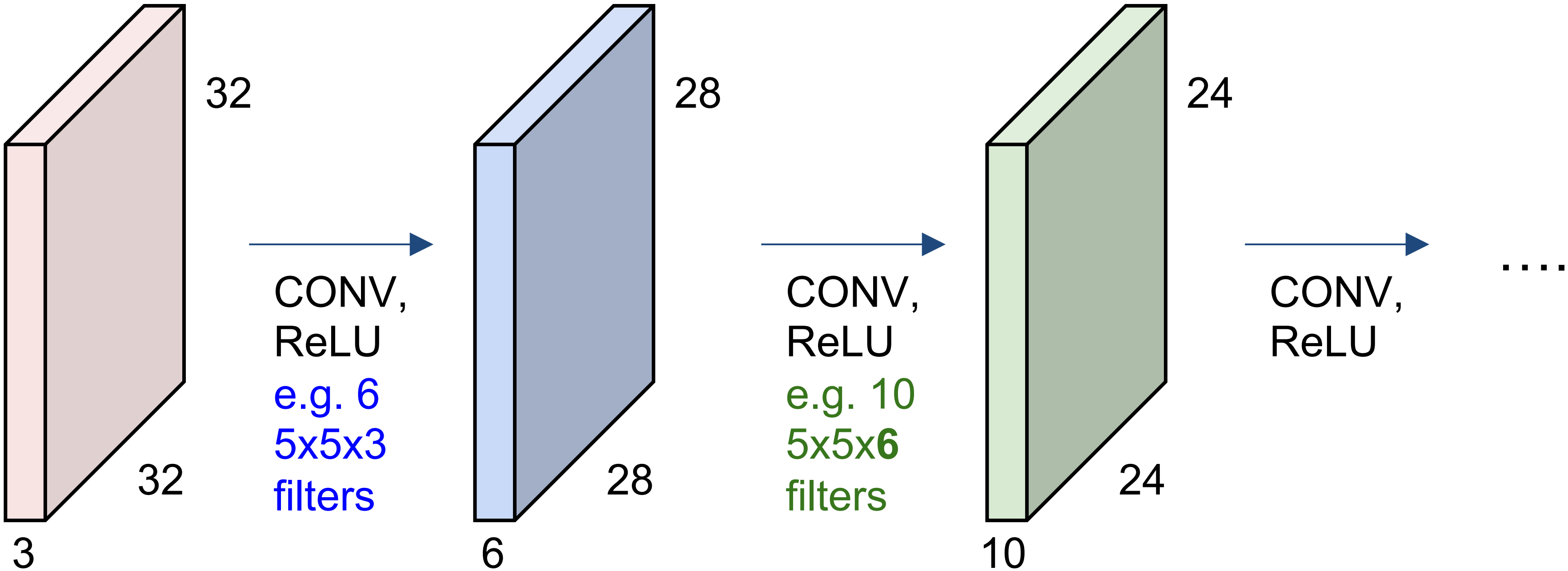



We stack these up to get a "new image" of size 28x28x6!

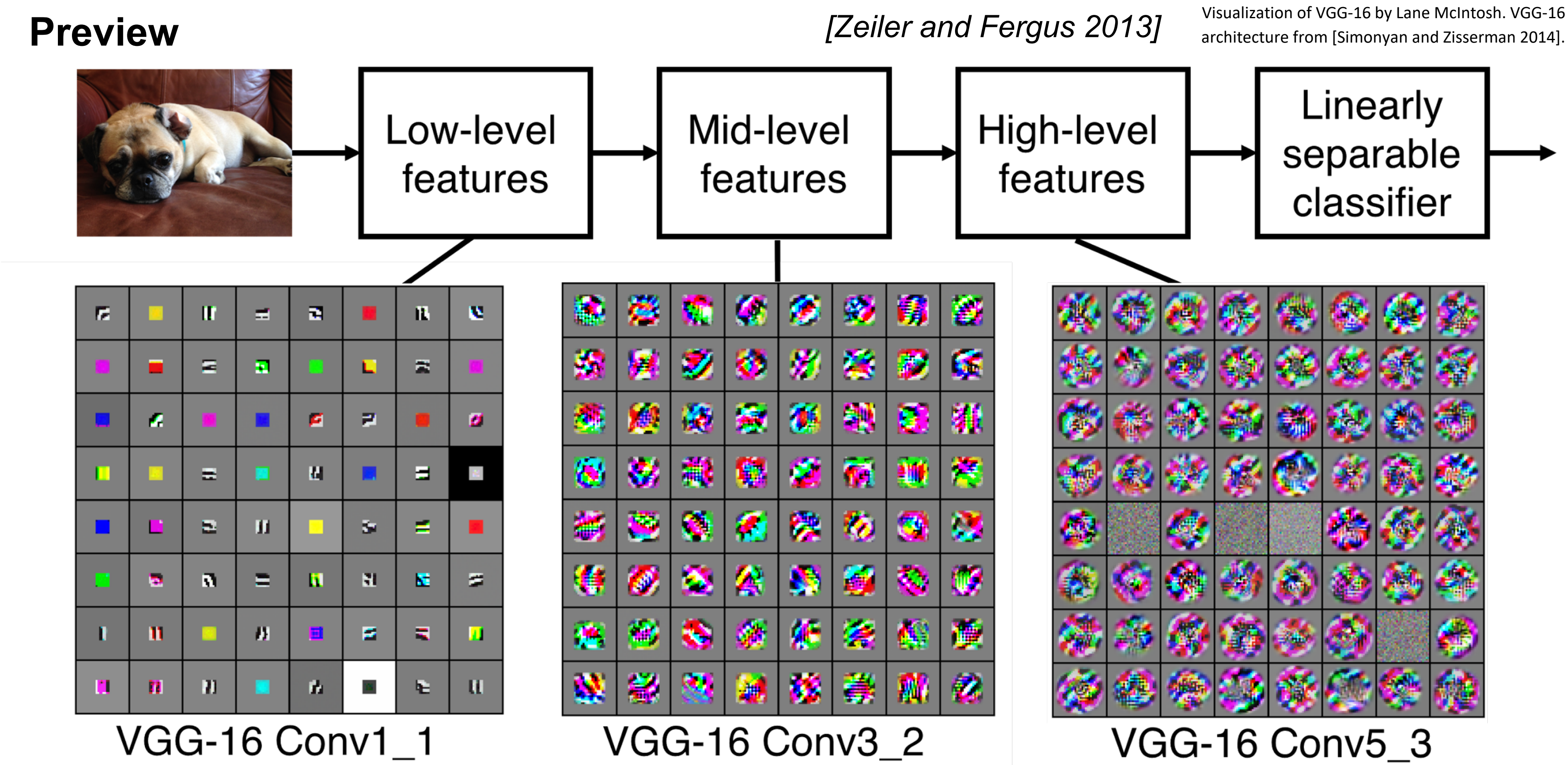
**Preview:** ConvNet is a sequence of Convolution Layers, interspersed with activation functions



**Preview:** ConvNet is a sequence of Convolutional Layers, interspersed with activation functions



## Preview


[Zeiler and Fergus 2013]
Visualization of VGG-16 by Lane McIntosh. VGG-16 architecture from [Simonyan and Zisserman 2014].
Low-level features
Mid-level features
High-level features
Linearly separable classifier
VGG-16 Conv1_1
VGG-16 Conv3_2
VGG-16 Conv5_3

