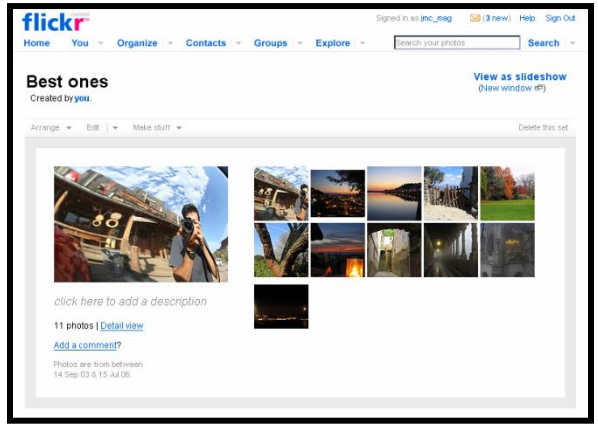
Visual Search

Convolutional Neural Networks

Web Data Mining and Search

Visual search

Web content

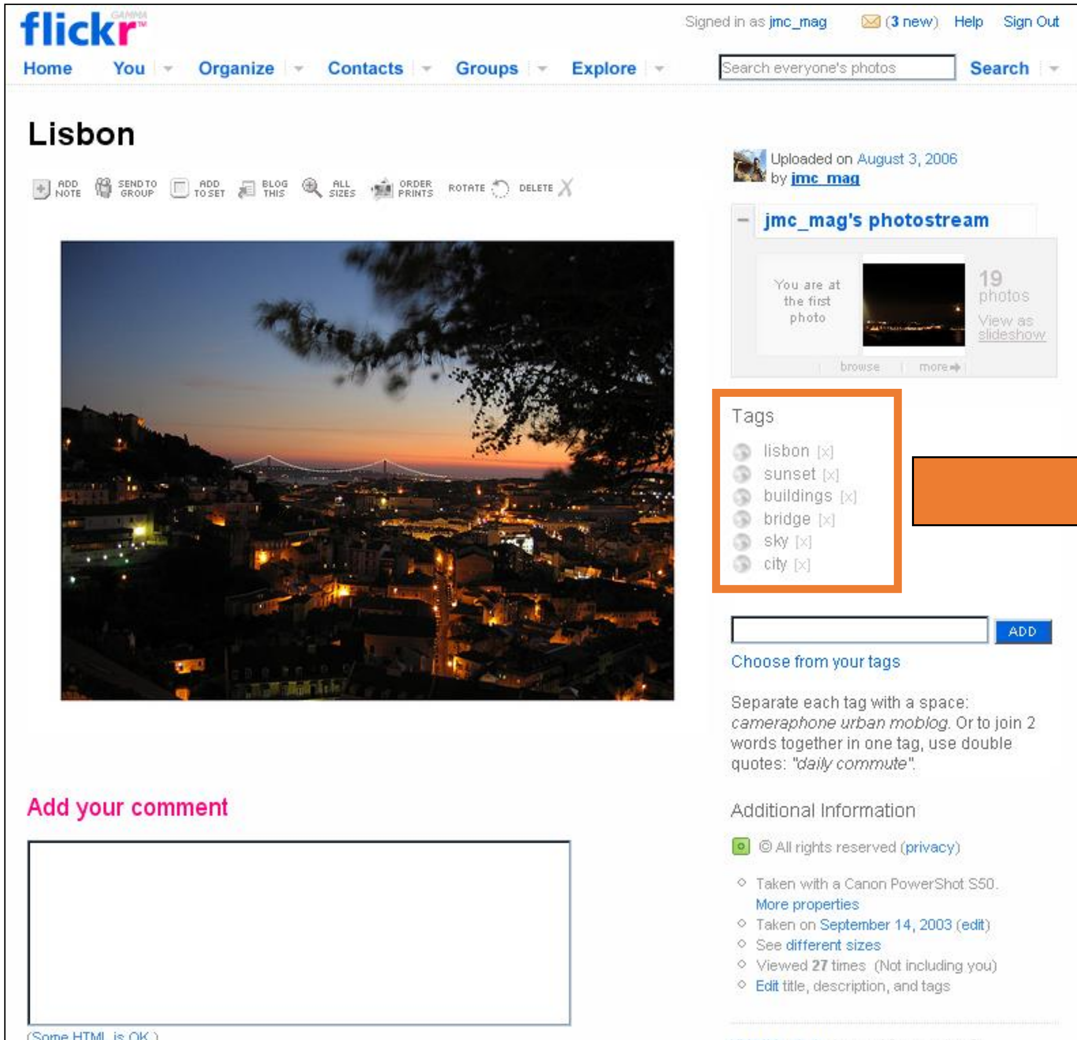


Photographs

News/Sports/Movies

How to improve search capabilities
over these types of content?

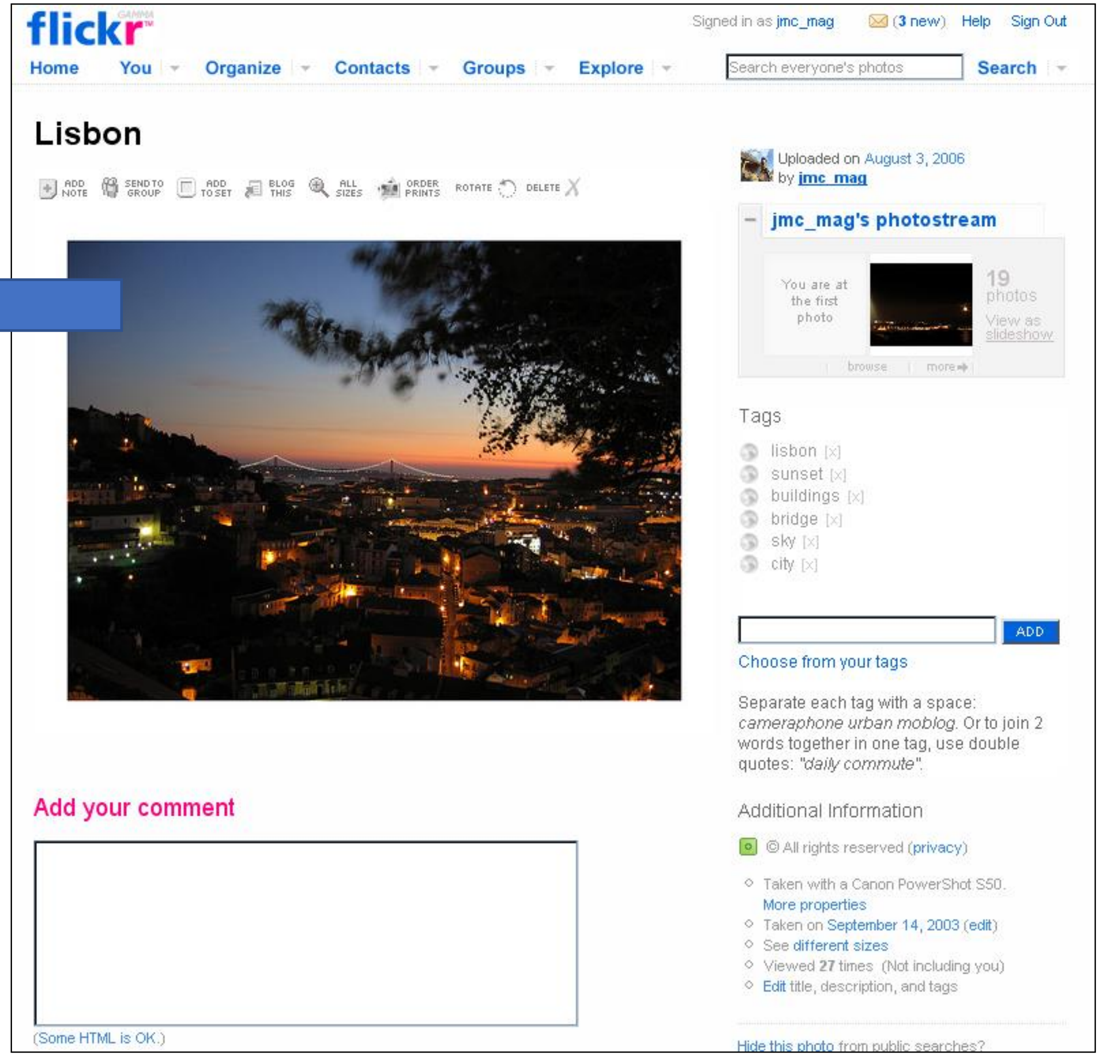
Visual search by keyword



ElasticSearch

Visual search by text

Sunset picture taken in Lisbon from Miradouro da Graça, with Ponte 25 de Abril on the landscape and with streetlights already on.



