Convolutional Neural Networks I

Semester 2, 2025

Kris Ehinger

Outline

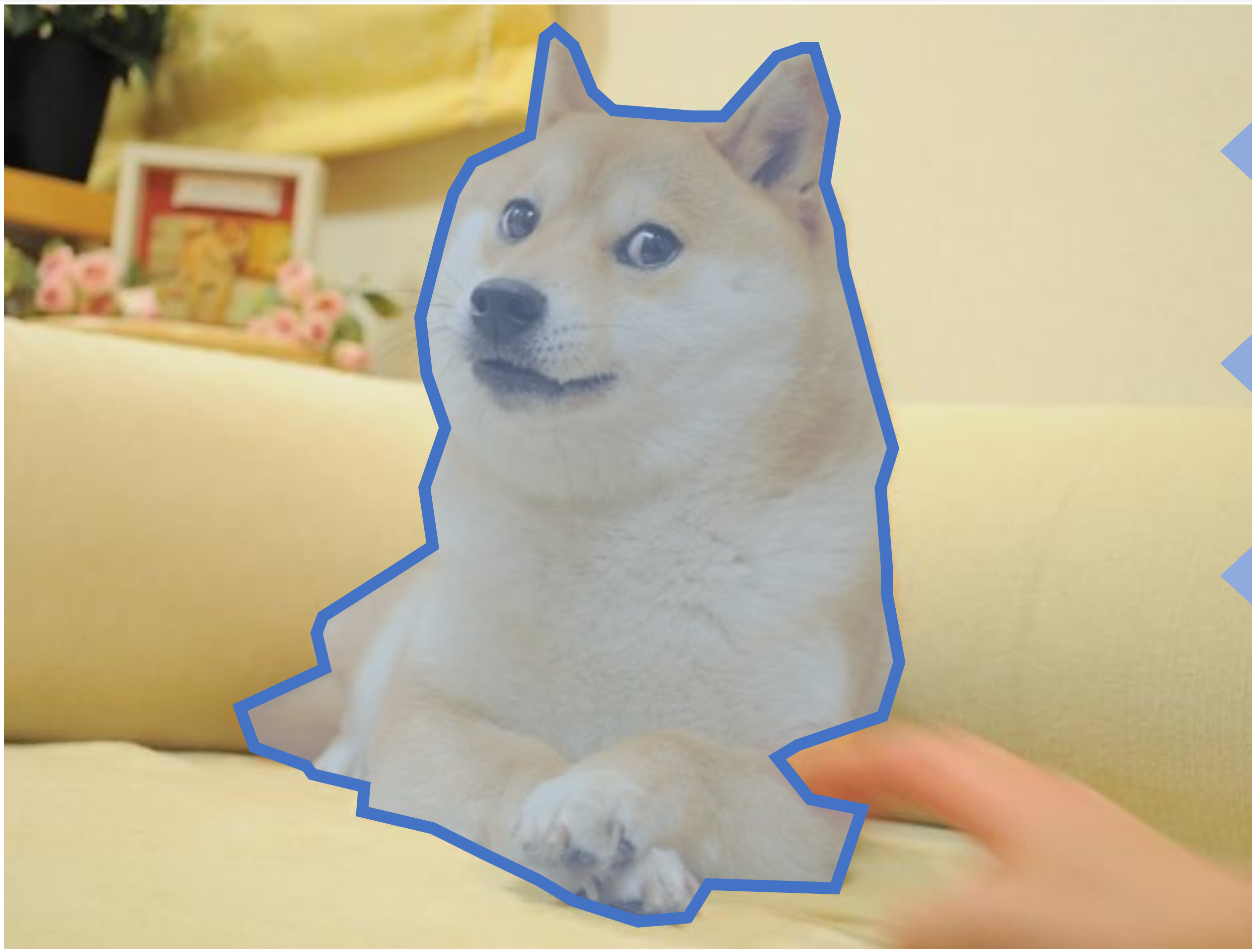
- Image recognition
- Review of neural networks
- Convolutional neural networks

Learning outcomes

- Define image recognition as a computer vision problem
- Explain the differences between convolutional and fully-connected networks
- Implement convolutional layers

Image recognition

What is “recognition”?



“dog”

“shiba inu”

“Doge”

Image: https://kabochan.blog.jp/archives/9733755.html

What is “recognition”?

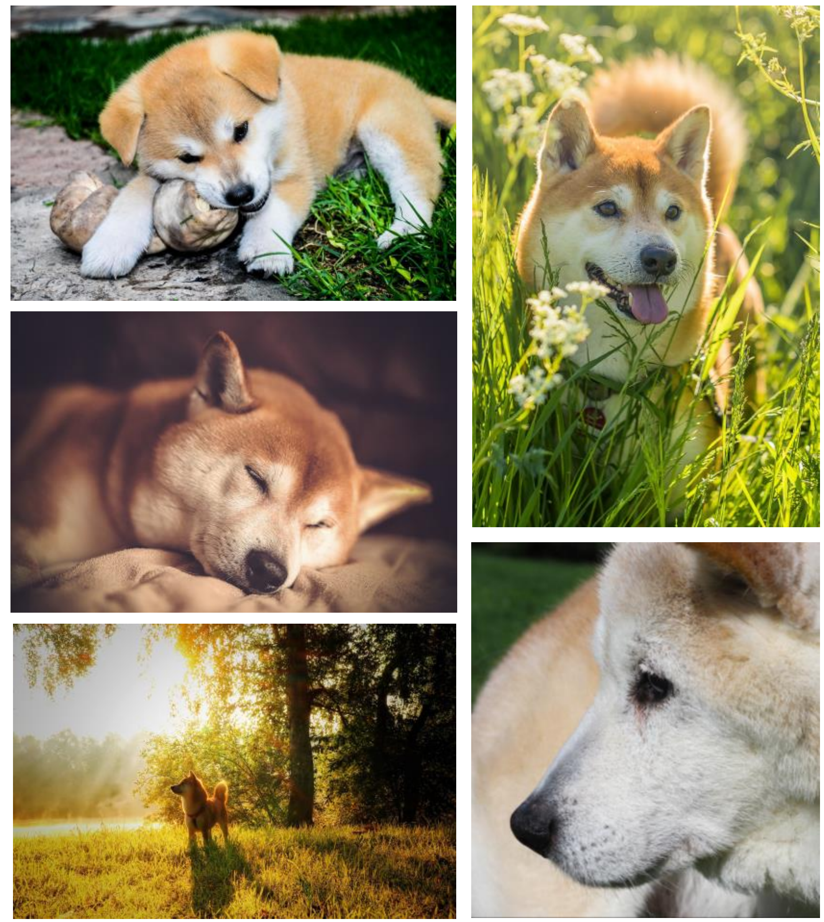
- In this section, we’ll define **image recognition** as category-level recognition of the whole image
- Category-level = group level
  - Groups may be more or less specific (“bird,” “duck,” “Australian wood duck”)
  - Different from **instance-level recognition**, recognising a specific individual
- Whole image = one label per image
  - Different from **detection** = locate object in image
  - Different from **segmentation** = label individual pixels

Why is recognition difficult?

- Inter-category similarity


Why is recognition difficult?

- Intra-category variability
  - Instances
  - Illumination
  - Scale
  - Viewpoint/pose
  - Background/occlusion