# Visualizing Filters

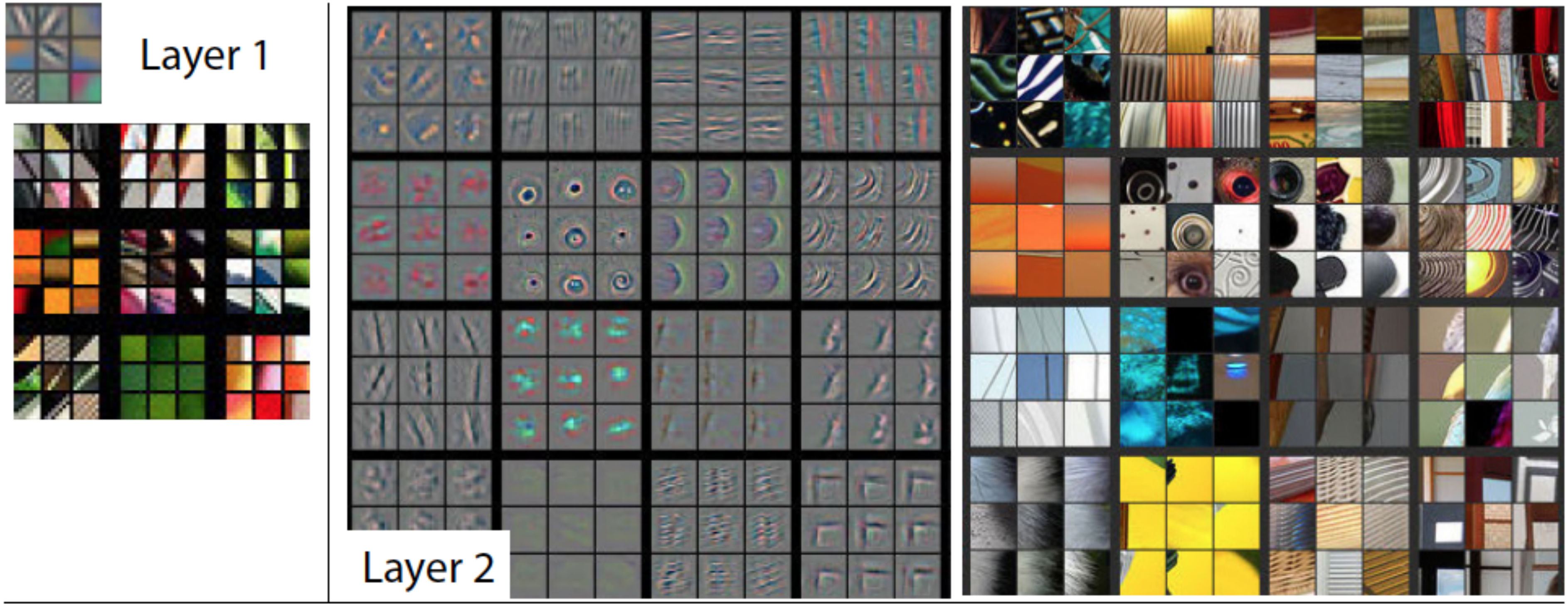





Figure Credit: [Zeiler & Fergus ECCV14]

# Visualizing Filters

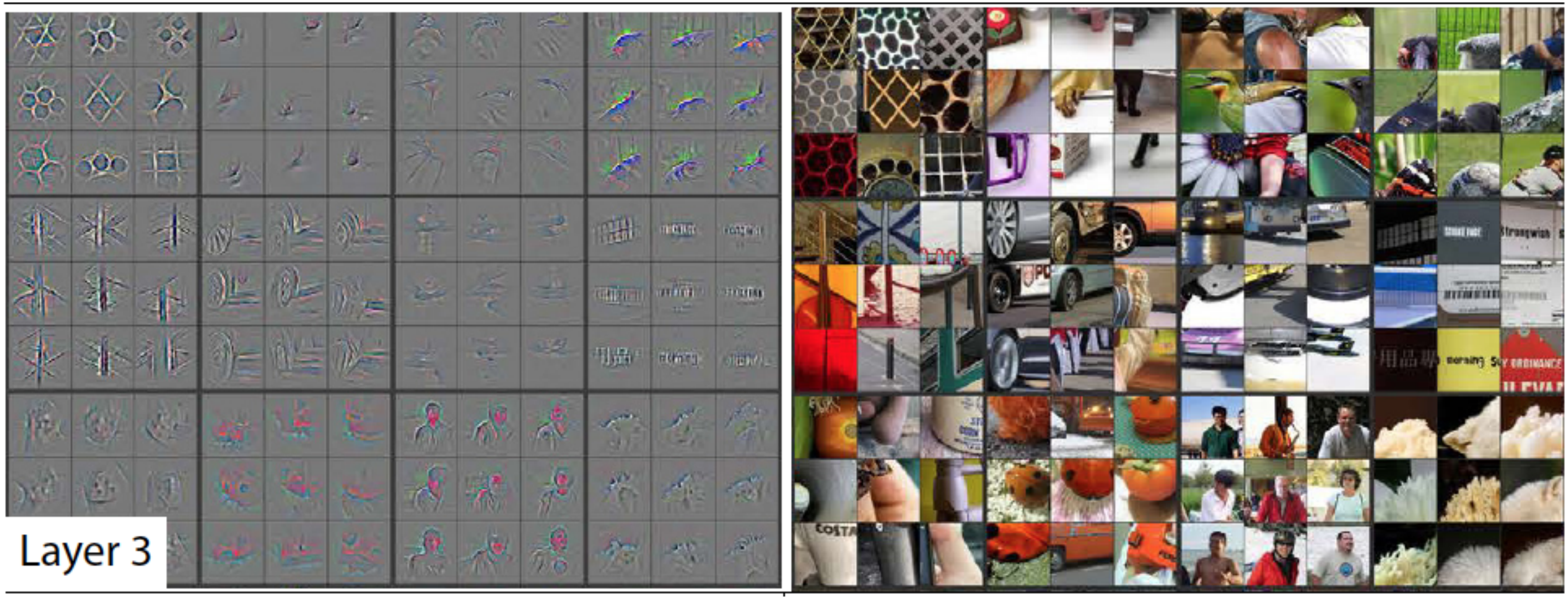





Figure Credit: [Zeiler & Fergus ECCV14]