ElasticSearch

Visual search by example

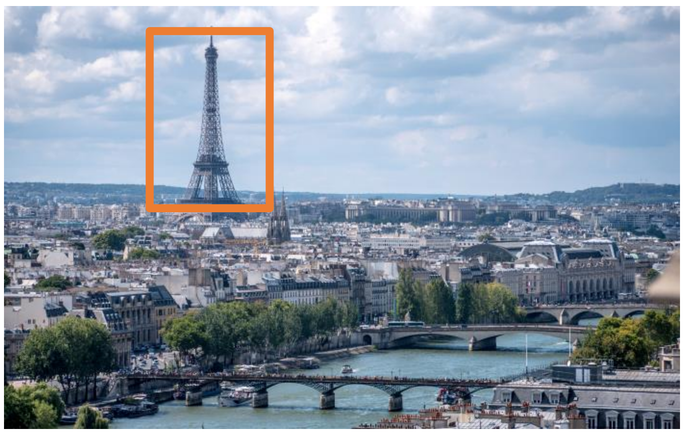
User query

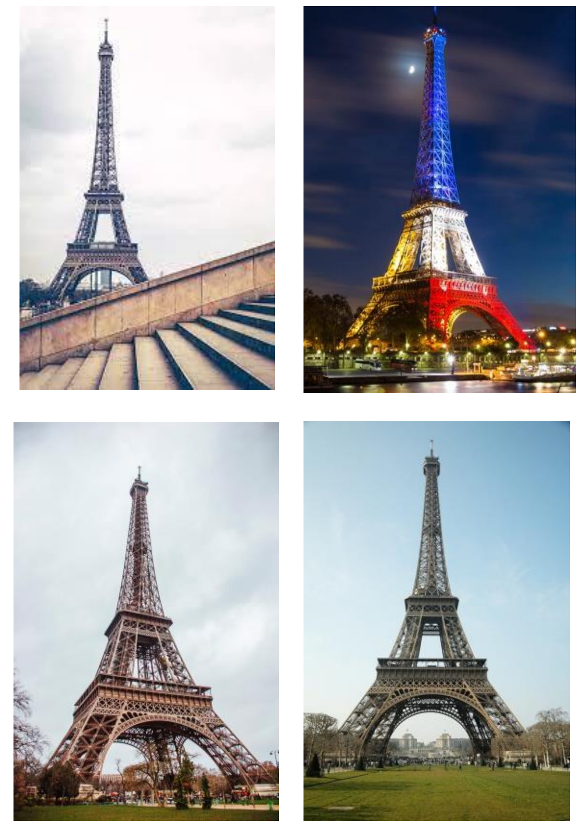
Search results

Visual search by region

Search results

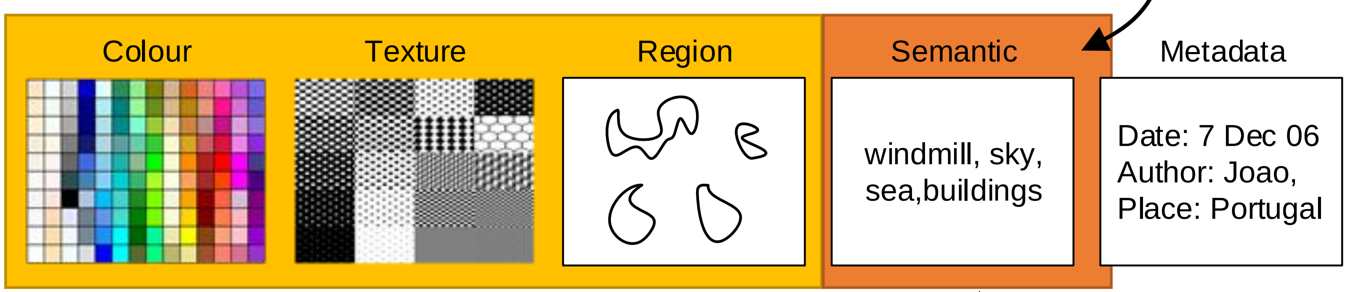
User bounded query





Hand-crafted vs learned feature extraction

Approach B: end-to-end classifiers learn task specific data representations.



Approach A: classifiers learn with hand-crafted methods for feature extraction.

Learning a keyword probability distribution

elkhound
otterhound
Saluki
Scottish deerhound
Weimaraner
Staffordshire
bullterrier
American
Staffordshire terrier
Bedlington terrier
Border terrier
Kerry blue terrier
Irish terrier
Norfolk terrier
Norwich terrier
Yorkshire terrier
wire-haired fox
terrier
Lakeland terrier
Sealyham terrier
Airedale
cairn

tench
goldfish
great white shark
tiger shark
hammerhead
electric ray
stingray
cock
hen
ostrich
brambling
goldfinch
house finch
junco
indigo bunting
robin
bulbul

Komodo dragon
African crocodile
American alligator
triceratops
thunder snake
ringneck snake
hognose snake
green snake
king snake
garter snake
water snake
vine snake
night snake
boa constrictor
rock python
Indian cobra
green mamba
sea snake
horned viper
diamondback
sidewinder

tick
centipede
black grouse
ptarmigan
ruffed grouse
prairie chicken
peacock
quail
partridge
African grey
macaw
sulphur-crested
cockatoo
lorikeet
coucal
bee eater
hornbill
hummingbird
jacamar
toucan
drake
red-breasted
merganser
goose
black swan
tusker
echidna
platypus

wallaby
koala
wombat
jellyfish
sea anemone
brain coral
flatworm
nematode
conch
snail
slug
sea slug
chiton
chambered
nautilus
Dungeness crab
rock crab
fiddler crab
king crab
American
lobster
spiny lobster
crayfish
hermit crab
isopod