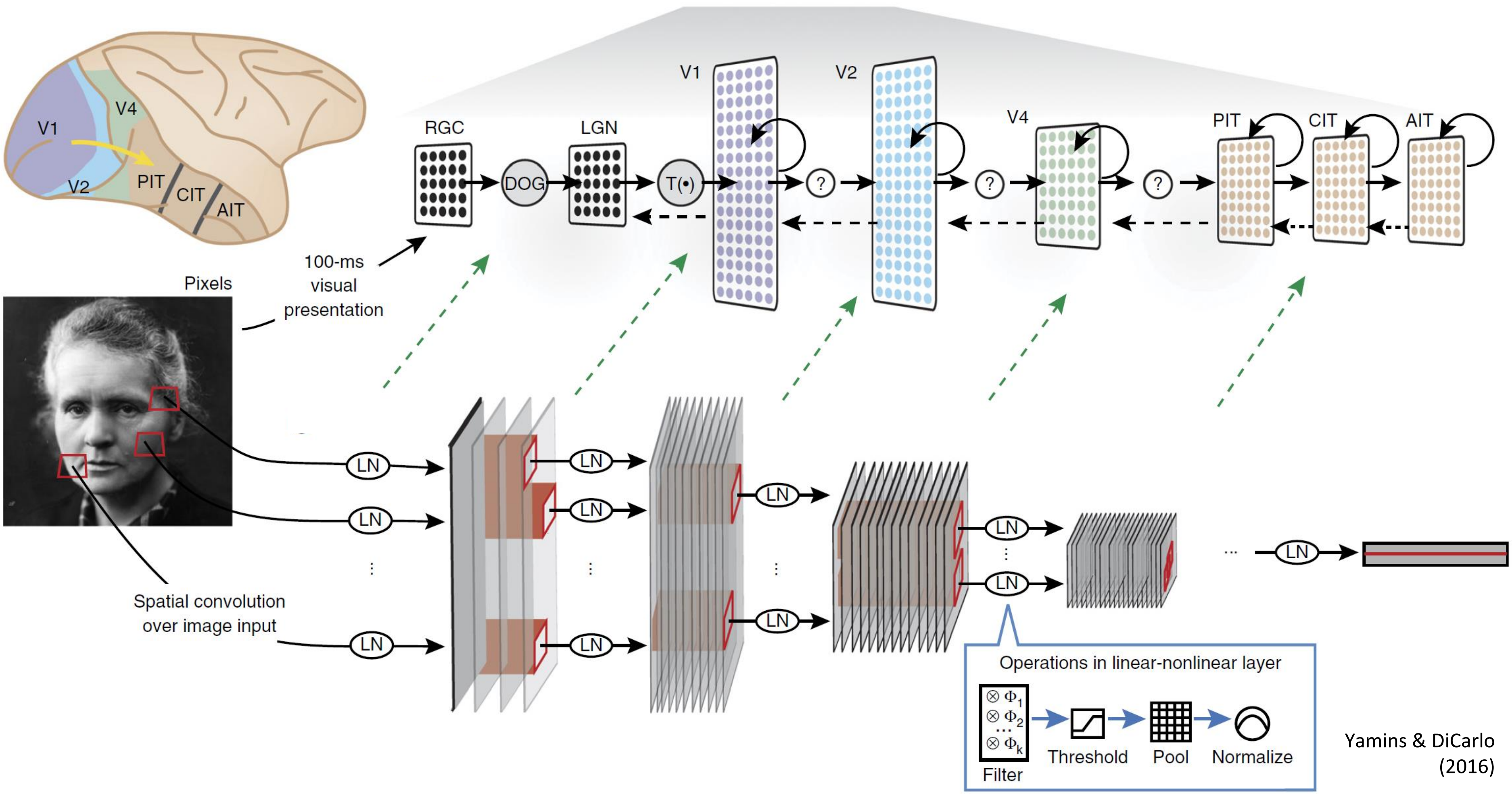


Goal of image recognition

Build a representation of the image that

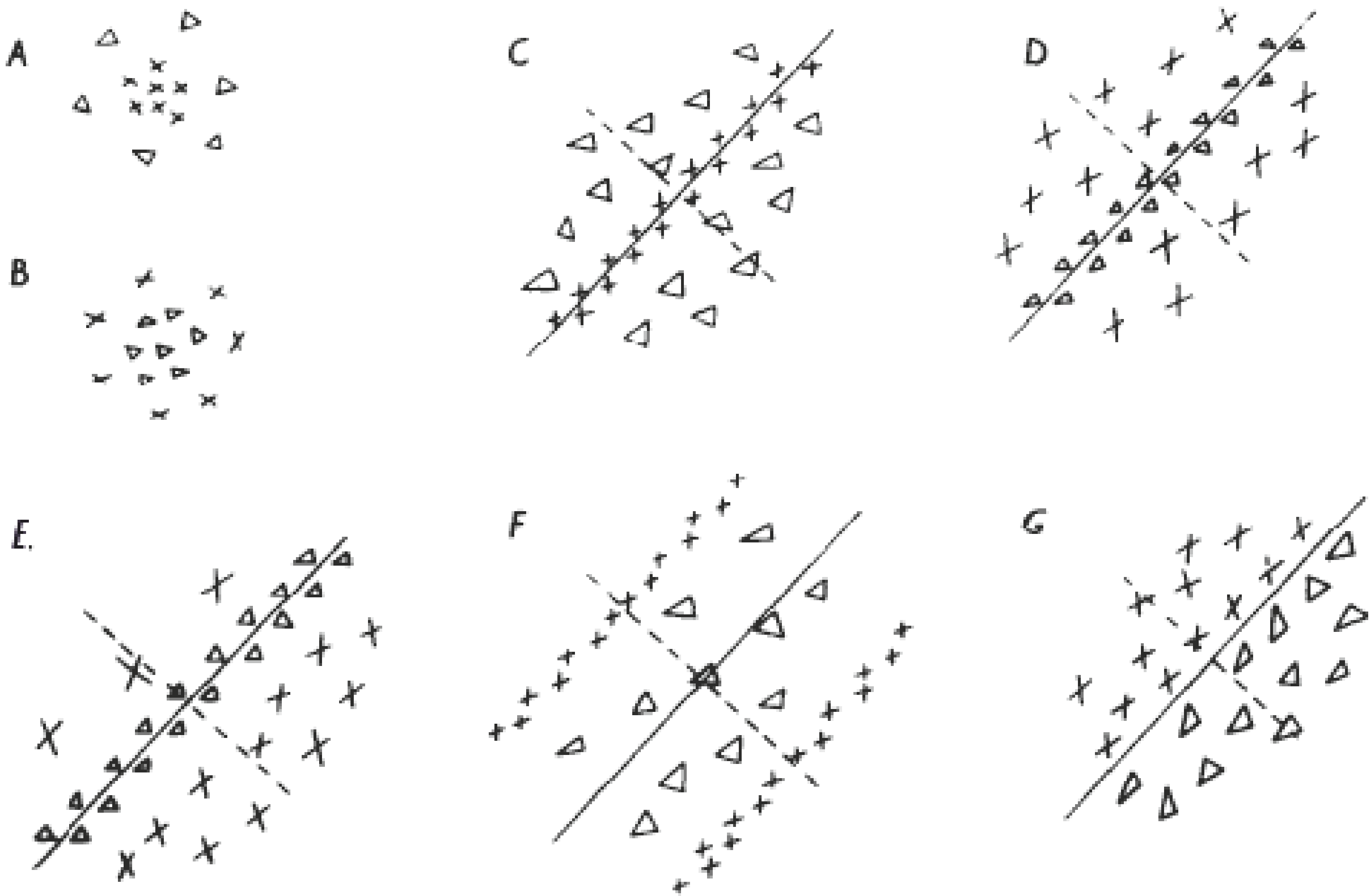
a) distinguishes different categories,
b) but is invariant (or tolerant) to variation within a category

Visual encoding



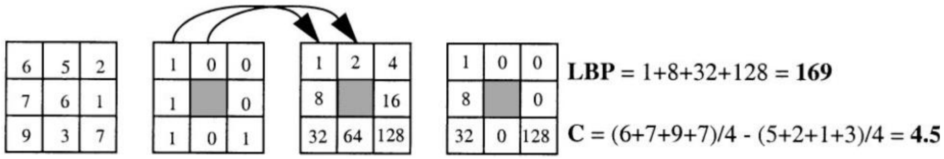
Yamins & DiCarlo (2016)

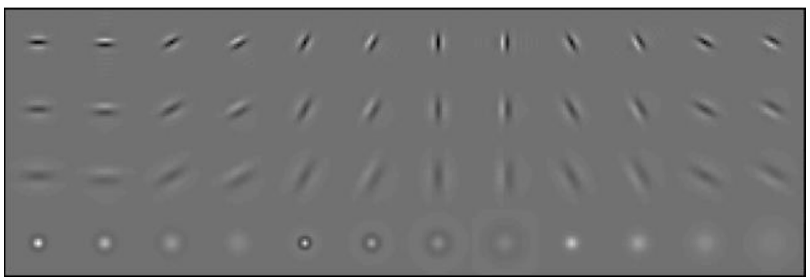
Visual encoding – early vision

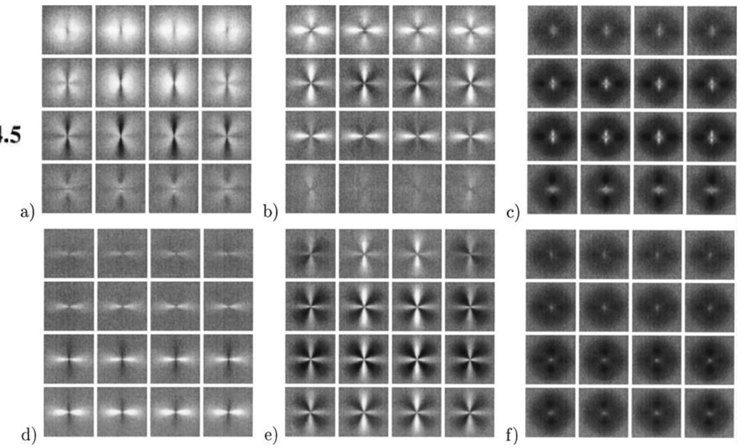


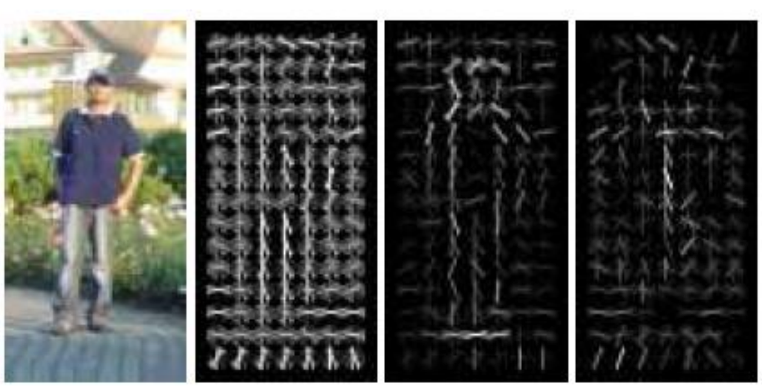
D. Hubel (1981)

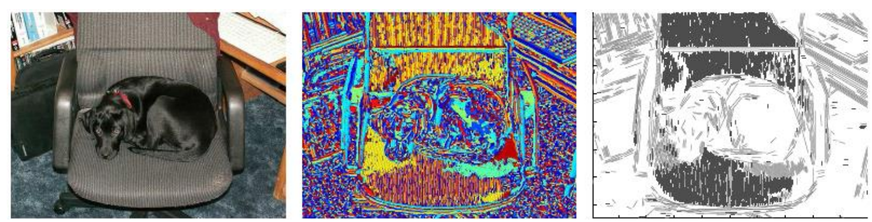
Hand-crafted features





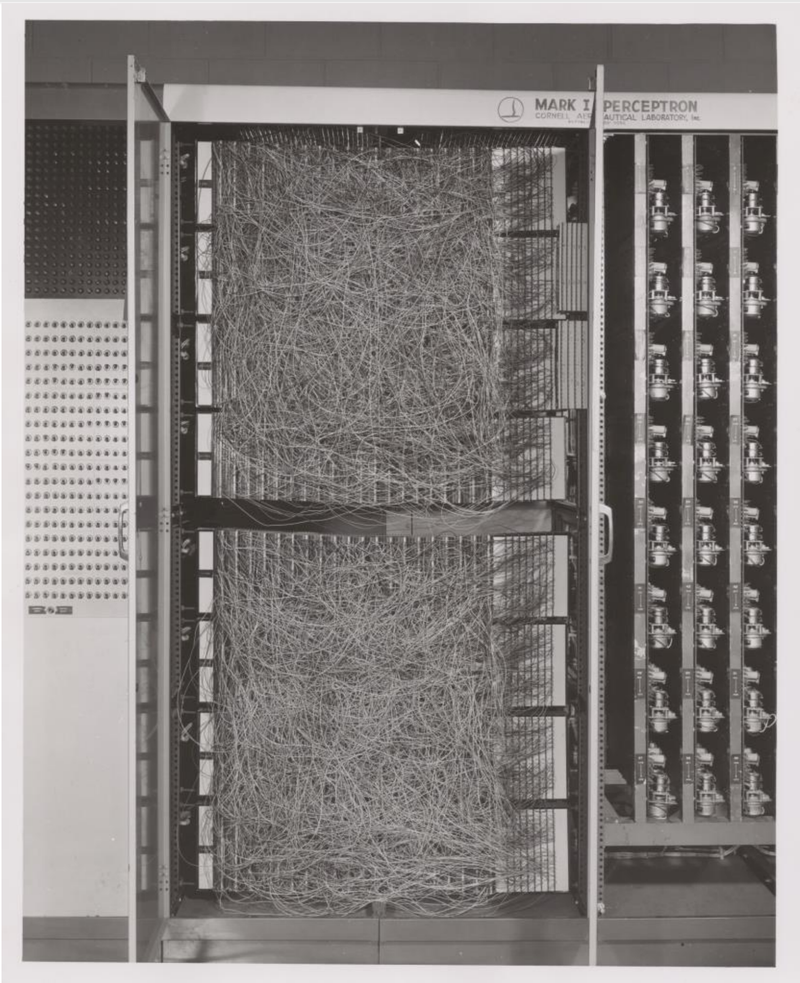






Ojala & Pietikäinen (1999)
Leung & Malik (2001)
Oliva & Torralba (2001)
Dalal & Triggs (2005)
Vedaldi & Zisserman (2012)

Deep learning "revolution"



- Neural networks have been used for computer vision problems since 1950s
- But they only became the state-of-the-art around 2010
- What changed?