# Visualizing Filters

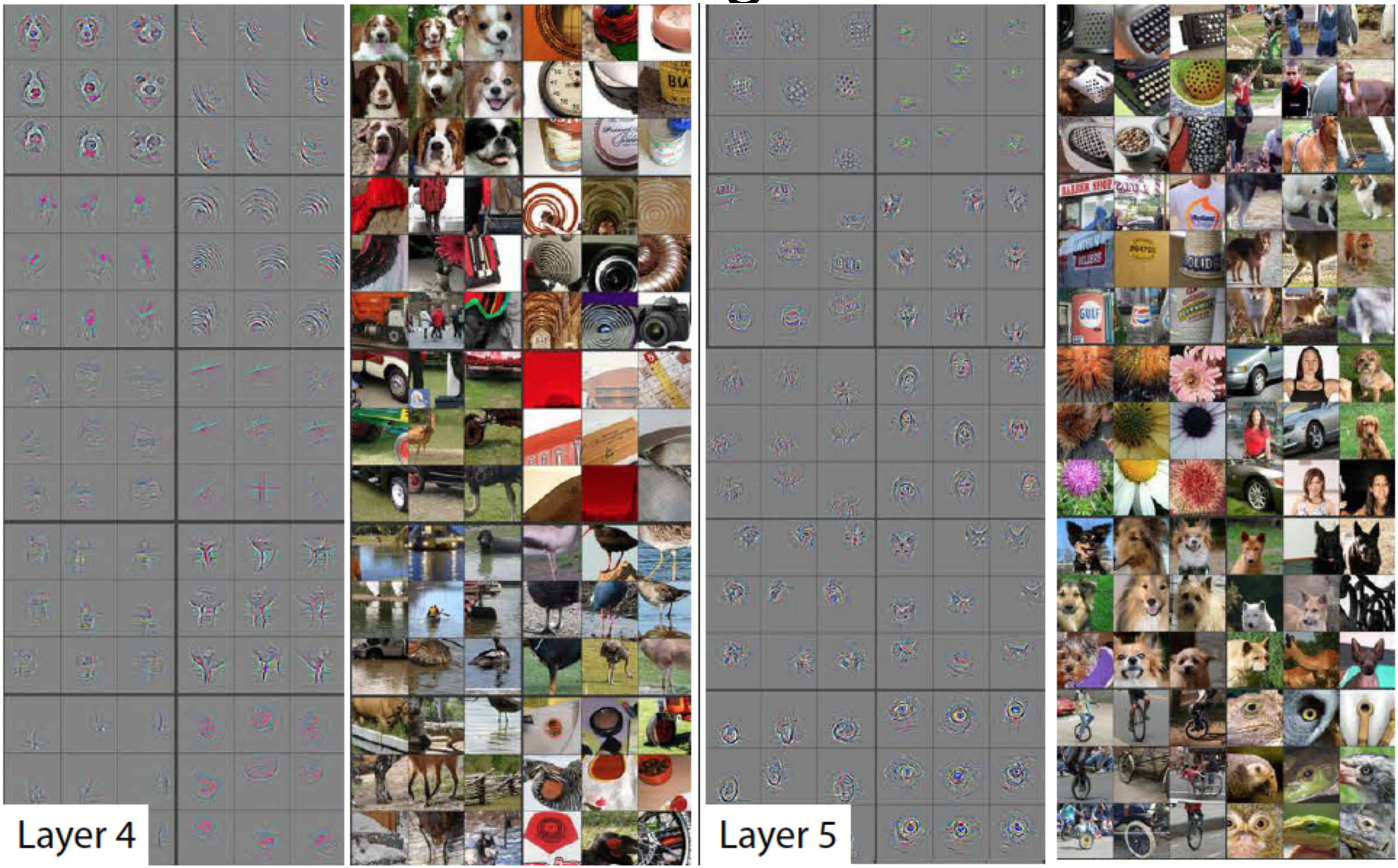


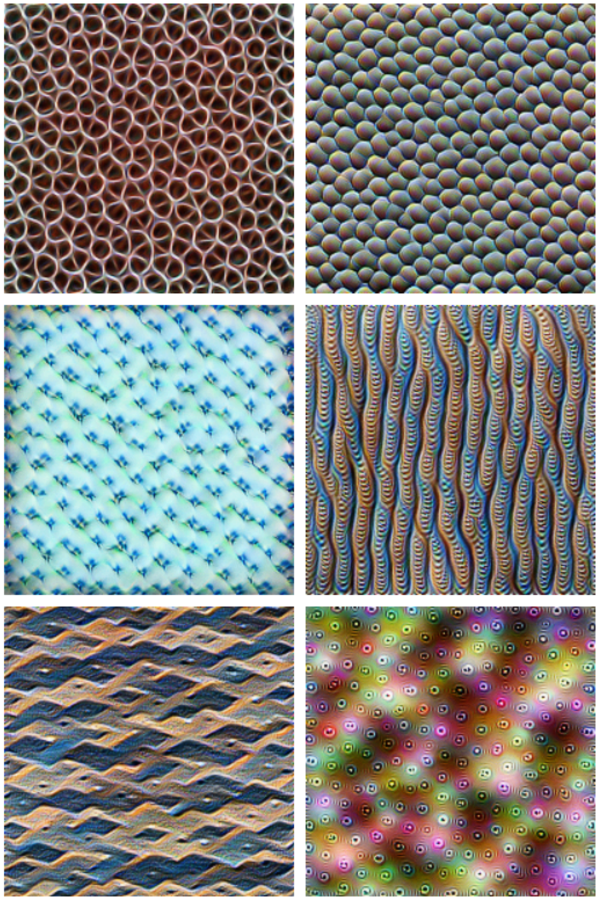



Figure Credit: [Zeiler & Fergus ECCV14]

# Distill Interactive Visualization

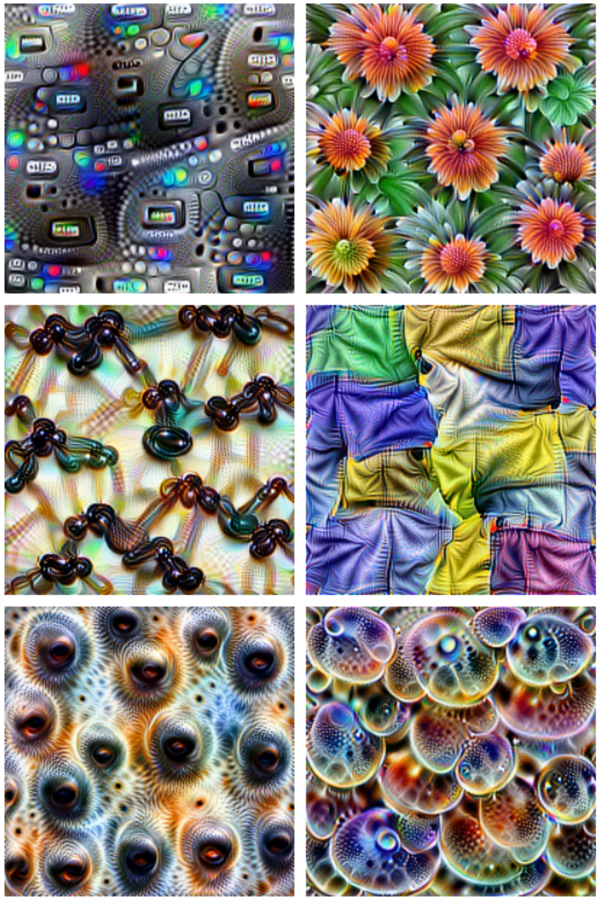
## Feature Visualization

How neural networks build up their understanding of images



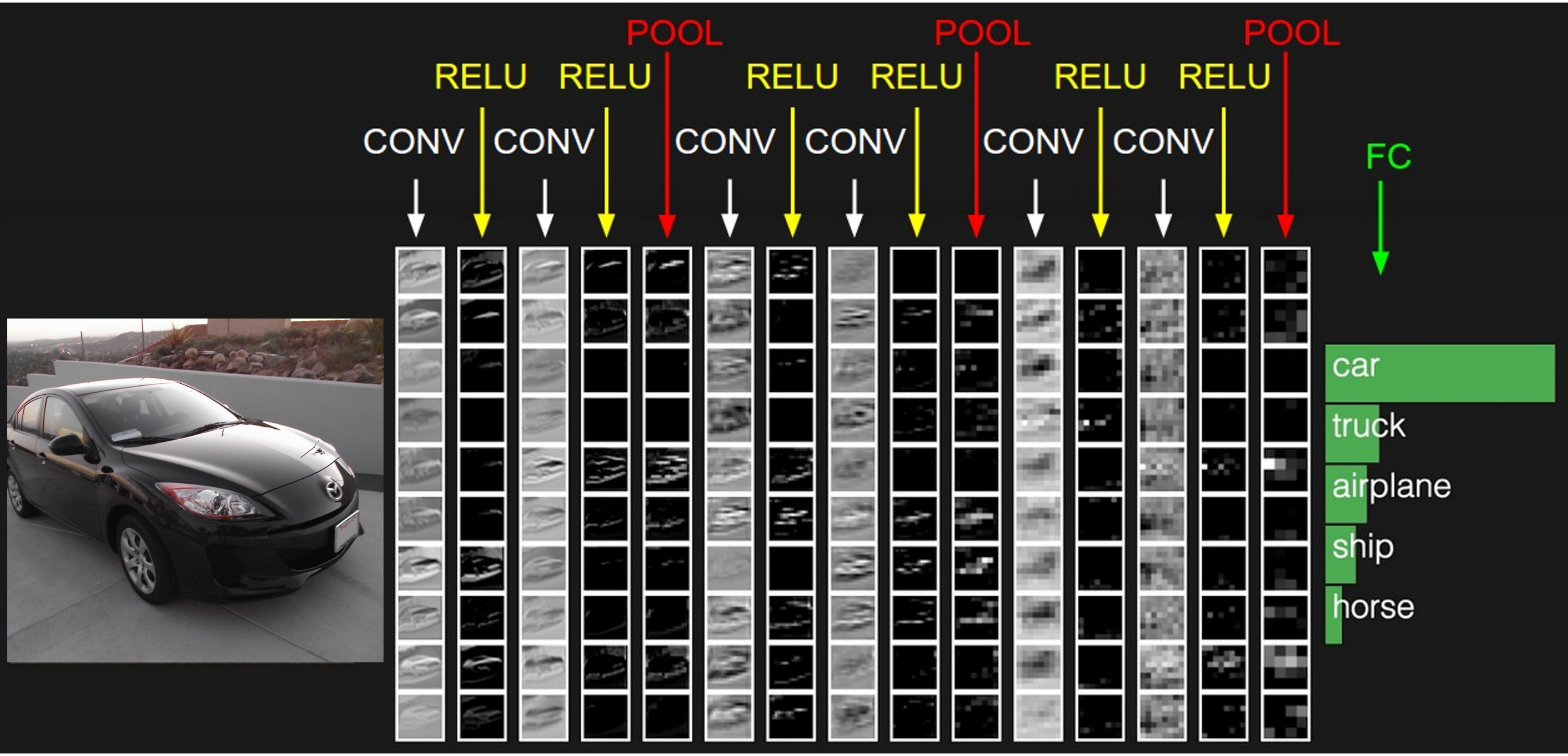
**Textures** (layer mixed3a)