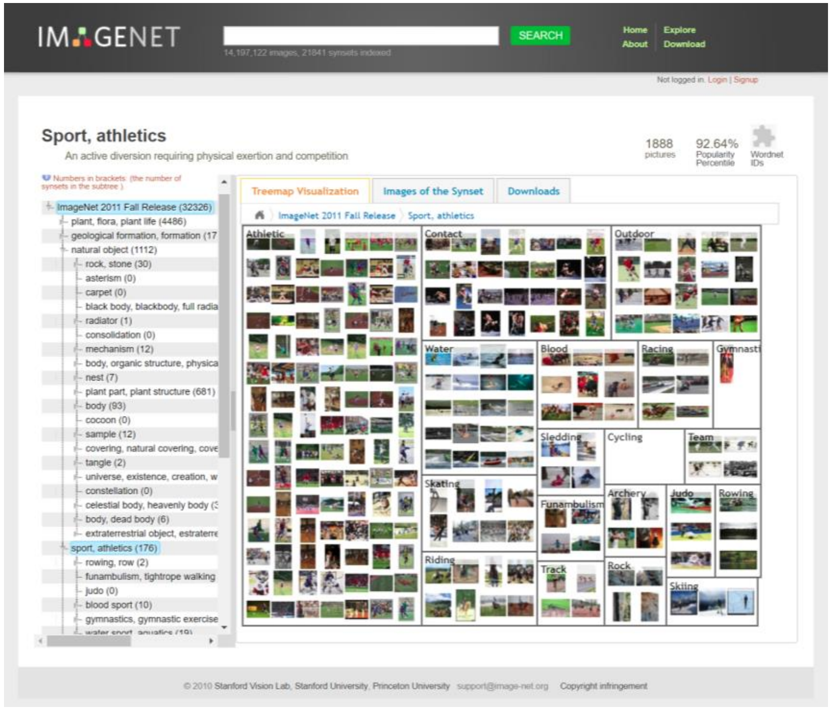
white stork
black stork
spoonbill
flamingo
little blue
heron
American
egret
bittern
crane
limpkin
European
gallinule
American coot
bustard
ruddy
turnstone
red-backed
sandpiper
redshank
dowitcher
oystercatcher
pelican
king penguin
albatross
grey whale
killer whale
dugong
sea lion

valley
volcano
ballplayer
groom
scuba diver
rapeseed
daisy
yellow lady's
slipper
corn
acorn
hip
buckeye
coral fungus
agaric
gyromitra
stinkhorn
earthstar
hen-of-the-woods
bolete
ear
toilet tissue

ImageNet competition

- A total of 1.43 million images annotated with 1.000 object classes
- The goal is to annotated a test sample and be as accurate as possible.
- Human error is 5.1%
- Great impact in advancing the state of the art.

http://image-net.org/explore.php

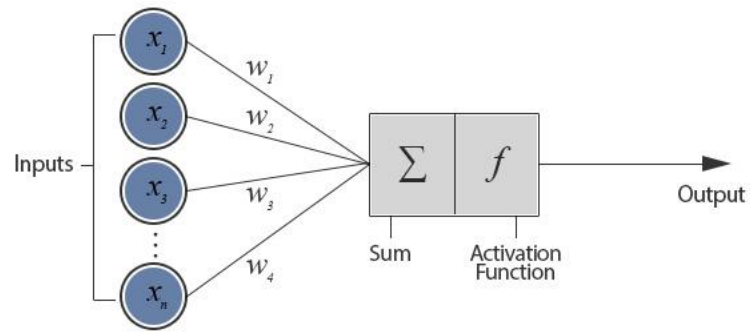


Perceptron: general formulation

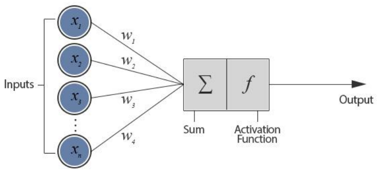
- **Binary classification:**

$$z = w_0 + w_1 x_1 + \ ... + w_n x_n$$

$$\hat{y} = f(z) = \begin{cases} +1 & , if\ z \geq 0 \\ -1 & , if\ z < 0 \end{cases}$$

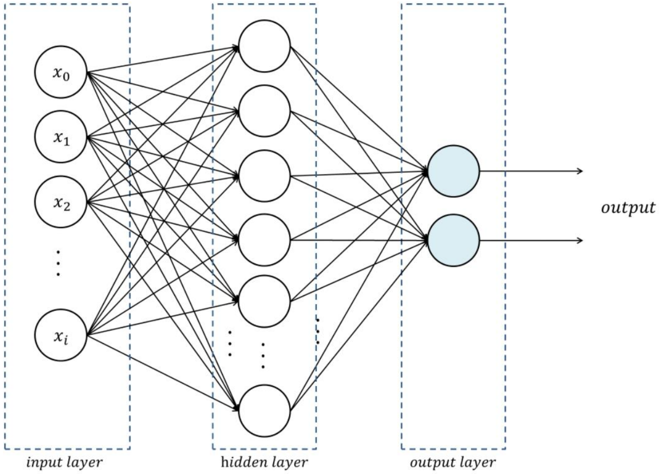


- **Input:** Vectors $\boldsymbol{x^{(j)}}$ and labels $\boldsymbol{y^{(j)}}$
  - Vectors $\boldsymbol{x^{(j)}}$ are real valued where $\|\boldsymbol{x}\|_2 = \mathbf{1}$
- **Goal:** Find vector $\boldsymbol{w} = (\boldsymbol{w_1}, \boldsymbol{w_2}, \dots, \boldsymbol{w_d})$
  - Each $\boldsymbol{w_i}$ is a real number

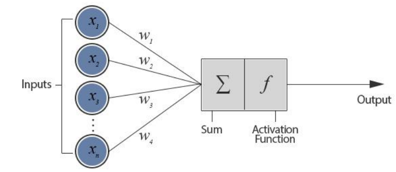


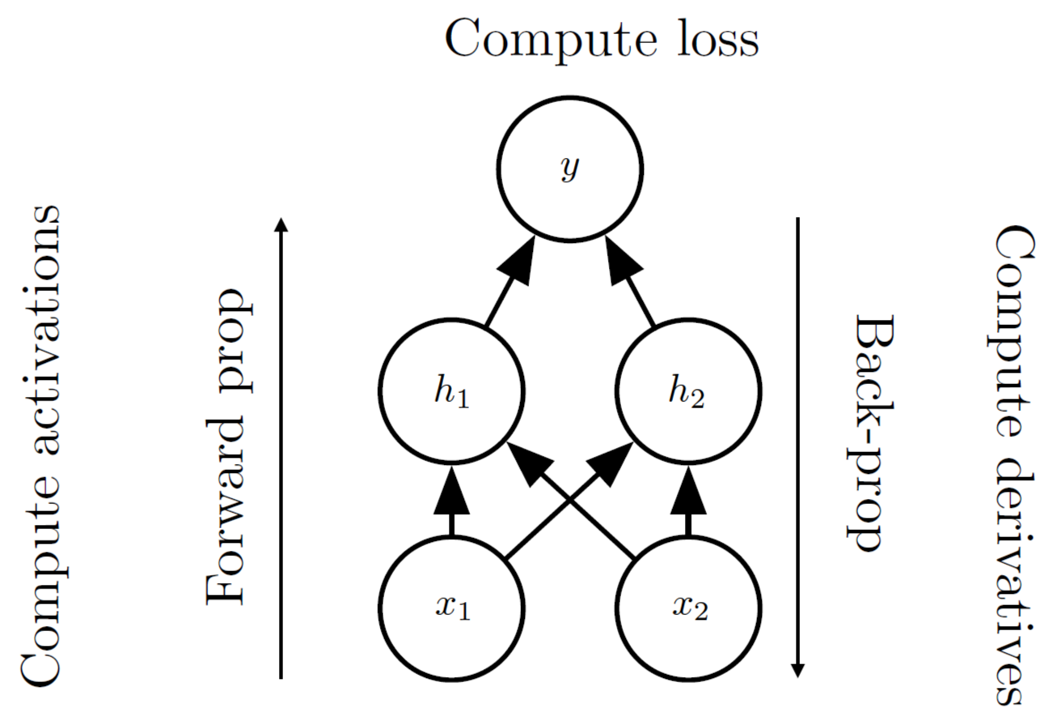
Multi-layer classifiers

- Multi-layer classifiers allow to learn non-linear relations, i.e. complex relationships such as exclusive-OR.
- Usually one to two hidden layers produce the best results.
- Trained with the back-propagation algorithm