Cornell University News Service records, #4-3-15. Division of Rare and Manuscript Collections, Cornell University Library. https://digital.library.cornell.edu/catalog/ss:550351

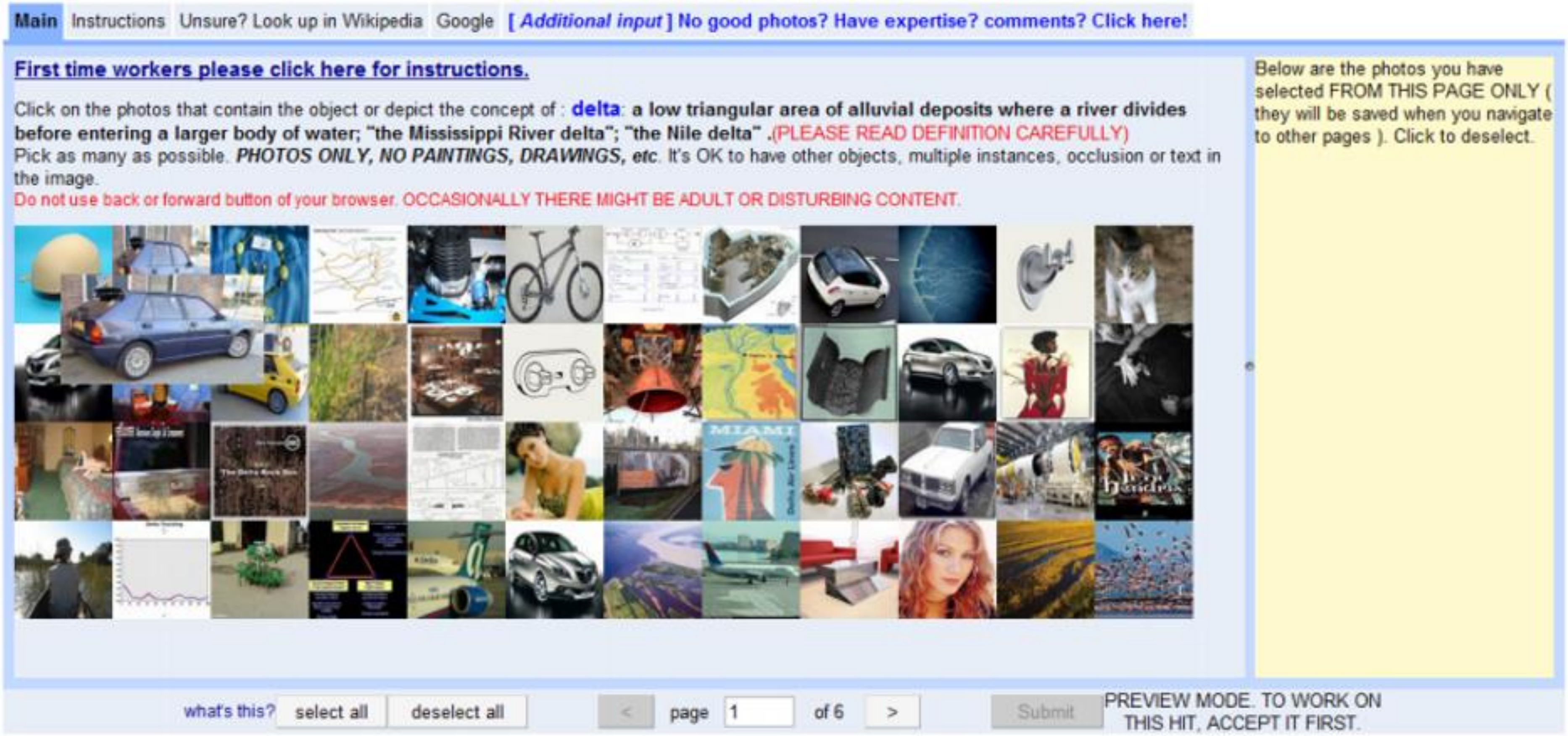
What changed?

- Graphics processing units (GPUs) for fast parallel computation
  - CUDA (Nvidia, 2007) – platform for parallel processing on GPU
- Algorithm improvements:
  - Unsupervised pretraining, ReLU activation function, regularisation (e.g., dropout)
- The internet
  - Massive amounts of text, photos, etc. *with human-annotated labels*

ImageNet

- Based on WordNet, a database of English words organised by concepts
- Class images collected online, manually cleaned by human annotators (2.5 years of annotation work)
- Over 5,000 classes, but commonly-used dataset includes just 1000 of the classes with most exemplars

ImageNet



http://www.image-net.org/

Large-scale image datasets

IMAGENET

http://www.image-net.org/

1.2 million images
1000 object classes

Open images dataset

https://g.co/dataset/open-images
9 million images
19,700 classes

YouTube-8M

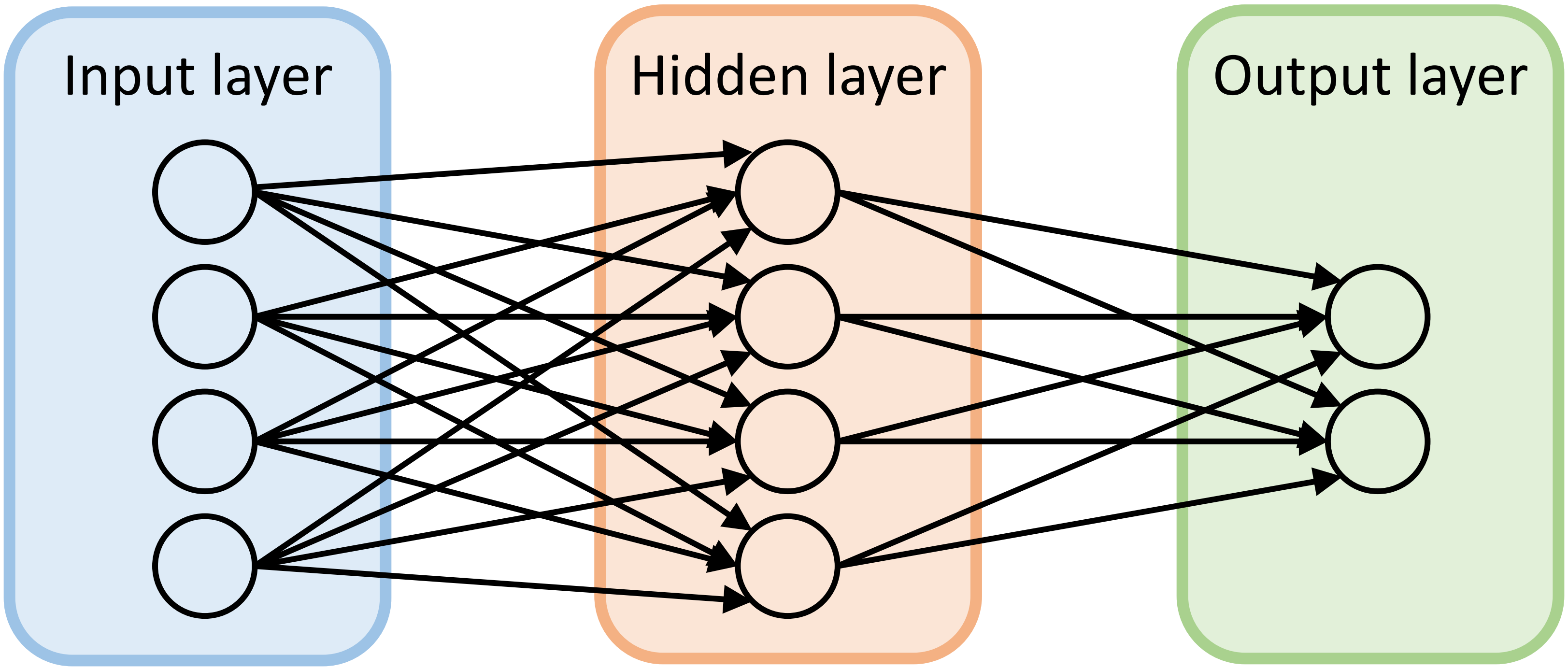
https://research.google.com/youtube8m/
6 million videos
3800 object classes

Image recognition

- Supervised learning problem – map image to class label
- Pre-2010: small number of classes (order of 10-100), hand-crafted features
- 2010-now: “large scale” image recognition (order of 1000-10,000 classes), millions of images, deep-learned features

Neural network review

Neural networks / MLPs



MLP = multilayer perceptron

Neural networks

- Multiple layers of neurons working in parallel on the same input
- Each neuron on layer L receives input from all neurons on layer L-1 (**fully connected layer**) and produces one output
- Neuron's output is a weighted sum of the input, followed by a non-linear activation function

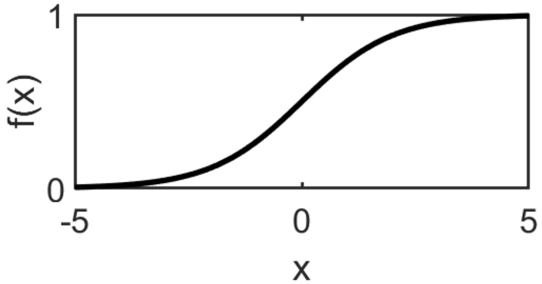
$$y = f\left(\left[\sum_j w_j x_j\right] + b\right) = f(\mathbf{w} \cdot \mathbf{x} + b)$$

Weights and bias learned from data

Non-linear activation function?

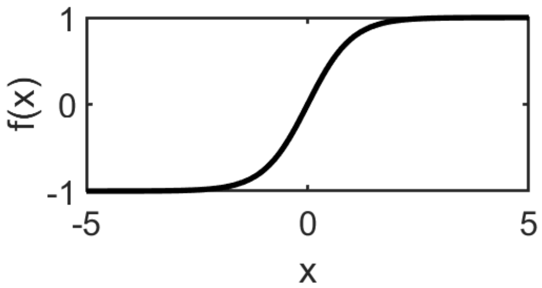
- (logistic) sigmoid (σ):

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



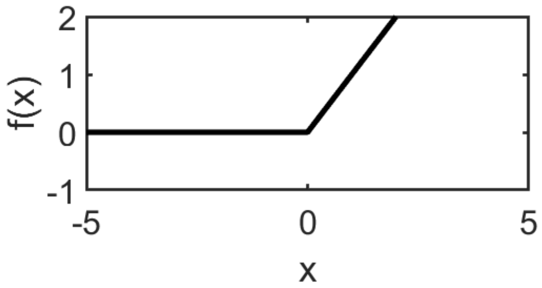
- hyperbolic tan (tanh):

$$f(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$



- rectified linear unit (ReLU):

$$f(x) = \max(0, x)$$



Neural networks

- Train through backpropagation (a form of gradient descent)
- Compute gradient of the loss function with respect to network parameters, starting with output layer and propagating to earlier layers, and adjust weights to reduce loss
- Learning rate is a free parameter
- Loss function usually based on difference between ground truth and prediction (supervised learning)

Neural networks

- Advantages
  - Universal approximator – able to approximate any continuous function on $\mathbb{R}^n$
  - Feature embedding – learns complex features
  - Parallelisable – within each layer, neurons are independent
- Disadvantages
  - Very large number of parameters – high memory/time/data requirements, prone to overfitting

Convolutional layers

Convolutional neural network

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)