**Patterns** (layer mixed4a)



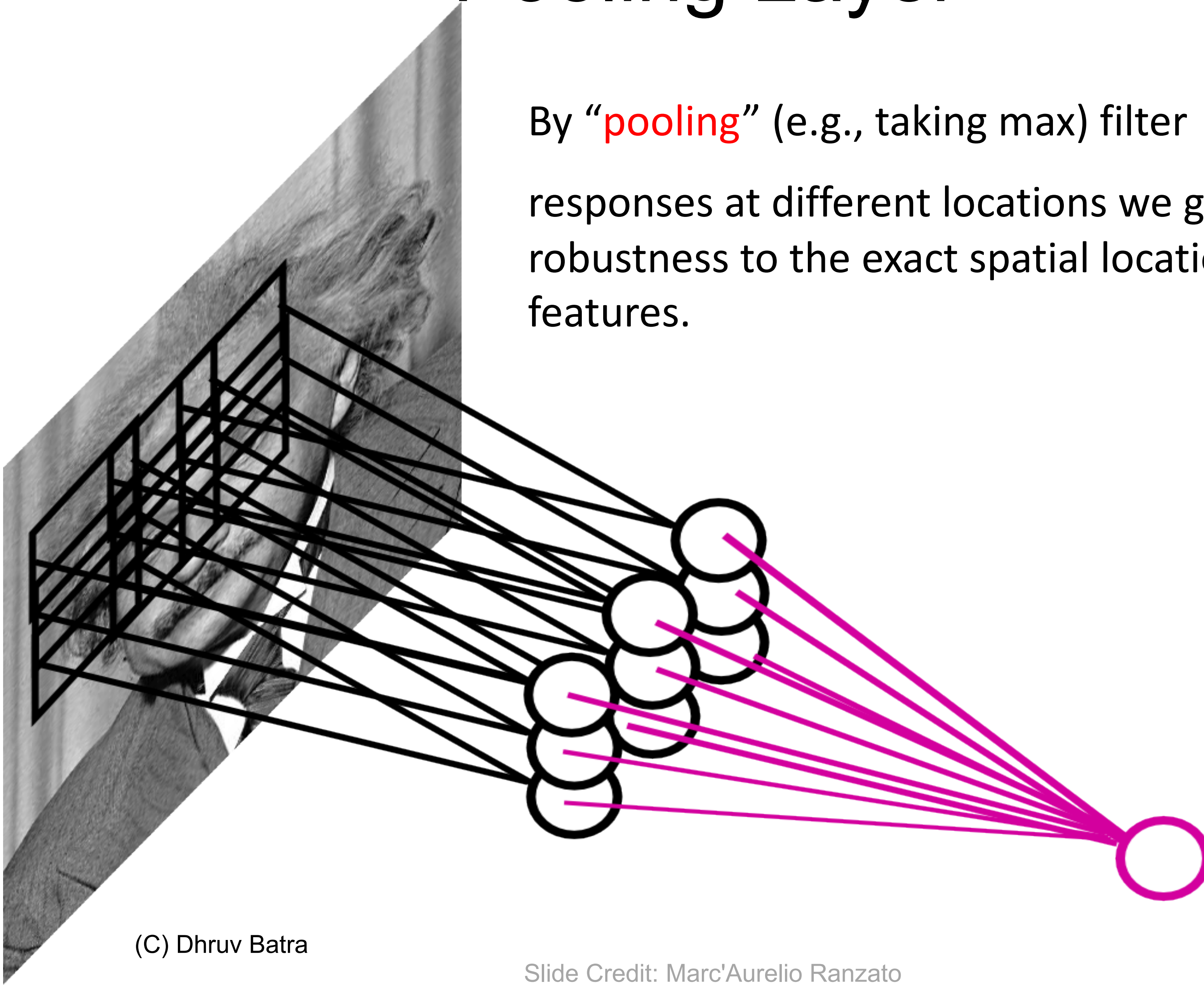
**Parts** (layers mixed4b & mixed4c)

two more layers to go: POOL/FC




Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

# Pooling Layer



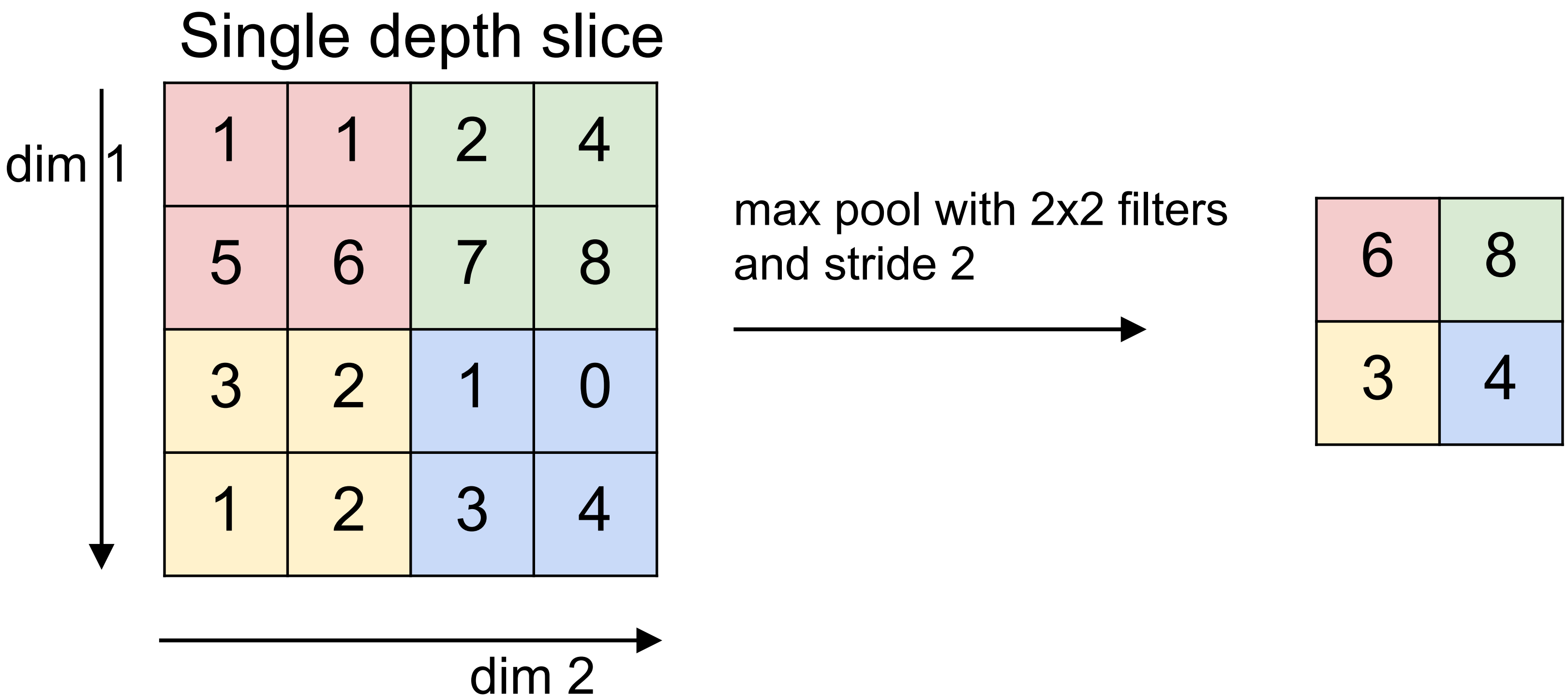
By "pooling" (e.g., taking max) filter

responses at different locations we gain robustness to the exact spatial location of features.