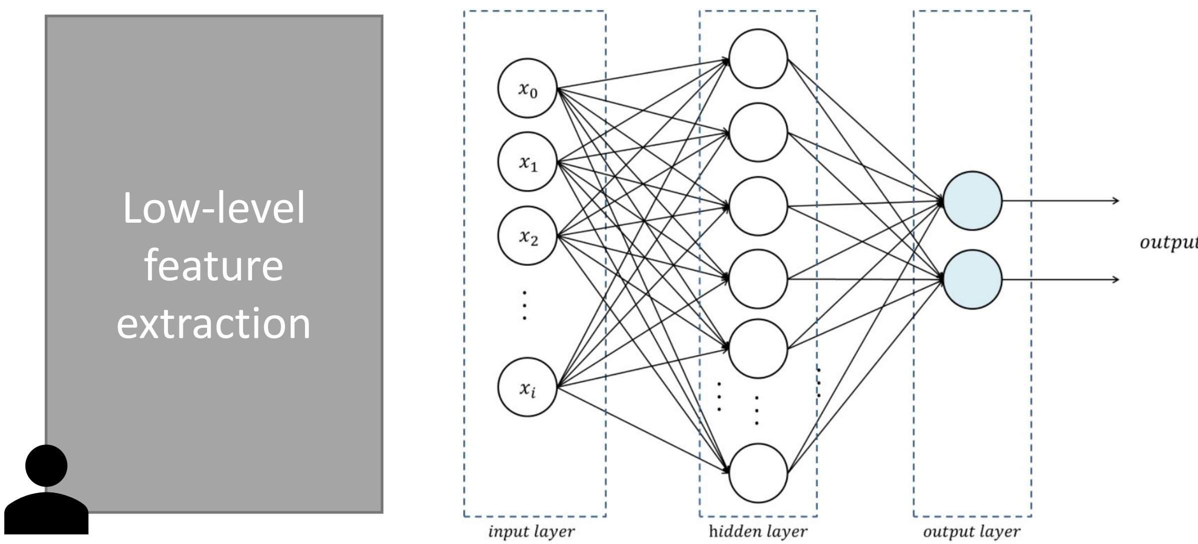
Simple back propagation



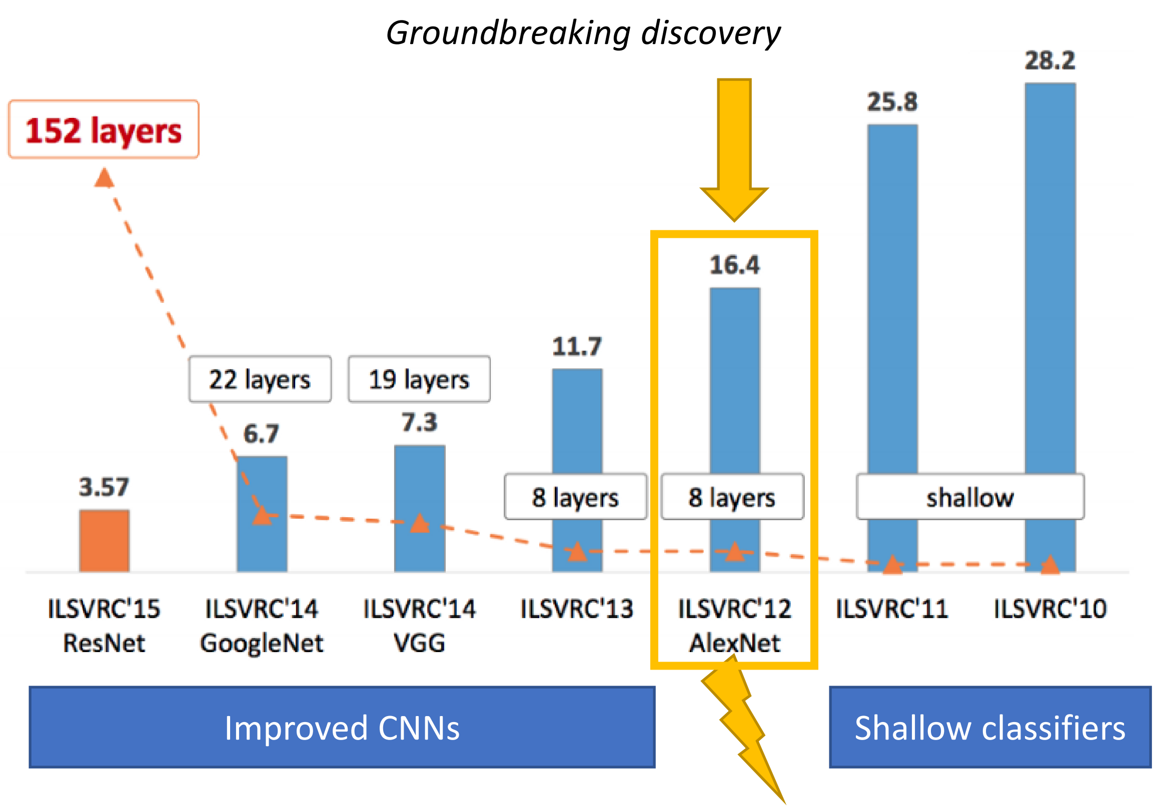


Traditional neural network architectures

- Traditionally, neural networks receive input features that are extracted from data (text, images, etc.) and are task independent.
- This creates a bottleneck: only so much can you learn from those task independente features.

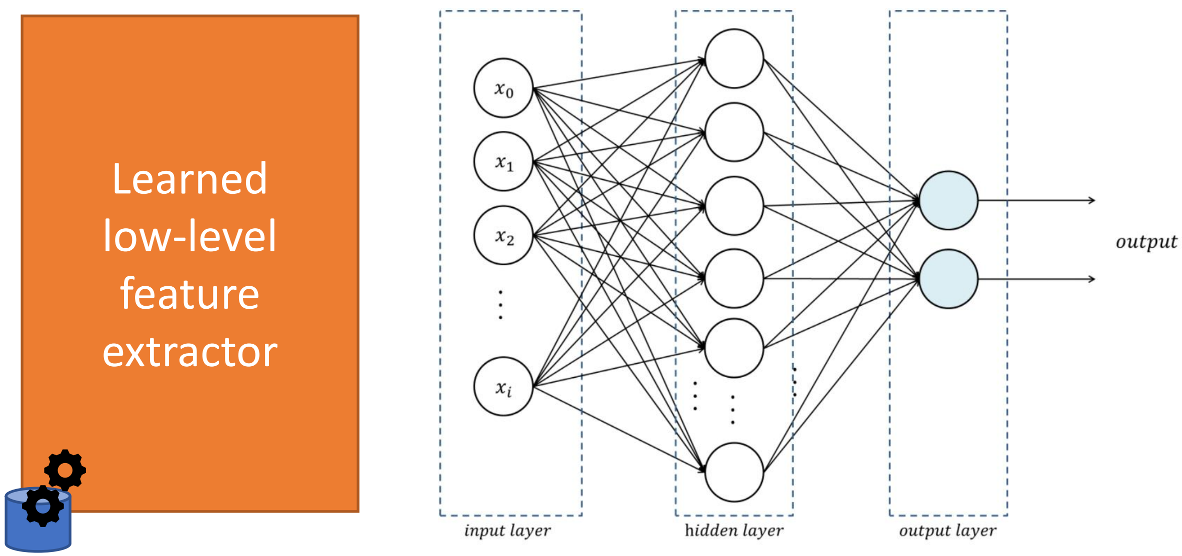


ImageNet Challenge top-5 error



Visual representation learning

- Deep architectures were introduced to learn data representations that were better suited to each task.
- Deep architectures look at the most basic data element, i.e., an image pixel or a text character, to learn new data representations.