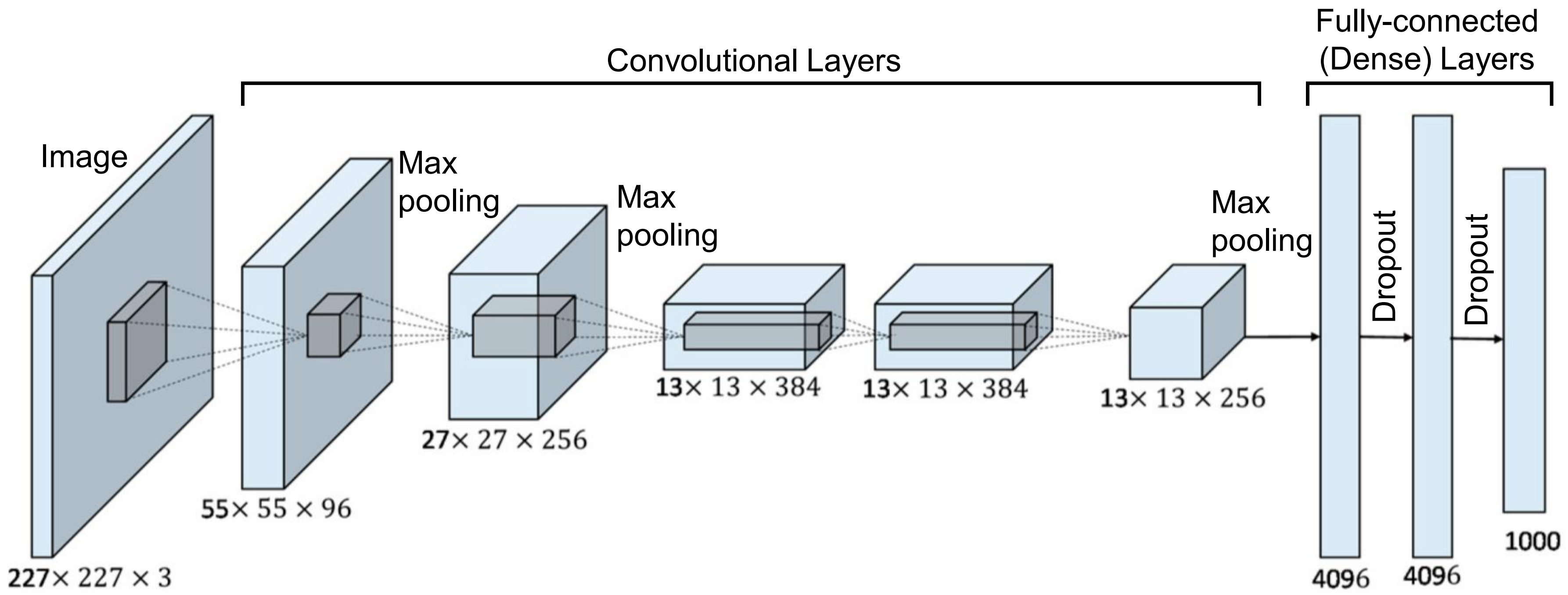


Image: Han, Zhong, Cao, & Zhang (2017)

Why convolution?

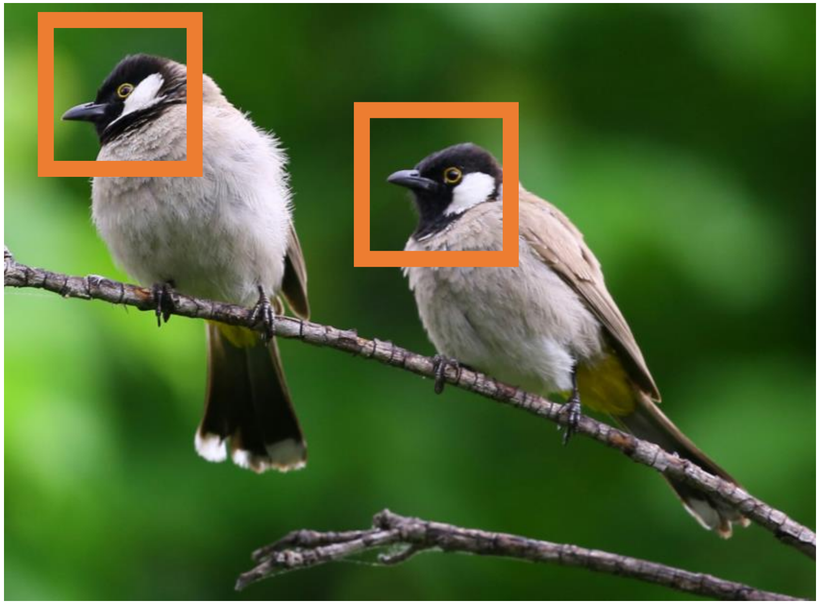
- Regular neural networks can be used for image recognition
- But **convolutional** neural networks are more common for large images
- Why?

Why convolution?

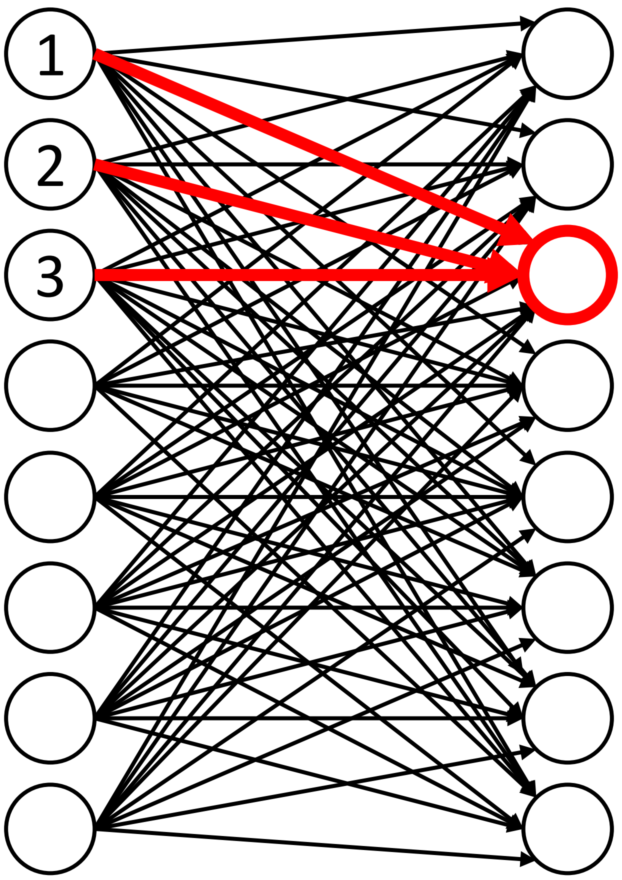
This is how i felt while watching this film. I loved it. It was hilarious.

I loved it, it was fun and moved quickly, no boring drawn out scenes.

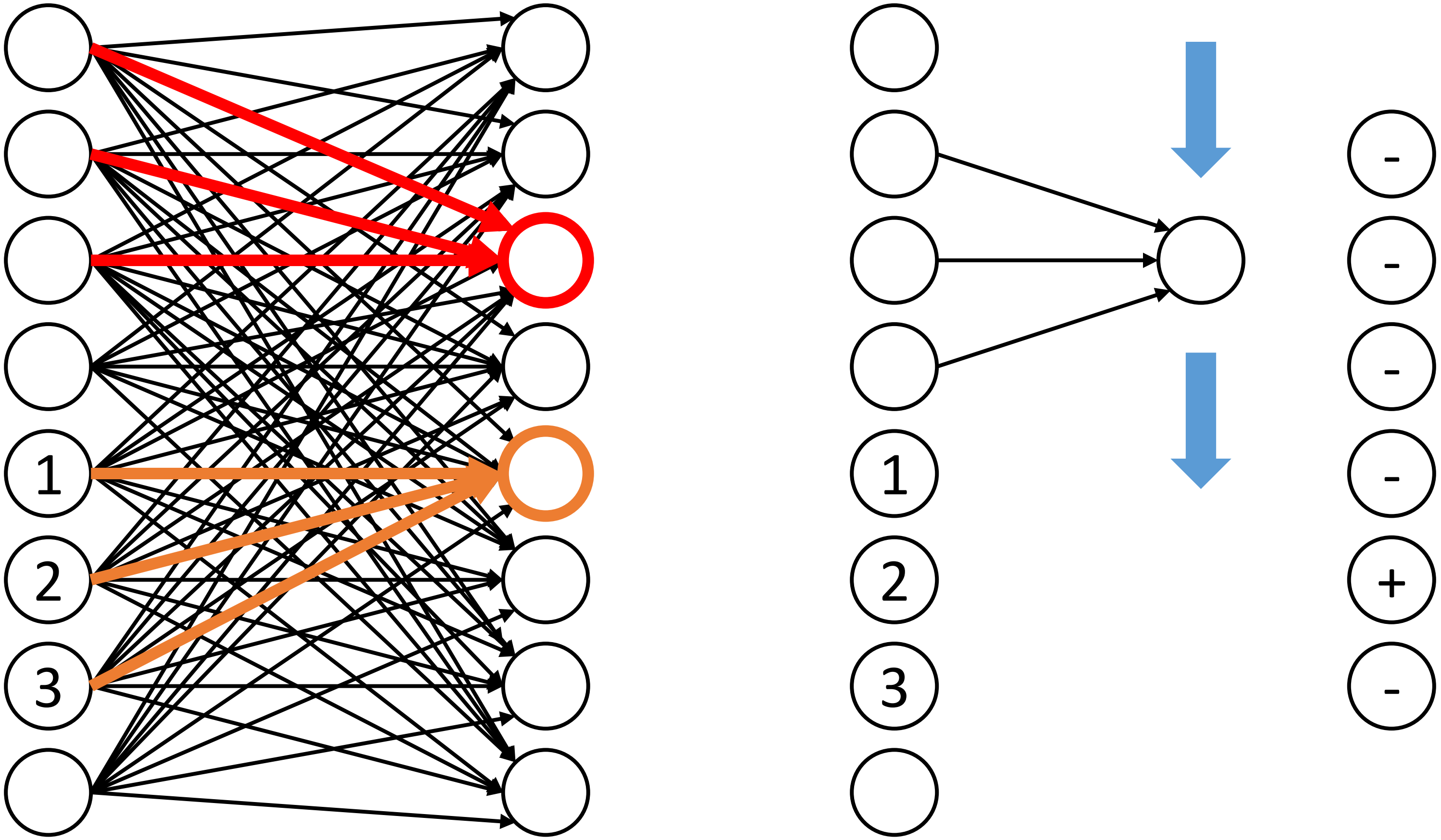
I just got it and it is a great movie!! I loved it!



Convolutional layer



Convolutional layer



Why convolution?