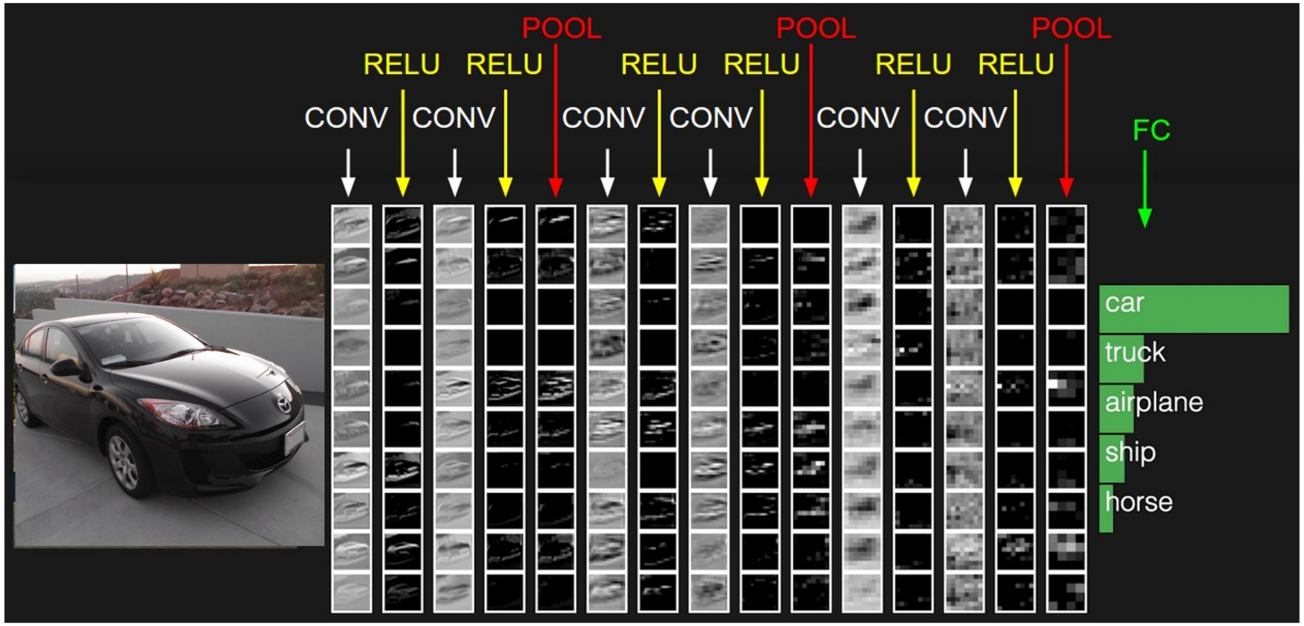
Slide Credit: Marc'Aurelio Ranzato

# MAX POOLING

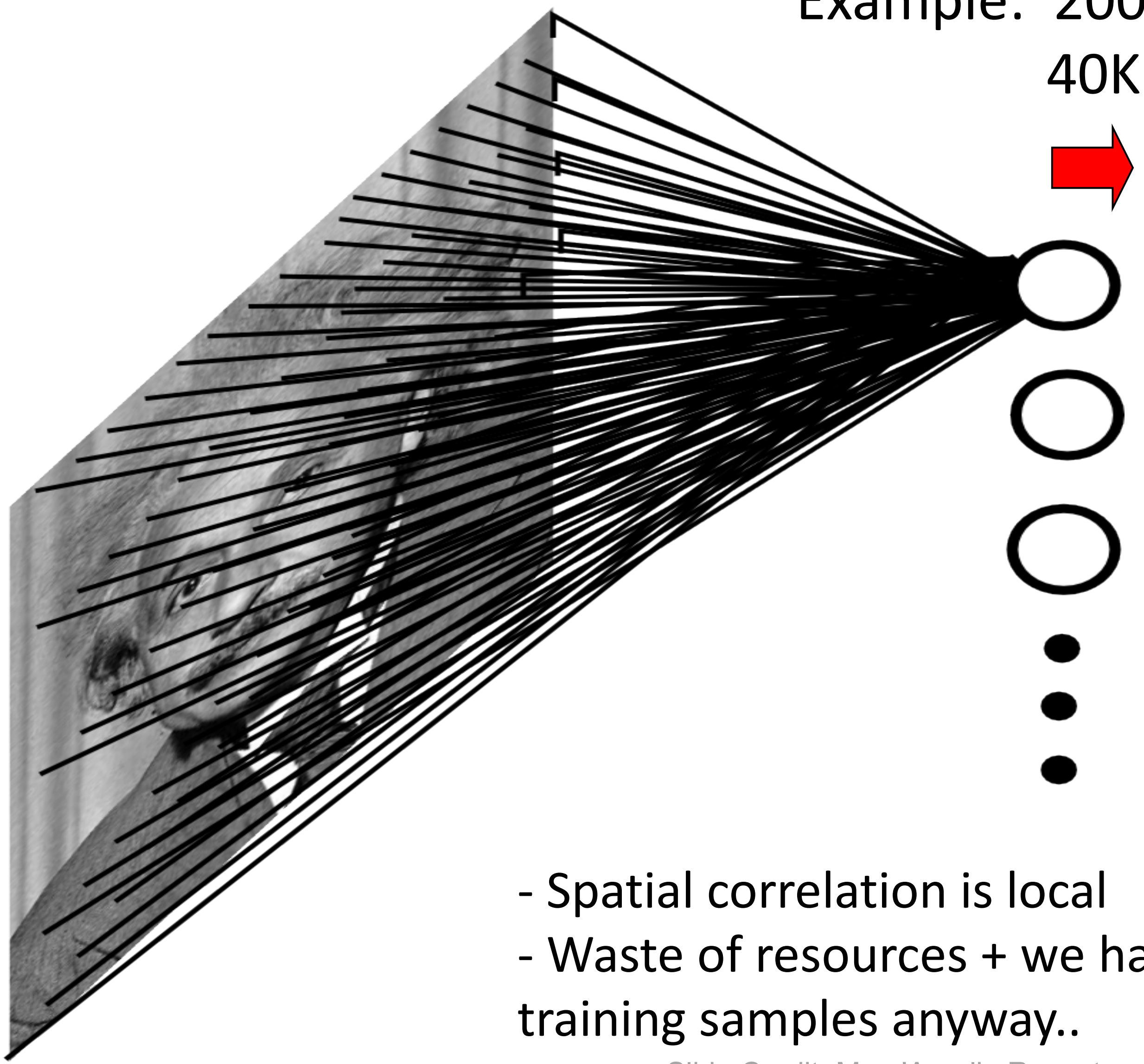

Single depth slice
dim 1
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
dim 2
max pool with 2x2 filters
and stride 2
6 8
3 4

# Fully Connected Layer (FC layer)

- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks



# Fully Connected Layer



Example: 200x200 image
40K hidden units
**~2B parameters**!!!