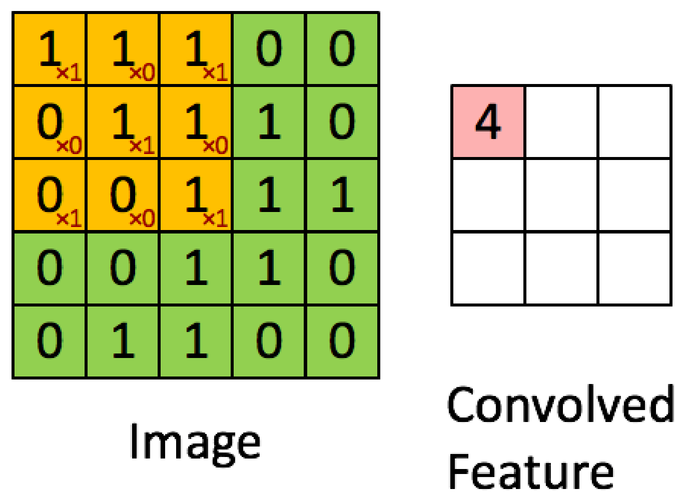


Convolution filters

- A convolution filter applies a kernel to the all image by performing the convolution operation.

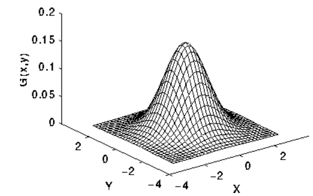
$$h * A = g(x, y) = \sum_{j=-M}^{M} \sum_{i=-M}^{M} h(i, j) \cdot A(x + i, y + j)$$

$$h(i, j) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$



Low-pass convolution filters

- The low-pass convolution filter applies a gaussian filter to the input image.



- The Gaussian filter is approximated by a kernel with a given width.

$$h_0(x, y) = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- Example:

Input image

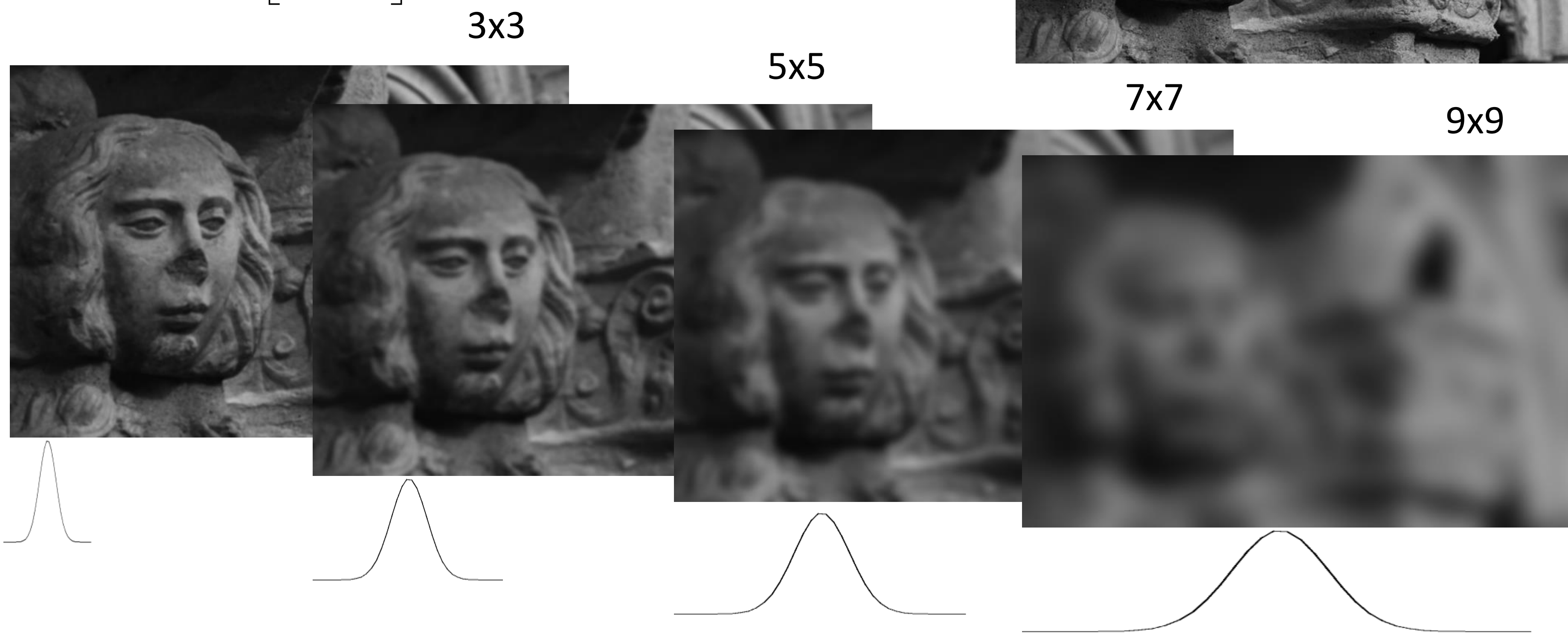
$$A(x, y) = \begin{bmatrix} 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \end{bmatrix}$$

Output image

$$g(x, y) = \begin{bmatrix} 255 & 191 & 64 & 0 \\ 255 & 191 & 64 & 0 \\ 255 & 191 & 64 & 0 \\ 255 & 191 & 64 & 0 \\ 255 & 191 & 64 & 0 \end{bmatrix}$$

Example

$$h_0(x,y) = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



High-pass convolution filters

- High pass filters aim to detect the image edges
- Different kernels are used to detect such edges at diferent scales and orientations.

Input image

$$A(x,y) = \begin{bmatrix} 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \\ 255 & 255 & 0 & 0 \end{bmatrix}$$

Output image after applying horizontal filter

Horizontal filter

$$h_h(i,j) = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad g_h(x,y) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Output image

Output image after applying vertical filter

Vertical filter

$$h_v(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad g_v(x,y) = \begin{bmatrix} 0 & 255 & 255 & 0 \\ 0 & 255 & 255 & 0 \\ 0 & 255 & 255 & 0 \\ 0 & 255 & 255 & 0 \\ 0 & 255 & 255 & 0 \end{bmatrix}$$

Output image

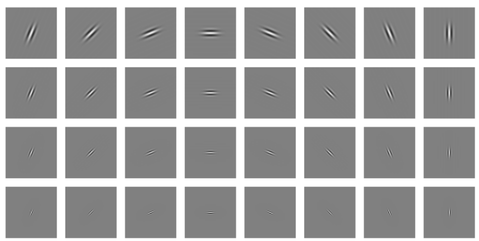
Example

$$h_v(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

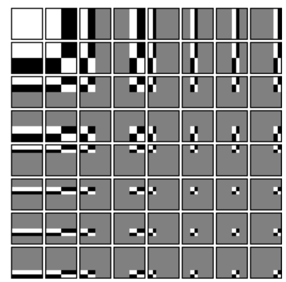
$$h_h(i,j) = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Convolution filter kernels

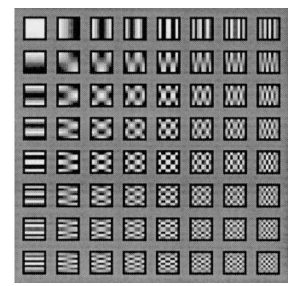
There are many diferent convolution filter kernels
that were studied over decades in the past.



Gabor



Haar



DCT

Can we learn the convolution kernels?