- Regular neural networks can be used for image recognition
- But **convolutional** neural networks are more common for large images
- Why?
  - More efficient learning of local, repeated patterns
  - However, limits what the network can learn

Convolutional layer

- Convolutions are defined by
  - A **kernel**, which is a matrix overlaid on the image and computes an element-wise product with the image pixels
  - A **stride** which defines how many positions in the image to advance the kernel on each iteration (stride = 1 means the kernel will operate on every pixel of the image)

Convolutional layer

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)

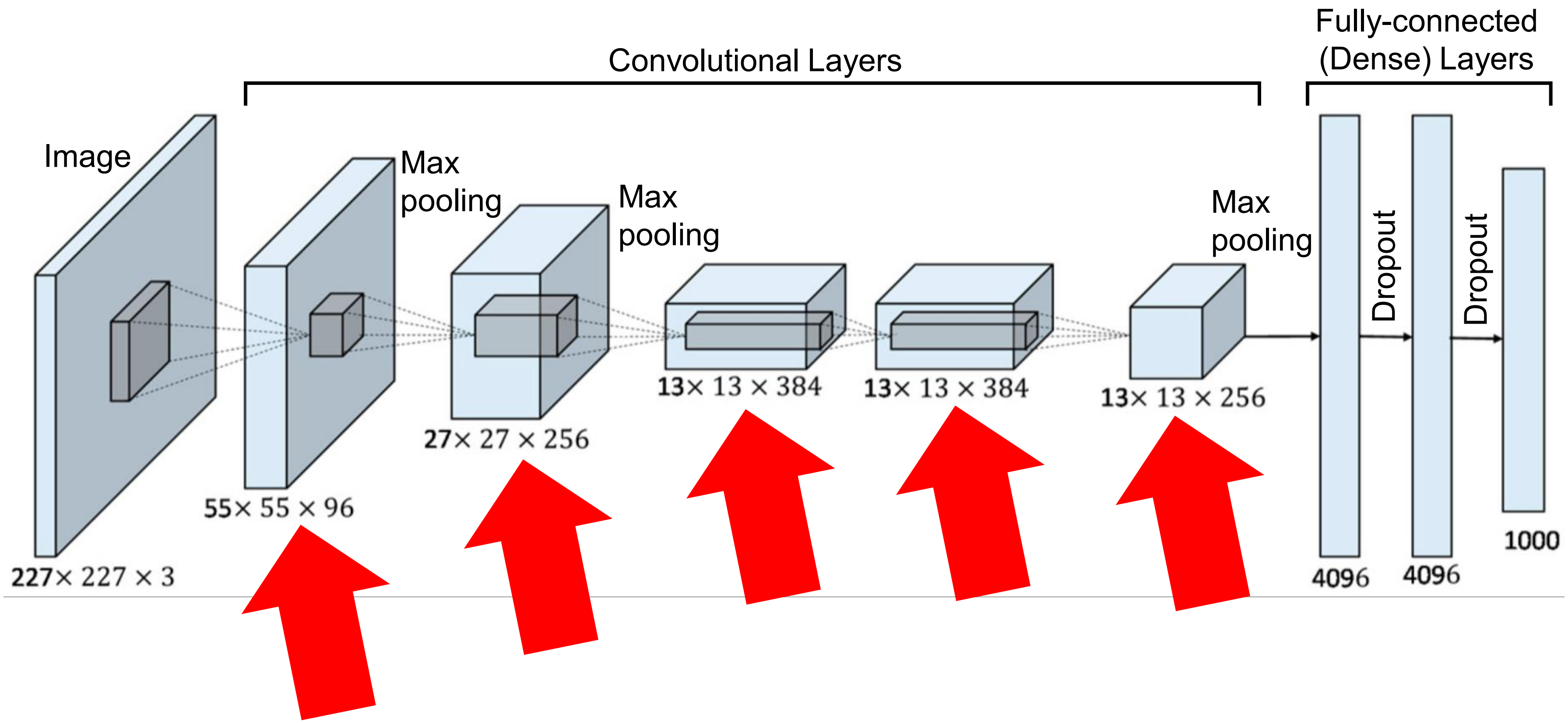


Image: Han, Zhong, Cao, & Zhang (2017)

Convolutional layer

32x32x3 image -> preserve spatial structure

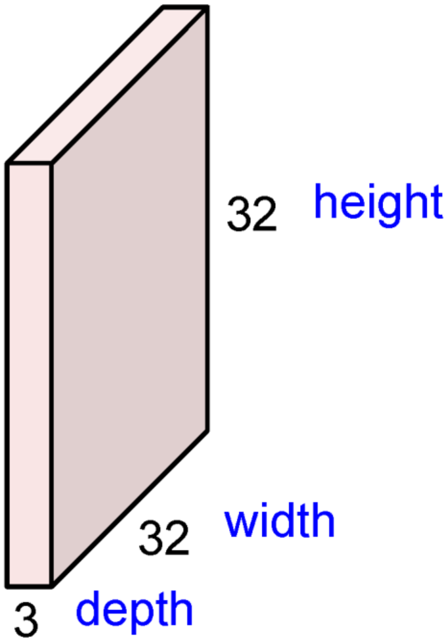
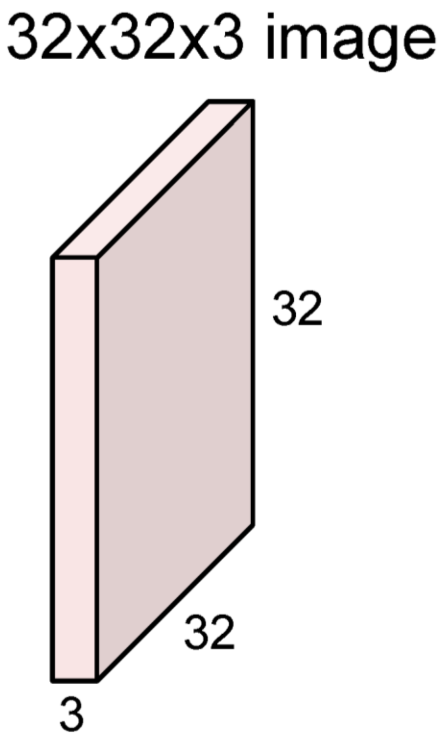
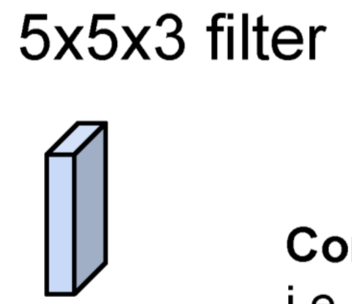


Image: http://cs231n.stanford.edu/

Convolutional layer





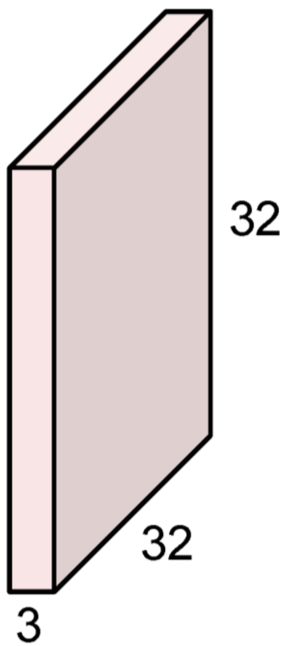
Convolve the filter with the image i.e. “slide over the image spatially, computing dot products”

Technically, most implementations do cross-correlation, but we call it convolution anyway

Image: http://cs231n.stanford.edu/

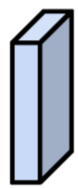
Convolutional layer

32x32x3 image



Colour image = 3 colour channels
Kernel also has 3 channels

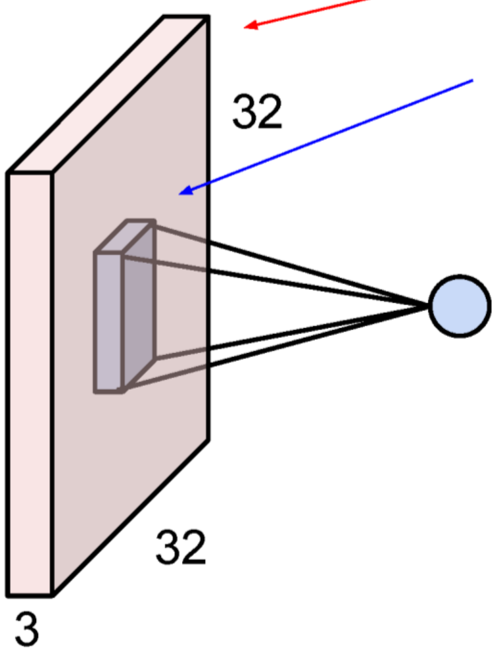
5x5x3 filter



Convolve the filter with the image
i.e. “slide over the image spatially, computing dot products”