- Spatial correlation is local
- Waste of resources + we have not enough training samples anyway..

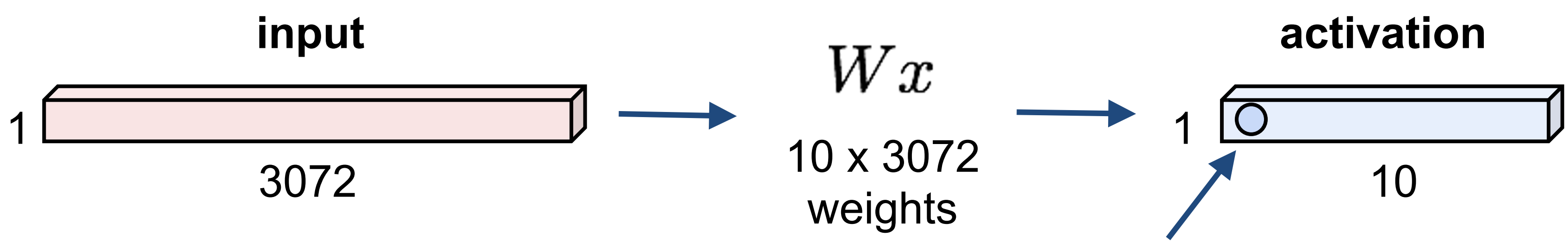


Slide Credit: Marc'Aurelio Ranzato

# Fully Connected Layer

Each neuron looks at the full input volume

32x32x3 image -> stretch to 3072 x 1




**1 number:**
the result of taking a dot product between a row of W and the input (a 3072-dimensional dot product)

# CNNs for Image Processing

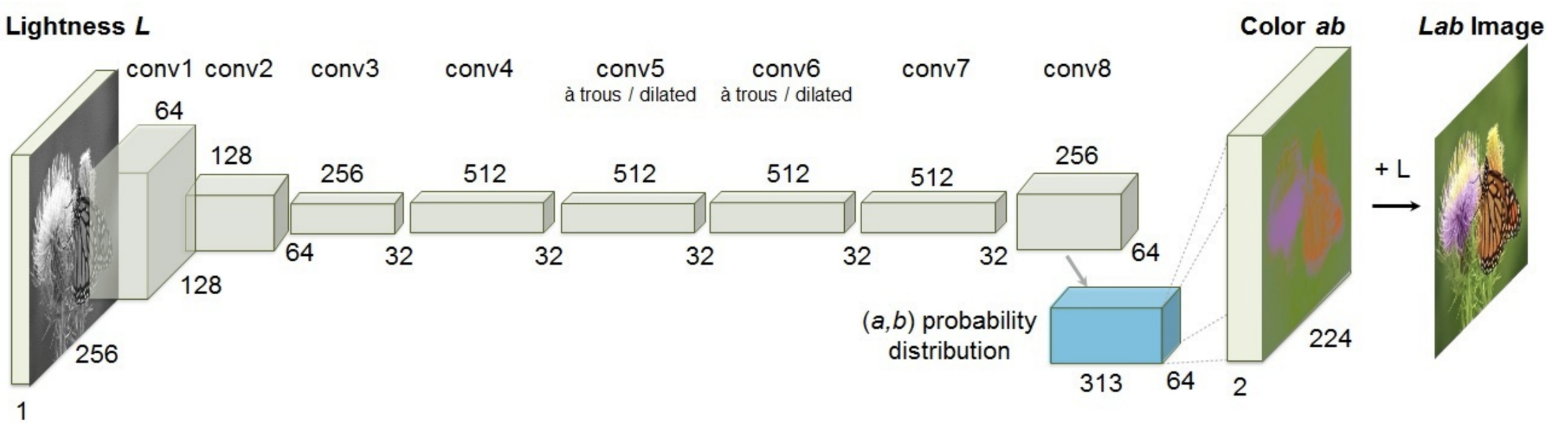
# Colorization

- Given a grayscale image, colorize the image realistically
- Zhang et al. pose colorization as classification task and use class-rebalancing to improve results
- Demonstrate higher rates of fooling humans using "colorization Turing test"

*Colorful Image Colorization*. Richard Zhang, Phillip Isola, Alexei A. Efros. ECCV 2016.

# Colorization

- Training data: decompose any RGB image into L*a*b color space
  - *L*: grayscale input (lightness channel)
  - *ab*: color channels
- Train CNN with ***one million color images*** and a new objective function to incorporate more diverse colors. Many possible correct colorizations!




*Colorful Image Colorization*. Richard Zhang, Phillip Isola, Alexei A. Efros. ECCV 2016.

# How to convert the inferred distribution to an image?

- 313-way classification over discretized ab color bins
- Network will output a distribution *z* over colors at each pixel. Need to convert to a single pixel value
    - Mode: vibrant but sometimes spatially inconsistent (e.g., the red splotches on the bus)
    - Mean: produces spatially consistent but desaturated results, exhibiting an unnatural sepia tone


$$\mathcal{H}(\mathbf{Z}_{h,w}) = \mathbb{E}\big[f_T(\mathbf{Z}_{h,w})\big], \quad f_T(\mathbf{z}) = \frac{\exp(\log(\mathbf{z})/T)}{\sum_q \exp(\log(\mathbf{z}_q)/T)}$$