Yes, we can!

Deep CNNs

Image annotation and feature extraction

Convolutional Networks

- Scale up neural networks to process very large images / video sequences
  - Sparse connections
  - Parameter sharing

- Automatically generalize across spatial translations of inputs

- Applicable to any input that is laid out on a grid (1-D, 2-D, 3-D, …)

Convolutional Network Components

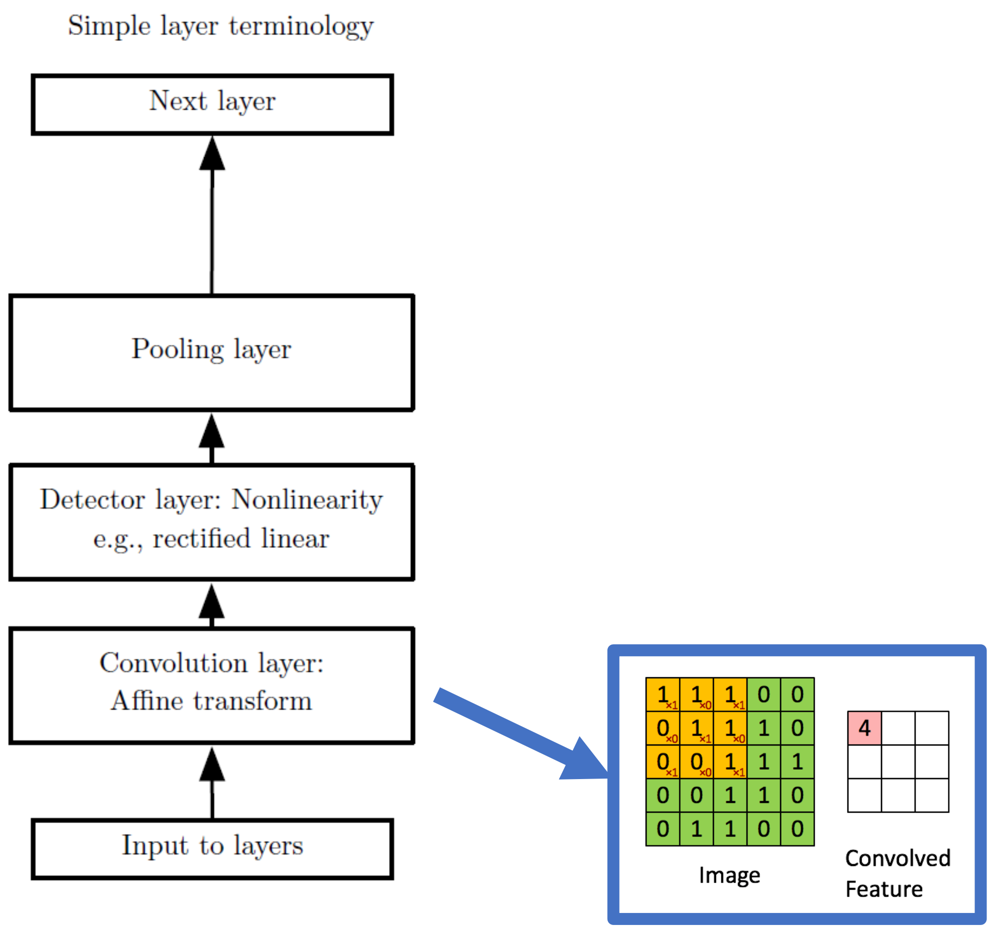
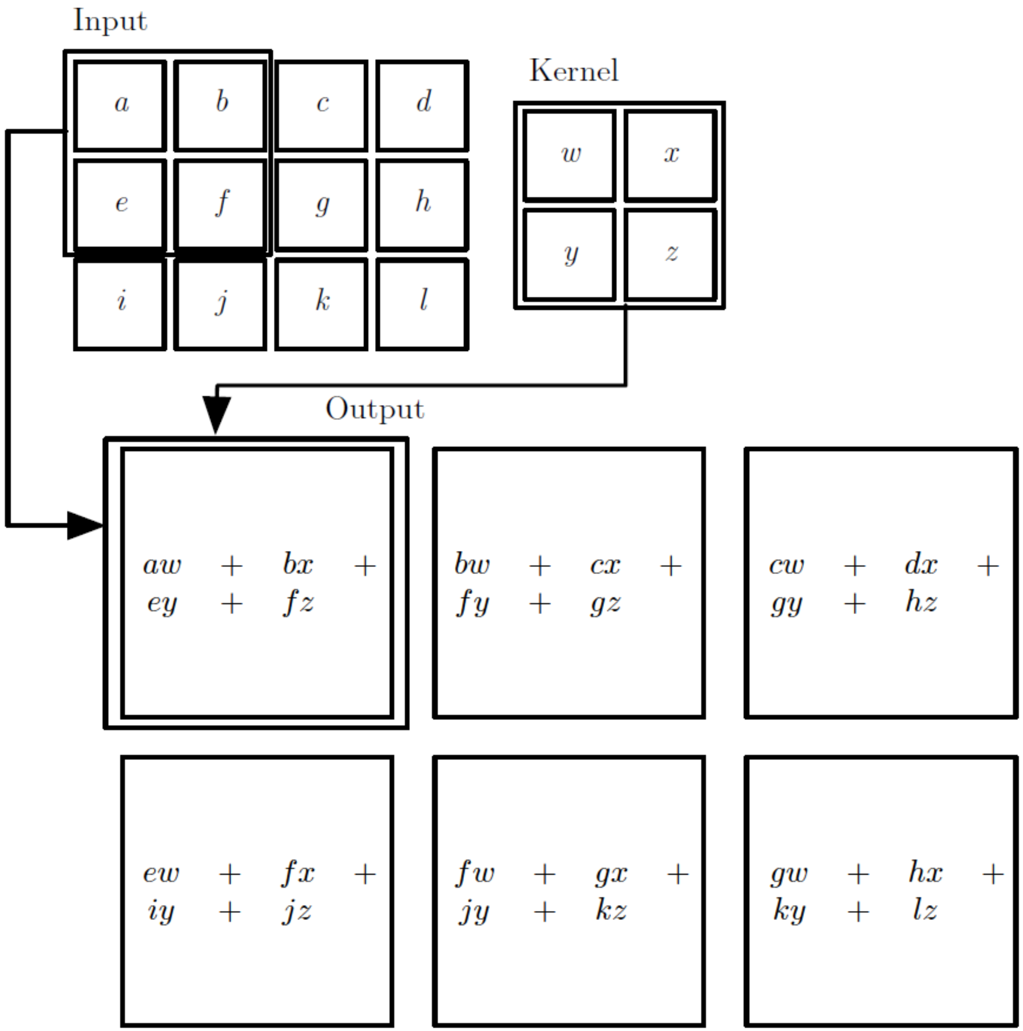


Figure 9.7

2D Convolution



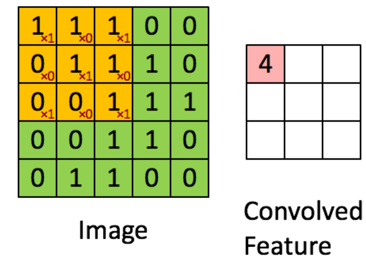


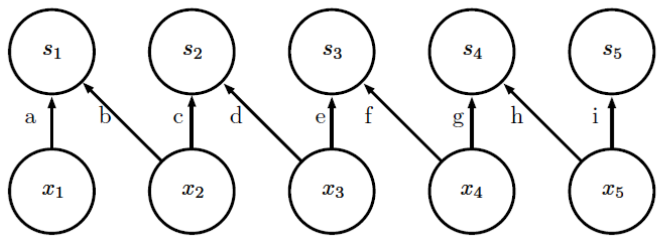
Figure 9.1

Types of connectivity

Kernel:

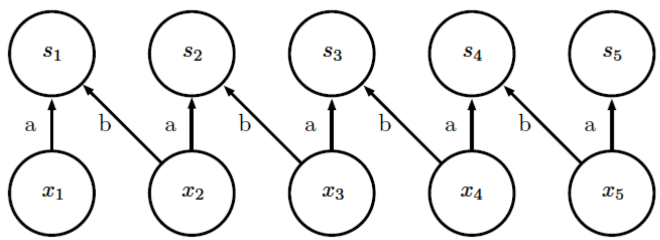
1	-1

a,b,c,d=?