Image: http://cs231n.stanford.edu/

Convolutional layer



32x32x3 image

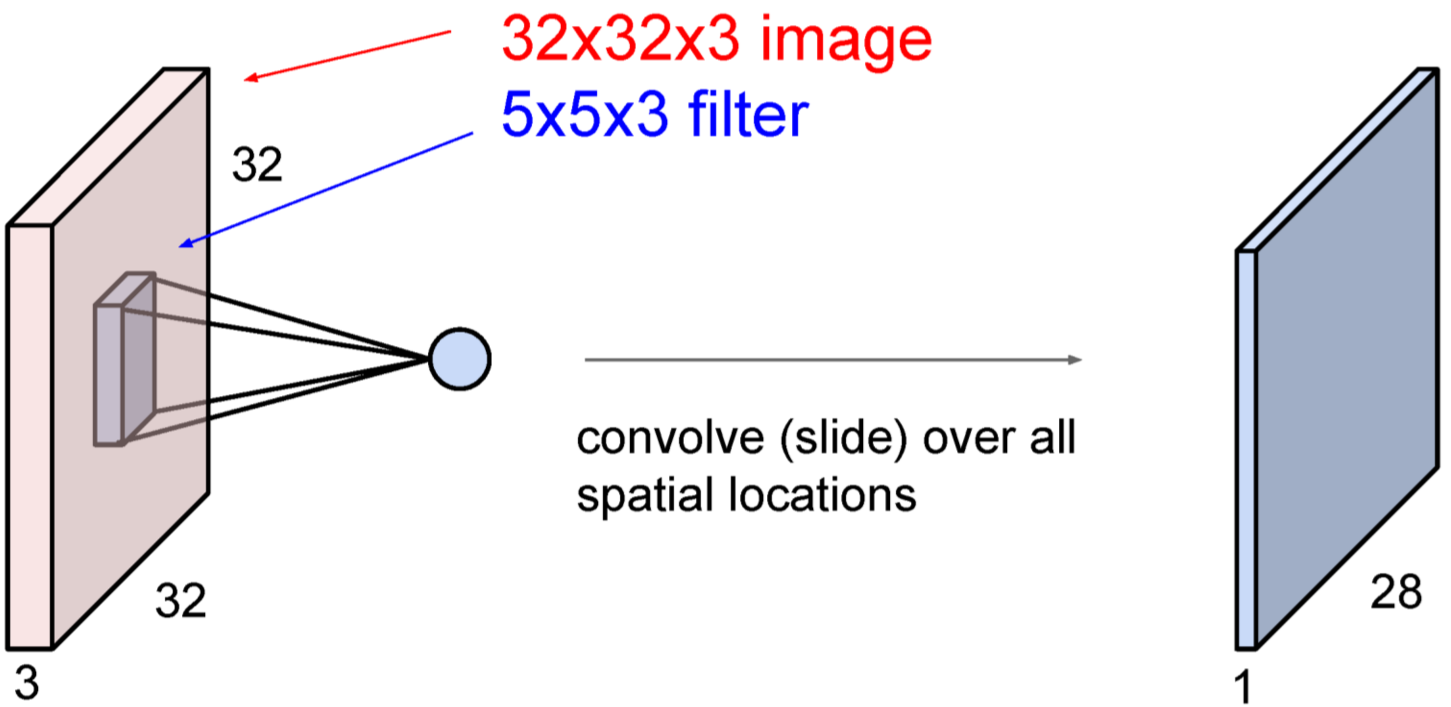
5x5x3 filter w

1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

Image: http://cs231n.stanford.edu/

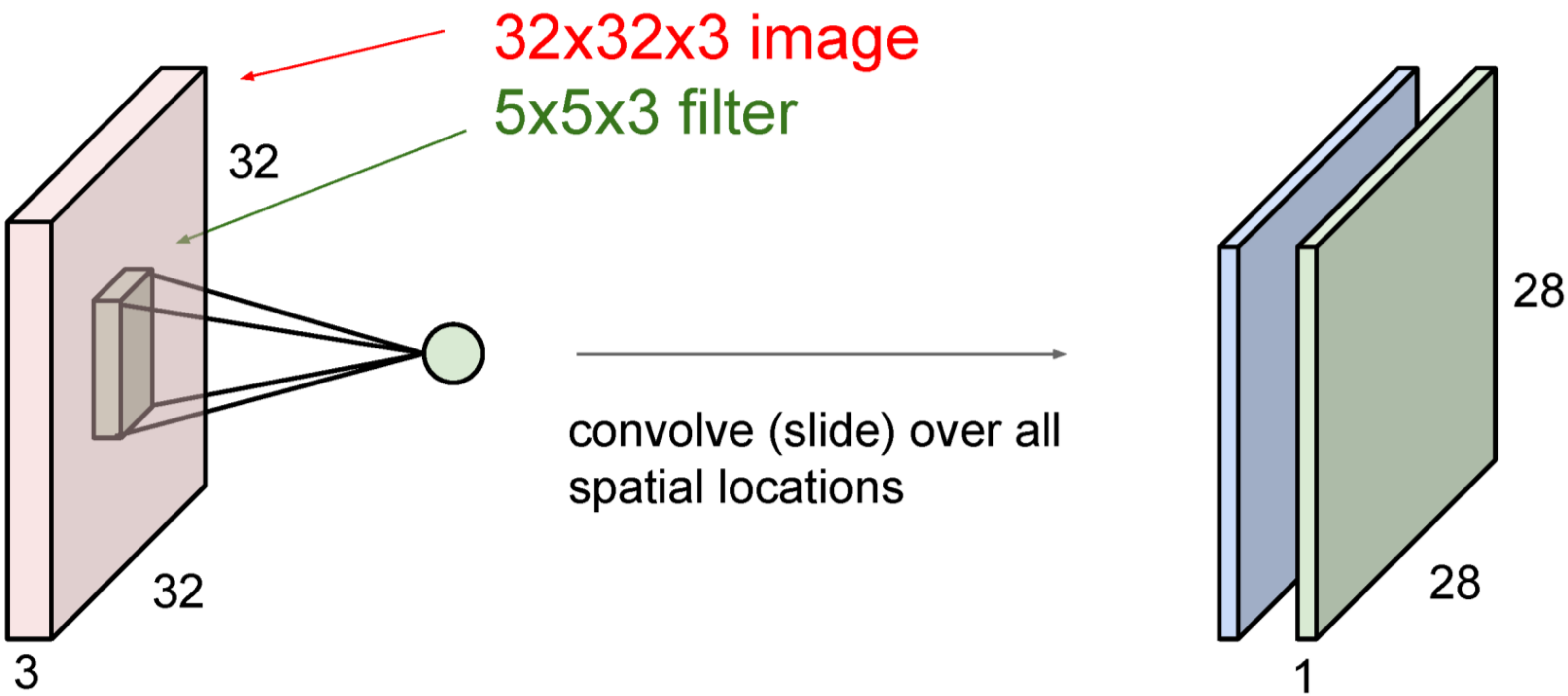
Convolutional layer



Activation map

Image: http://cs231n.stanford.edu/

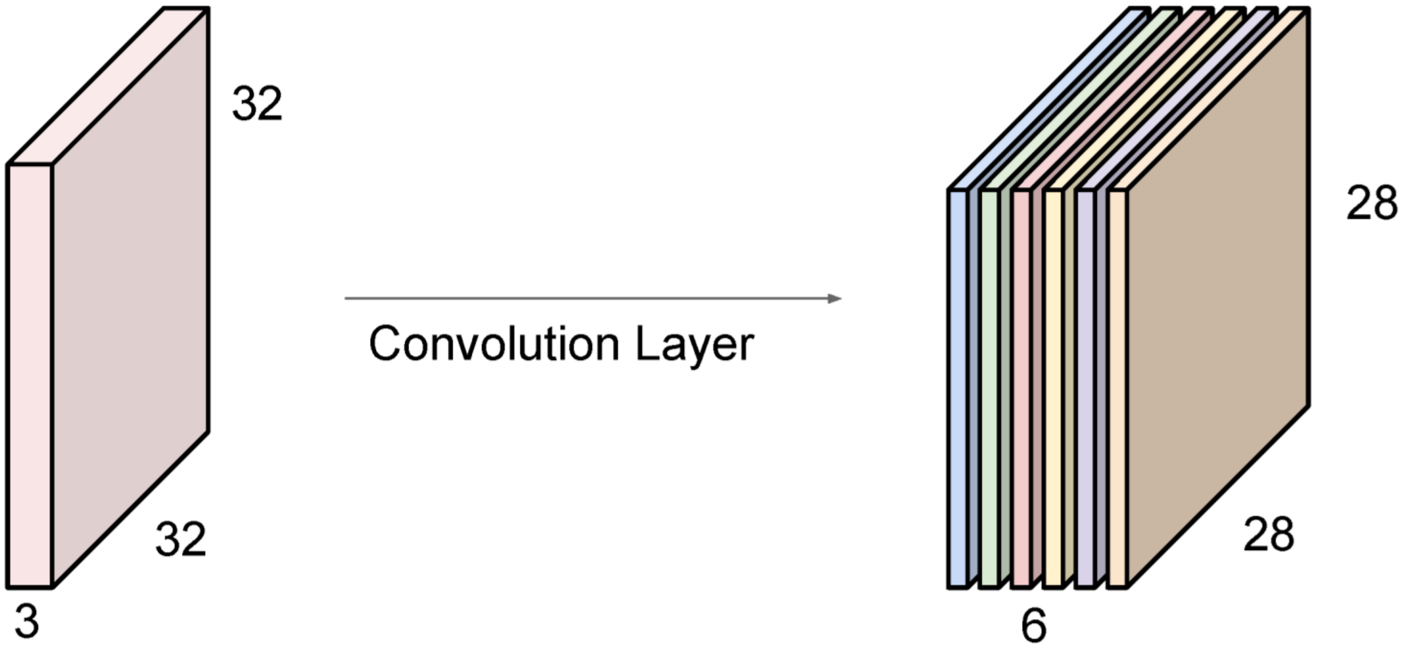
Convolutional layer



Activation maps

Image: http://cs231n.stanford.edu/

Convolutional layer



Activation maps

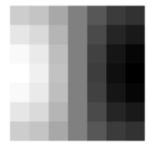
The stack of outputs makes a new 6-channel image, which will be the input to the next layer
6 channels = outputs of 6 different convolution kernels

Image: http://cs231n.stanford.edu/

Convolutional layer output

Input = image

Neuron 1 kernel:



Output:

Neuron 2 kernel:

Output:

…

…

Convolutional layer input/ouput

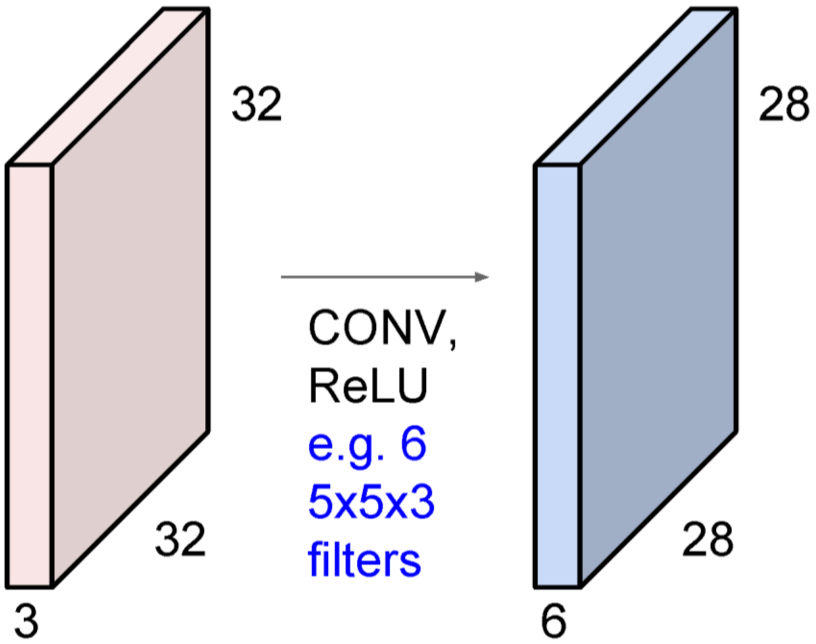


Image: http://cs231n.stanford.edu/

Convolutional layer input/ouput

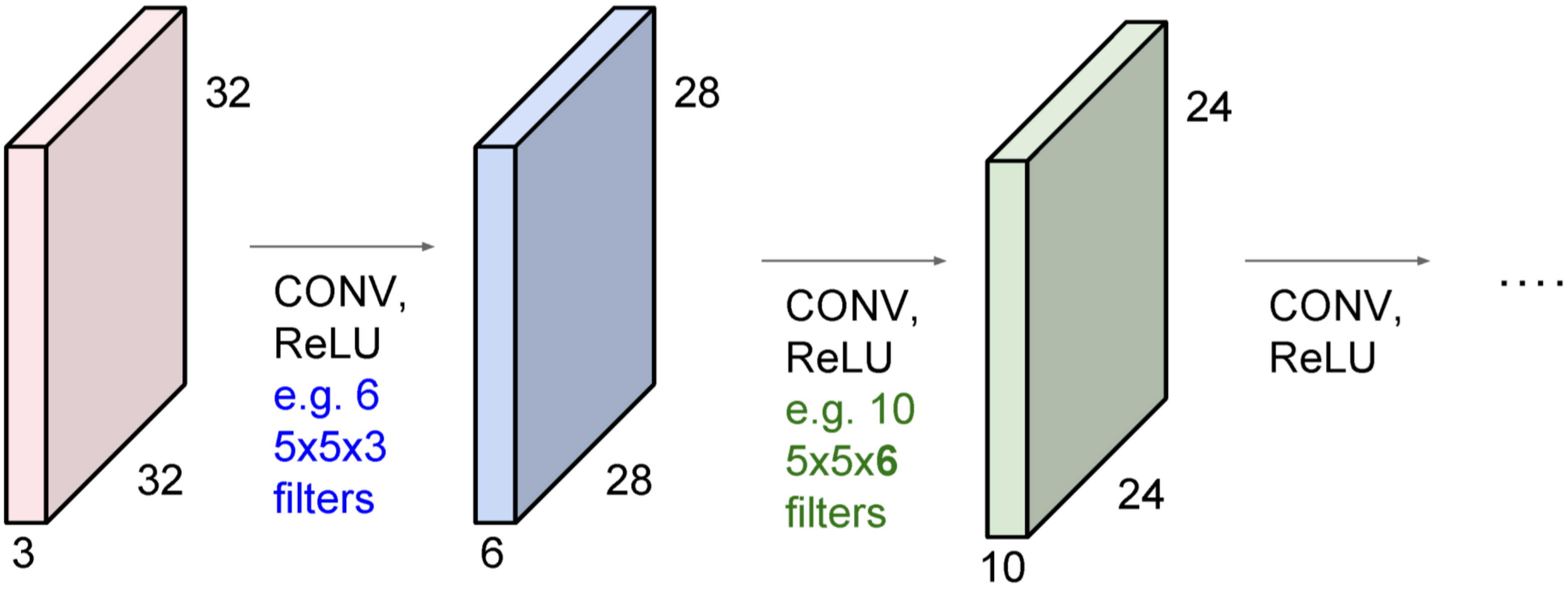


Image: http://cs231n.stanford.edu/

Fully-connected vs. Convolutional

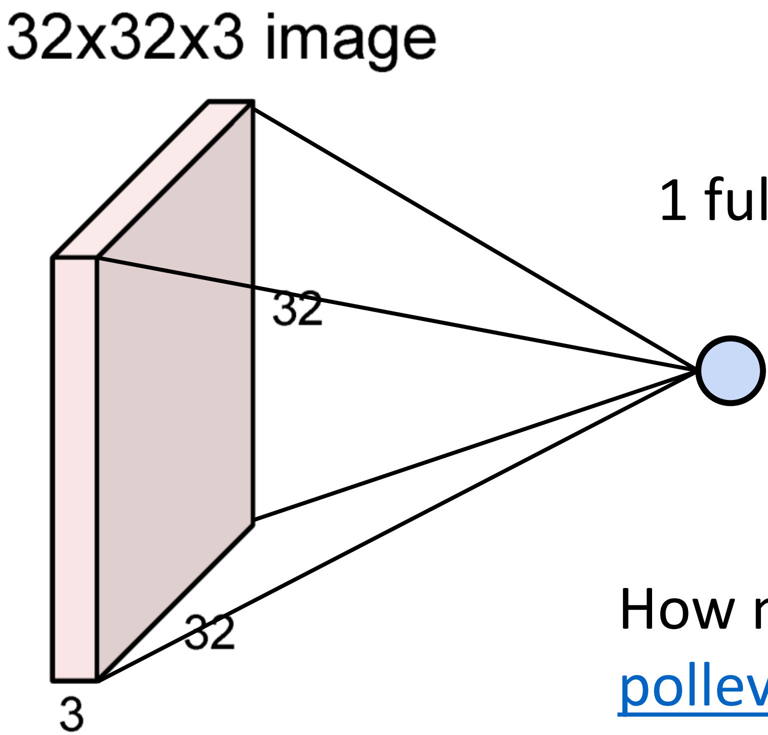
Fully-connected layer

- Each neuron is connected to every neuron in the input
- The neuron learns some combination of the input
- The output to the next layer is the neuron's response

Convolutional layer

- Each neuron is connected to a small patch of the input
- The neuron learns a convolutional kernel on the input
- The output to the next layer is the input convolved with the neuron's kernel

Fully-connected vs. Convolutional



1 fully-connected neuron

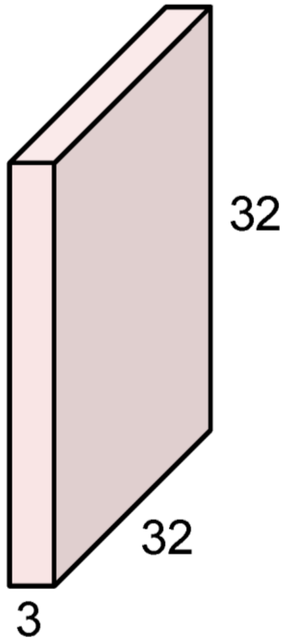
32x32x3 + 1(bias) = 3073

How many learned parameters?
pollev.com/krisehinger432

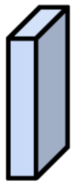
Image: http://cs231n.stanford.edu/

Fully-connected vs. Convolutional

32x32x3 image



1 5x5x3 filter



5x5x3+1 =76

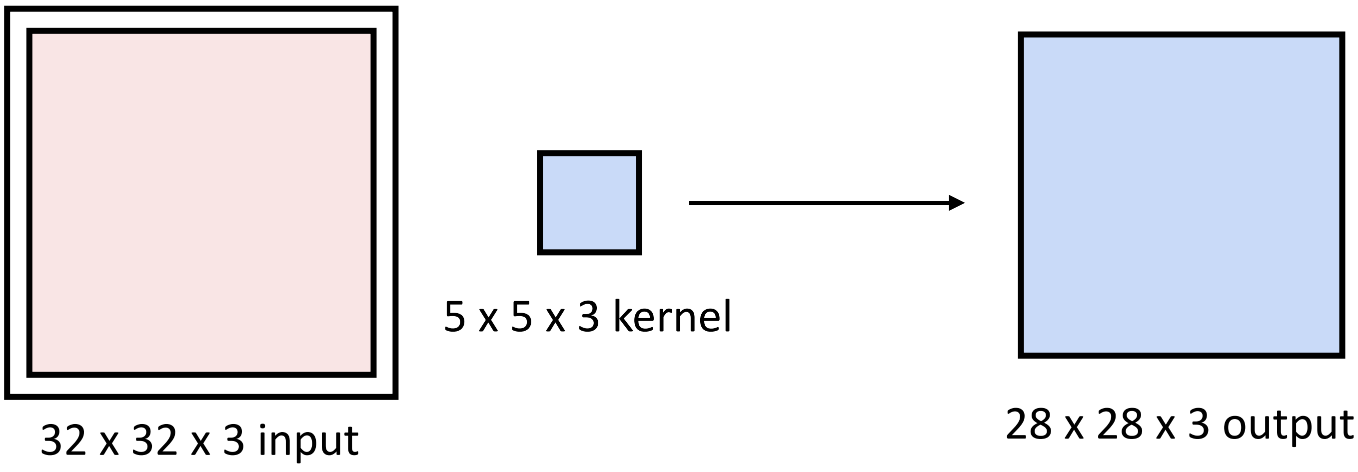
How many learned parameters?

pollev.com/krisehinger432

Image: http://cs231n.stanford.edu/

Convolution output size

- Valid convolution (with kernel larger than 1x1) results in output smaller than input
- If same-size output is needed, pad the input (zero padding is most common)



Convolutional neural network

"AlexNet": Krizhevsky, Sutskever, & Hinton (2012)

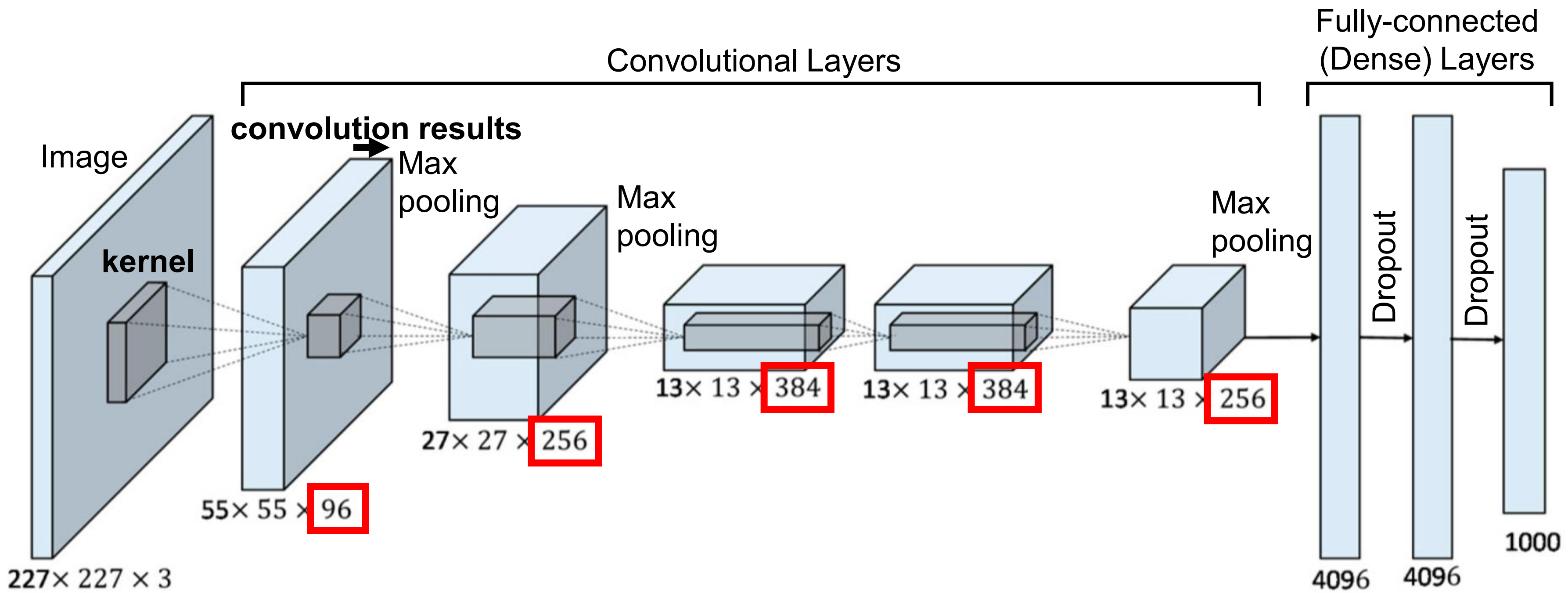
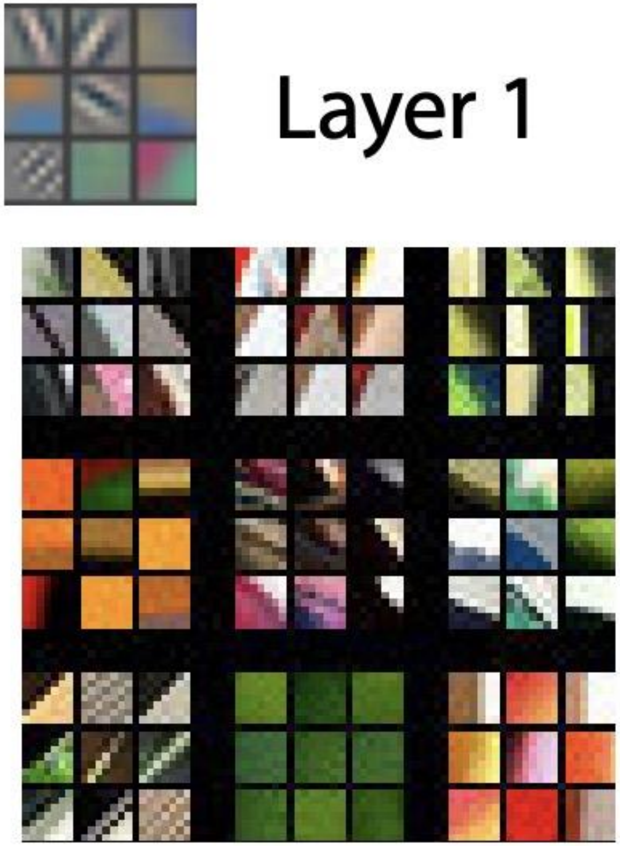