# DeOldify

https://github.com/jantic/DeOldify

# Super-Resolution

Low resolution

High resolution

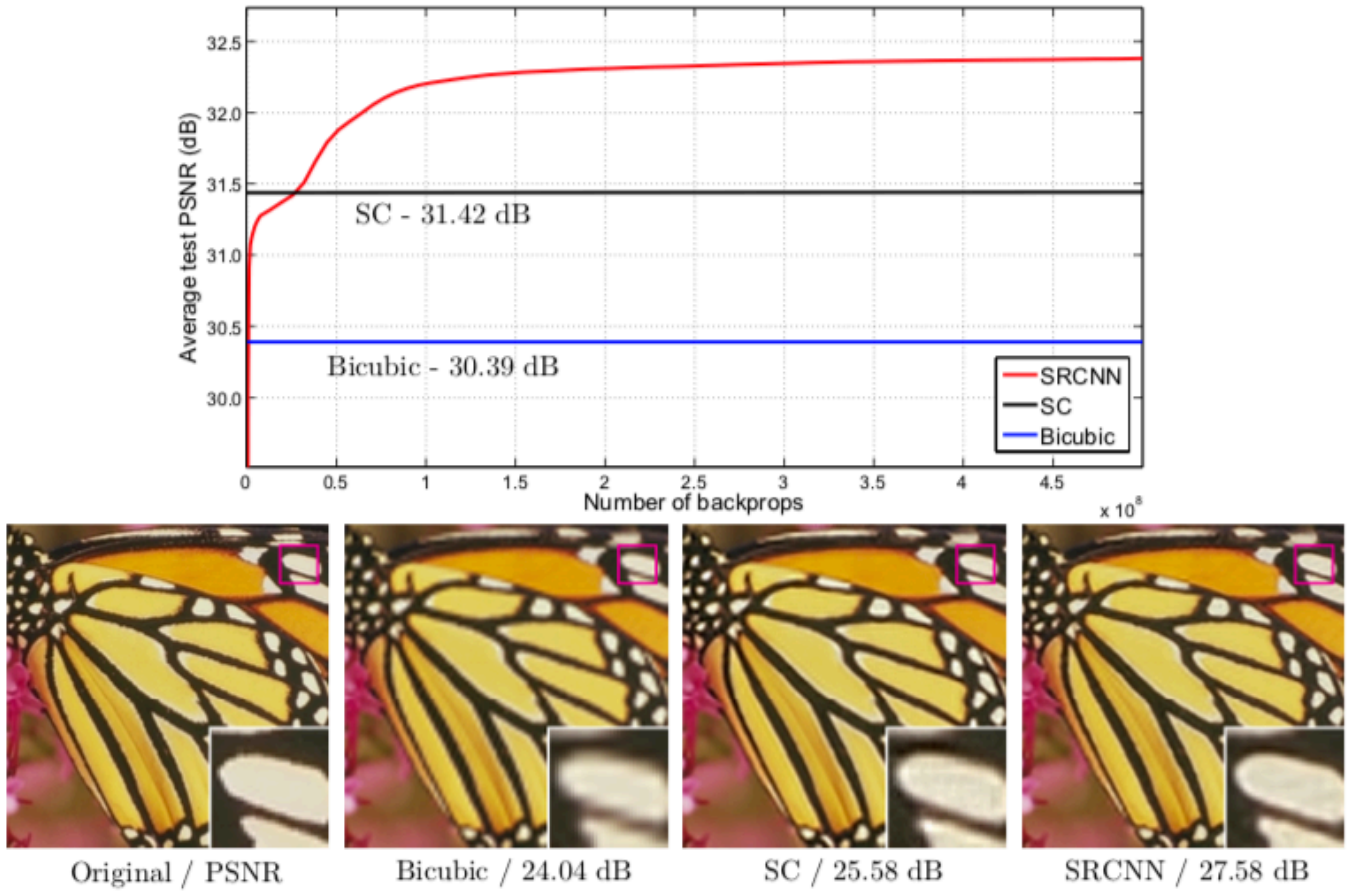
# Super-Resolution as a task

- Quality-degrading factors / sources of noise:
  - Camera shake, shadows, motion blur, radial distortion from fisheye/GoPro type cameras, poor contrast, poor lighting, lossy compression, transmission defects, dust, haze, smoke, and mist, motion of the camera sensor platform, moving objects captured within the observed scene, e.g. people and vehicles.
- How to measure super-resolution?
  - Peak signal-to-noise ratio (PSNR), higher is better. Relies upon the Mean Square Error (MSE) error metric to evaluate image compression quality between two images:

$$MSE = \frac{1}{MN} \sum_{M} \sum_{N} [I_1(m,n) - I_2(m,n)]^2 = \|I_1 - I_2\|_F$$

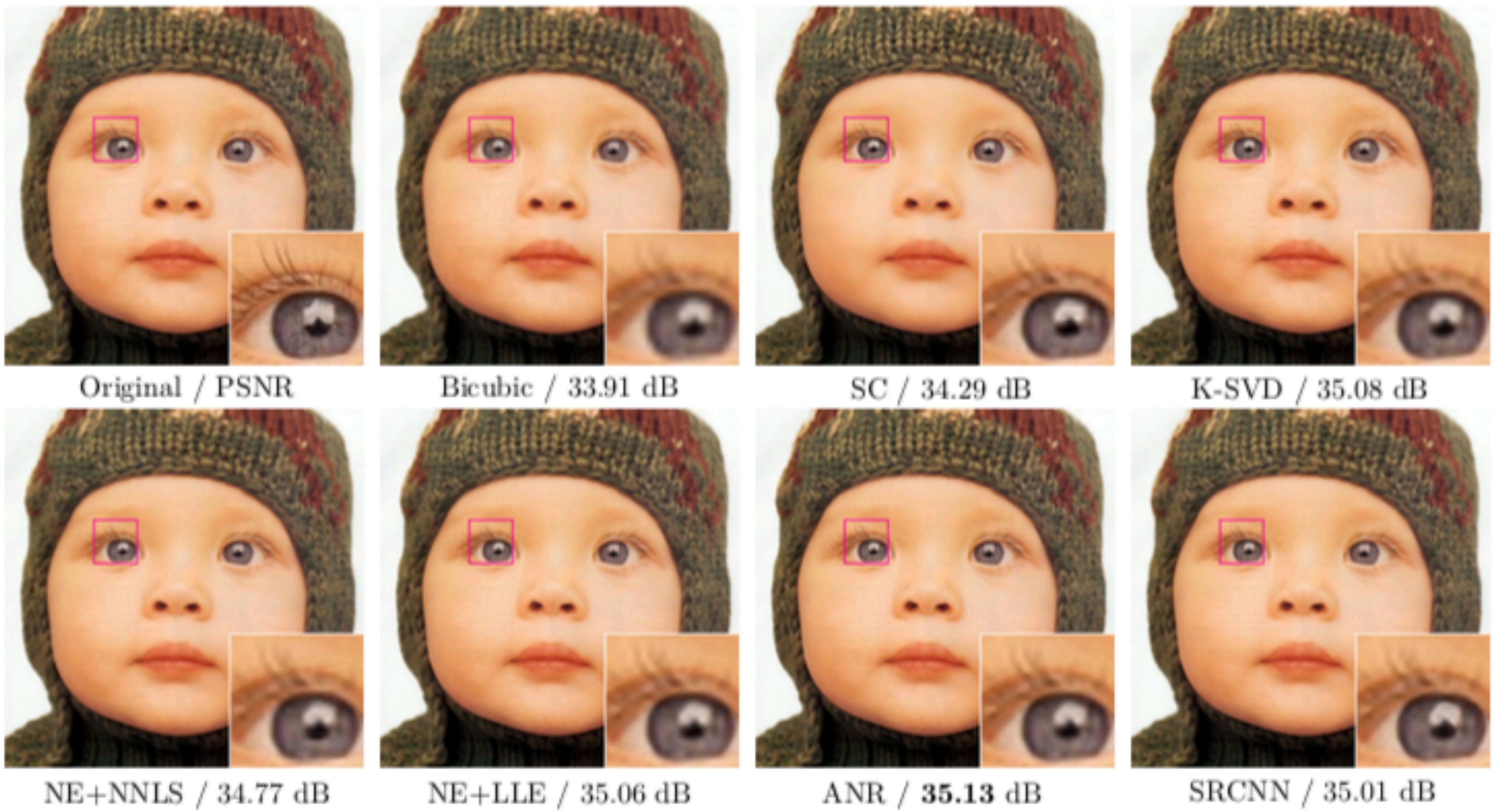
$$PSNR = 10 \log_{10}\left(\frac{R^2}{MSE}\right)$$

# An early CNN paper (2016)




Dong, Chao, et al. "Learning a deep convolutional network for image super-resolution." *European conference on computer vision*. Springer, Cham, 2014.

# An early CNN paper (2016)



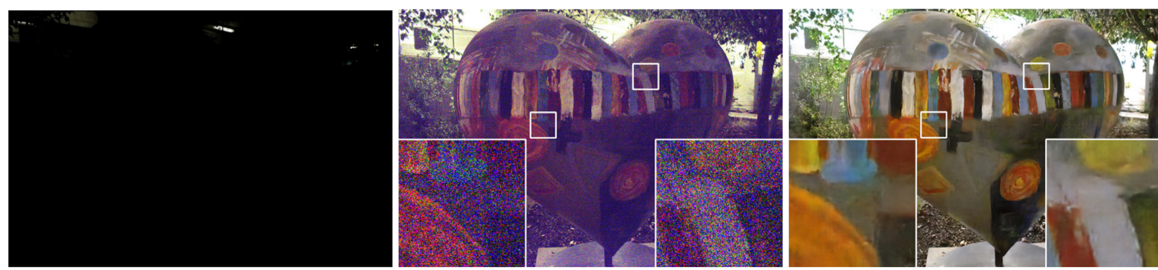

Upscaling factor of 3 !

Dong, Chao, et al. "Learning a deep convolutional network for image super-resolution." *European conference on computer vision*. Springer, Cham, 2014.