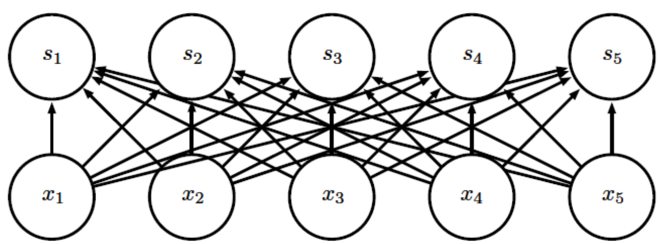


Local connection: like convolution, but no sharing

a,b=?

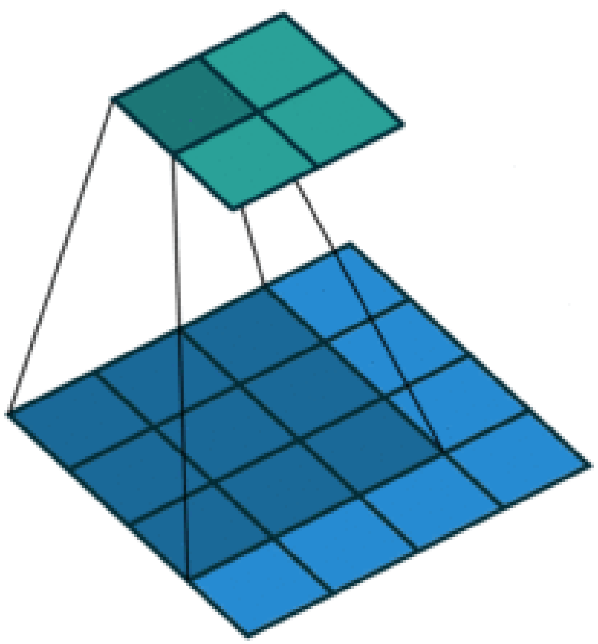


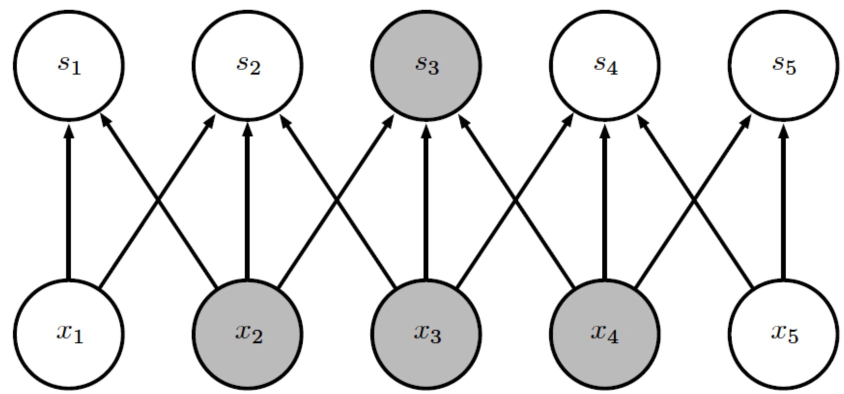
Convolution



Fully connected

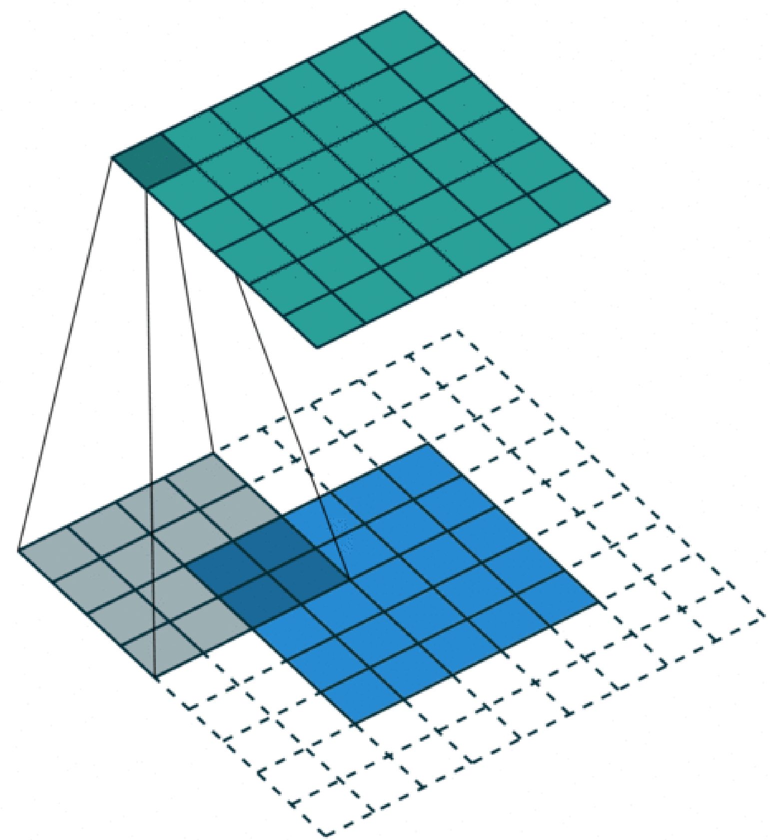
Convolution as a NN





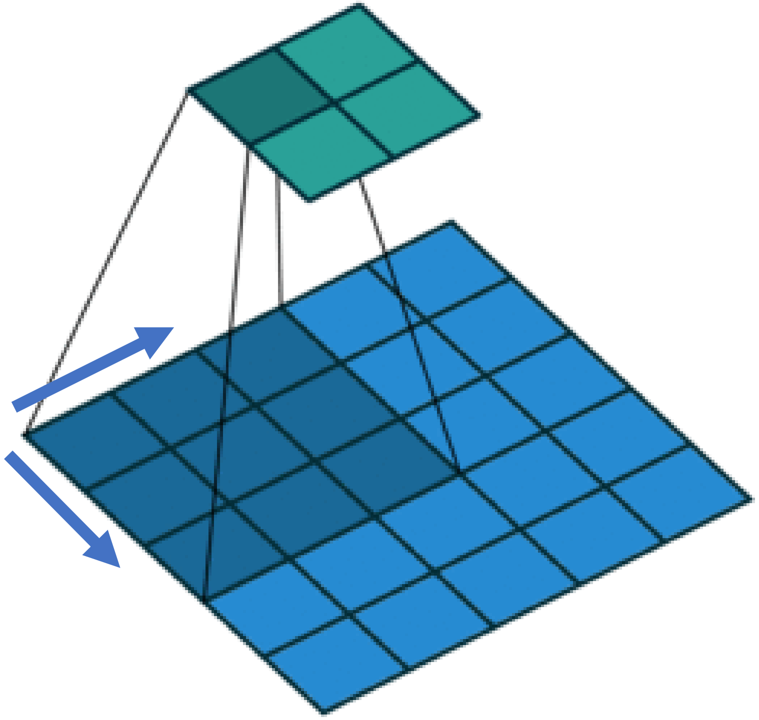
Padding

- Padding extends an image with adds zeros to let the convolution center run over the entire image.