Image: Han, Zhong, Cao, & Zhang (2017)

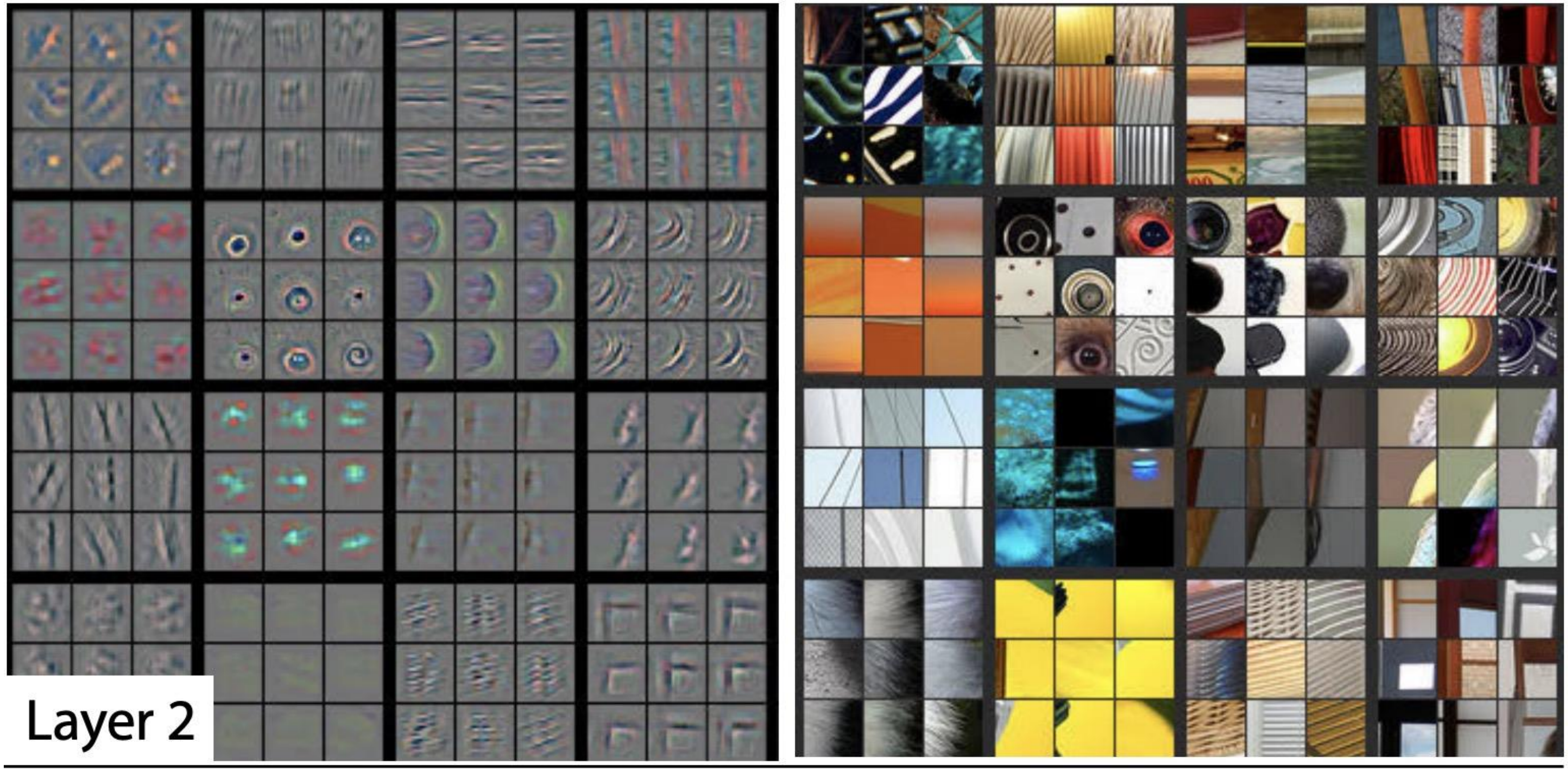
Convolutional layer kernels

- What kernels do the convolutional layers learn?
- In layer 1, mostly edges
- In higher layers?



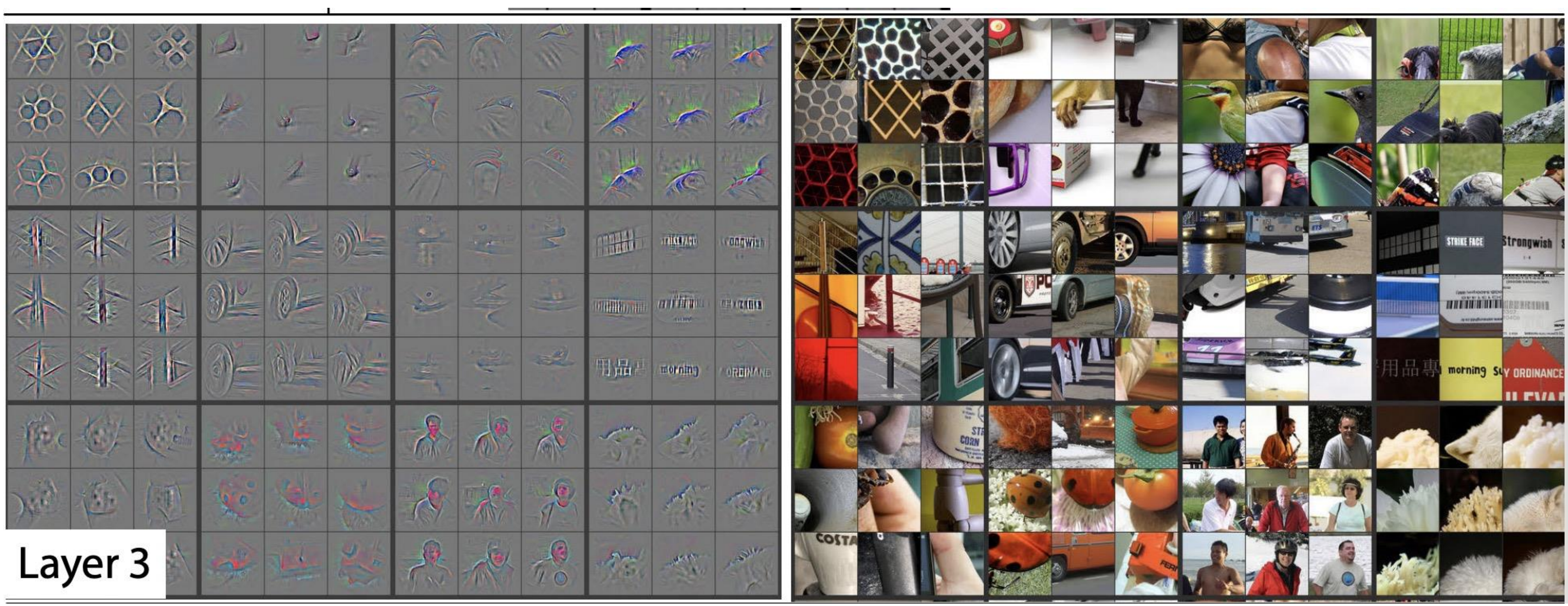
Zeiler & Fergus (2014)

Convolutional layer kernels



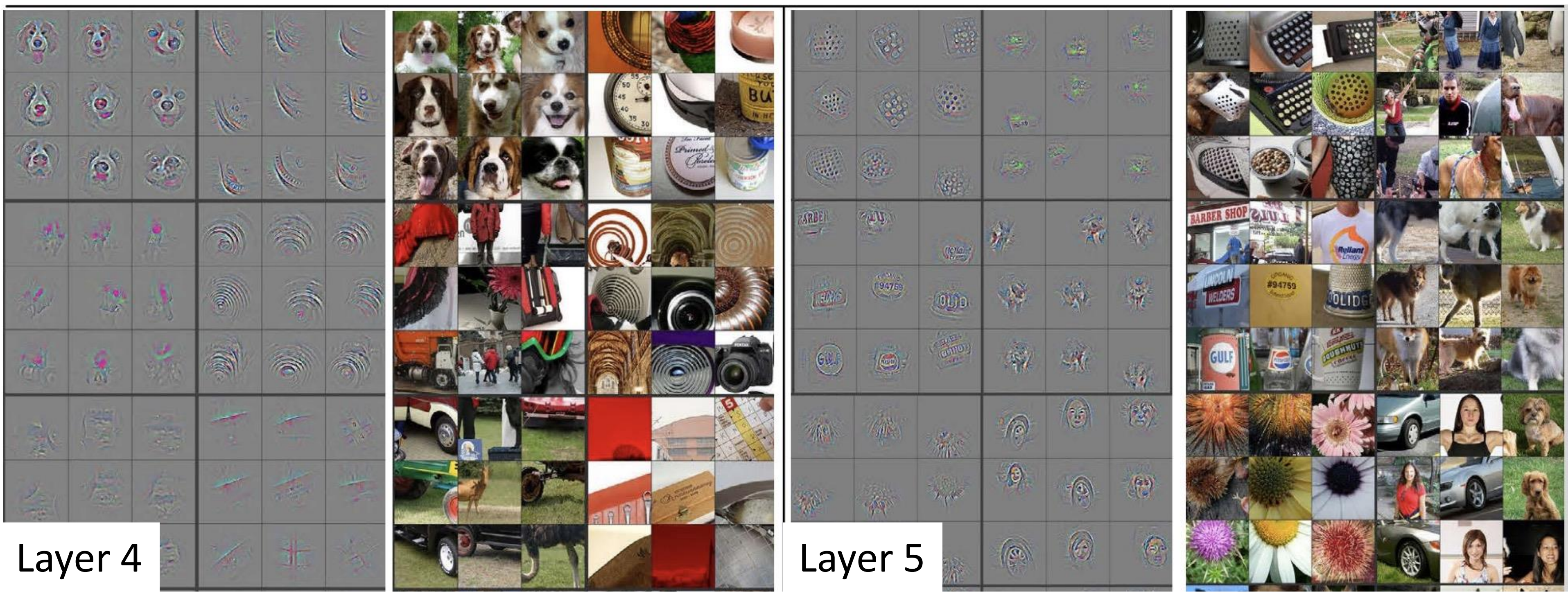
Zeiler & Fergus (2014)

Convolutional layer kernels



Zeiler & Fergus (2014)

Convolutional layer kernels



Zeiler & Fergus (2014)

Convolutional layers

- Advantages of convolutional layers
  - Efficient – learns to recognize the same features anywhere in the image, with fewer parameters compared to fully-connected layer
  - Preserves spatial relations – output is an image with values indicating where features are present
- Disadvantages of convolutional layers
  - Limited kernel size means model is limited to learning local features

Summary

- Convolutional neural networks – variation on standard (fully-connected) neural networks
- Each convolutional layer learns a set of kernels and outputs activation maps (= input convolved with learned kernel)