# Underexposed Photo Enhancement

- Goal: enhance extreme low-light imaging with severely limited illumination (e.g., moonlight) and short exposure (exposure time is set to 1/30 second)
- The less light there is, the more ISO you need
    - High ISO can be used to increase brightness, but amplifies noise
    - Leads to low signal-to-noise ratio (SNR) due to low photon counts


(a) Camera output with ISO 8,000 (b) Camera output with ISO 409,600 (c) Our result from the raw data of (a)



(a) JPEG image produced by camera (b) Raw data via traditional pipeline (c) Our result

*Learning to See in the Dark*. Qifeng Chen, Vladlen Koltun. CVPR 2018.

# Solution? Collect dataset and train a deep network

- See-in-the-Dark (SID) dataset contains 5094 raw short exposure images, each with a corresponding long-exposure reference image
- Corresponding reference (ground truth) images captured with 100-300x longer exposure (i.e. 10 to 30 seconds)
- Overcome low photon counts!
- Train deep neural networks to learn the image processing pipeline w/ L1 loss.

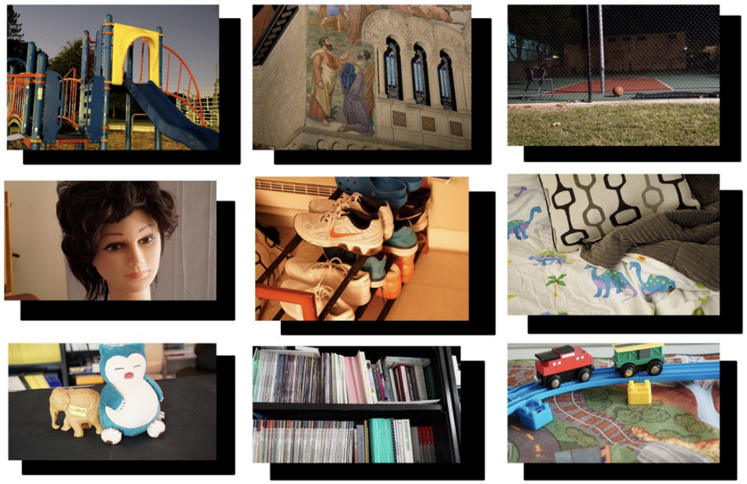


Figure 2. Example images in the SID dataset. Outdoor images in the top two rows, indoor images in the bottom rows. Long-exposure reference (ground truth) images are shown in front. Short-exposure input images (essentially black) are shown in the back. The illuminance at the camera is generally between 0.2 and 5 lux outdoors and between 0.03 and 0.3 lux indoors.

*Learning to See in the Dark*. Qifeng Chen, Vladlen Koltun. CVPR 2018.

# Underexposed Photo Enhancement

- Learn image-to-image mapping? Too hard!
- Instead estimate an image-to-illumination mapping (model varying-lighting conditions)
  - Illumination maps for natural images typically have relatively simple forms with known priors
- Then take illumination map to light up the underexposed photo.
- Minimize (reconstruction loss + smoothness loss + color loss)

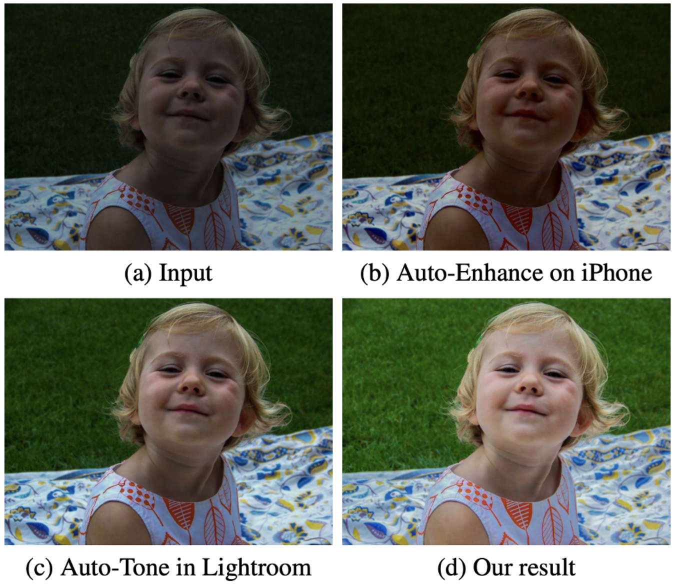
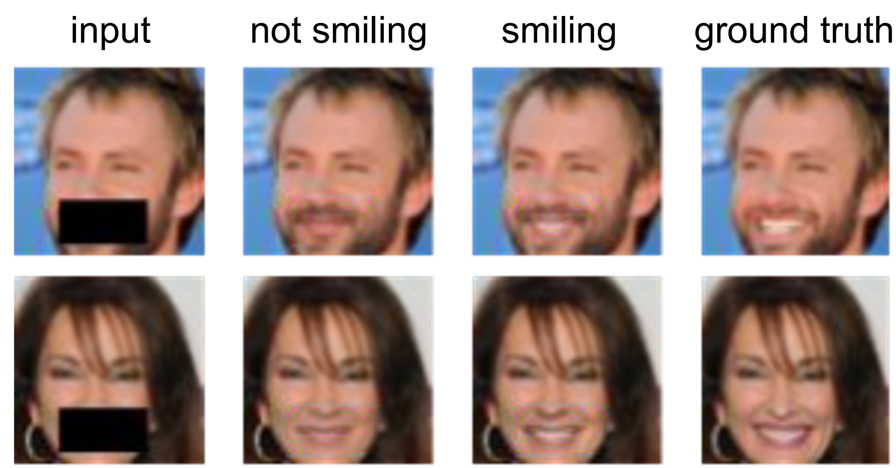



Figure 1: A challenging underexposed photo (a) enhanced by various tools (b)-(d). Our result contains more details, distinct contrast, and more natural color.

*Underexposed Photo Enhancement Using Deep Illumination Estimation*. Wang et al. CVPR 2019.

# Image Inpainting

- Perceptual loss is added to ELBO, the typical objective function used in variational autoencoders, to increase the sharpness and overall quality of inpainted images
- Demonstrate results on attribute-guided image completion



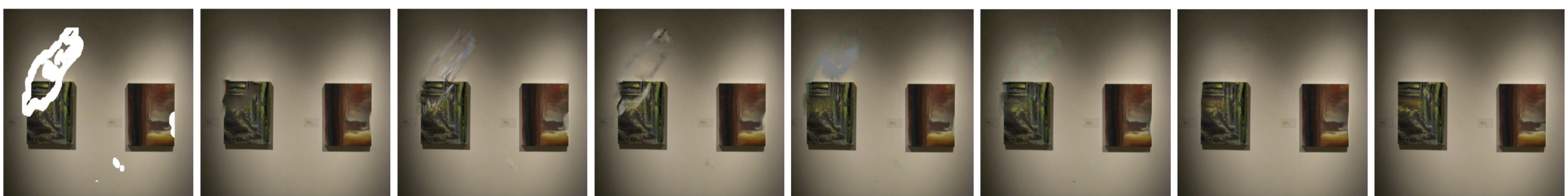

$$\mathcal{L}_{recon} = \|x_{gen} - x_{gt}\|^2 + \sum_l \lambda_l \|\eta_l(x_{gen}) - \eta_l(x_{gt})\|^2$$

$x_{gen}$ : generated image
$x_{gt}$ : ground truth image
$\eta_l$ : activation of the $l^{th}$ layer of a pre-trained VGG

*Variational Image Inpainting*. Cusuh Ham*, Amit Raj*, Vincent Cartillier*, Irfan Essa. NeuRIPS 2018 Workshop.

# Image Inpainting

- Proposes partial convolutions, comprised of a masked & re-normalized convolution operator
- Updates mask automatically after partial convolutions, removing any masking where partial convolution was able to operate on unmasked value



(a) Image with hole (b) PatchMatch (c) Iizuka et al.[10] (d) Yu et al.[38] (e) Hole=127.5 (f) Hole=IN_Mean (g) Partial Conv (h) Ground Truth

*Image Inpainting for Irregular Holes Using Partial Convolutions*. Liu et al. ECCV 2018.