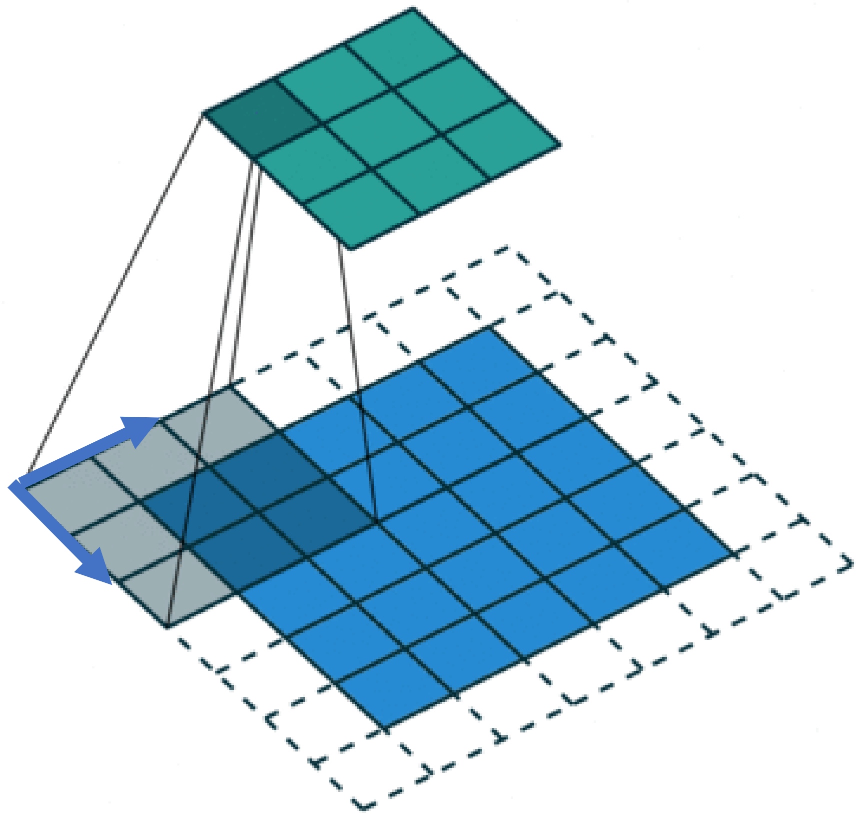


Strides

- Defines the number of pixels a convolution moves in each direction.



Padding with strides



Convolutional Network Components

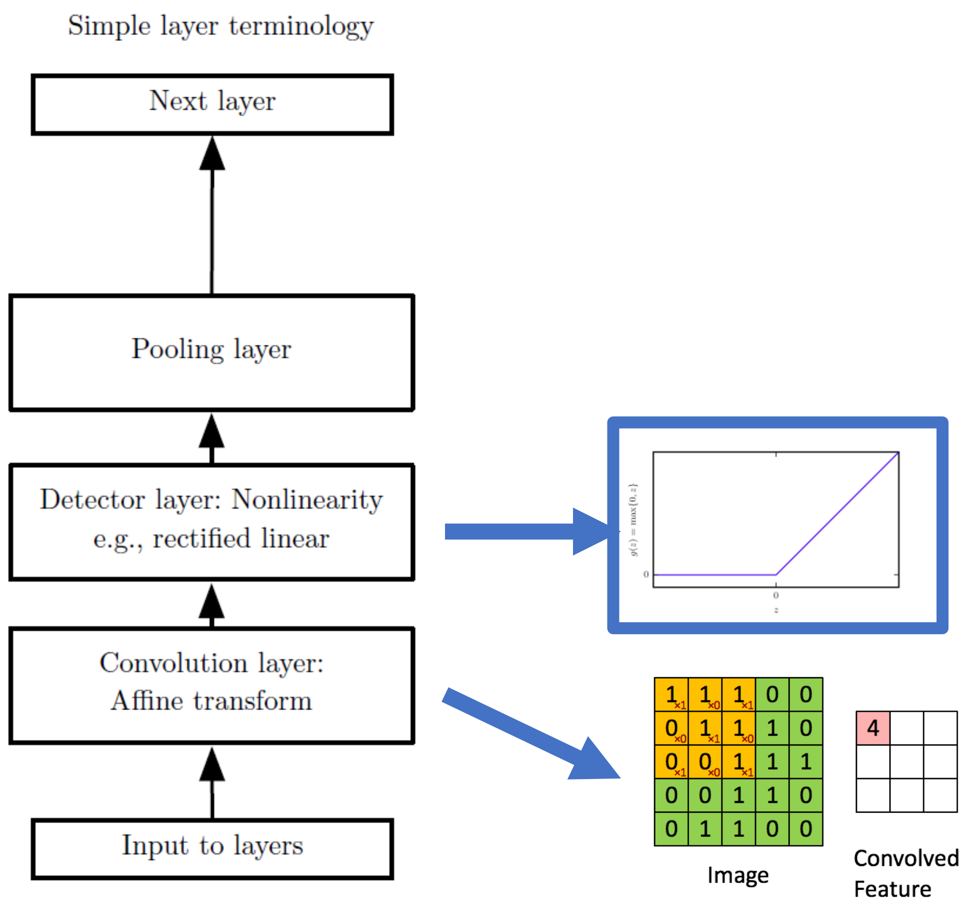
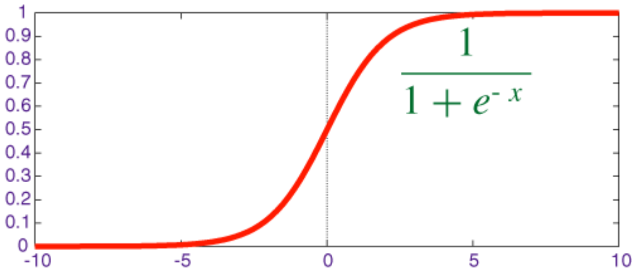


Figure 9.7

Softmax

- The softmax function was quite popular as the activation function of neural networks.
- It is diferentiable in all points
  - It is convenient from a mathematical point of view
- It can easily saturate for high values of inputs
  - Prevents passing information between layers

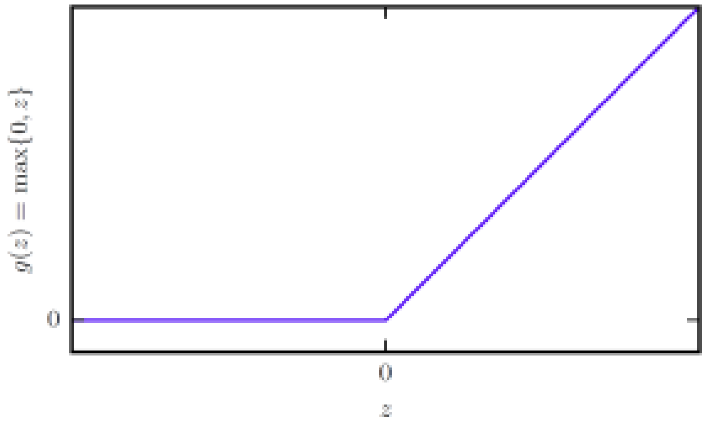


Rectified linear unit (ReLU)

- Rectified linear activation:

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

- Brings several advantages over traditional softmax for hidden layers:
  - Never saturates, i.e. never looses information between layers
  - Gradient is constant, i.e. faster training
  - Forces sparsity, thus removes contribution from noisy units



Convolutional Network Components

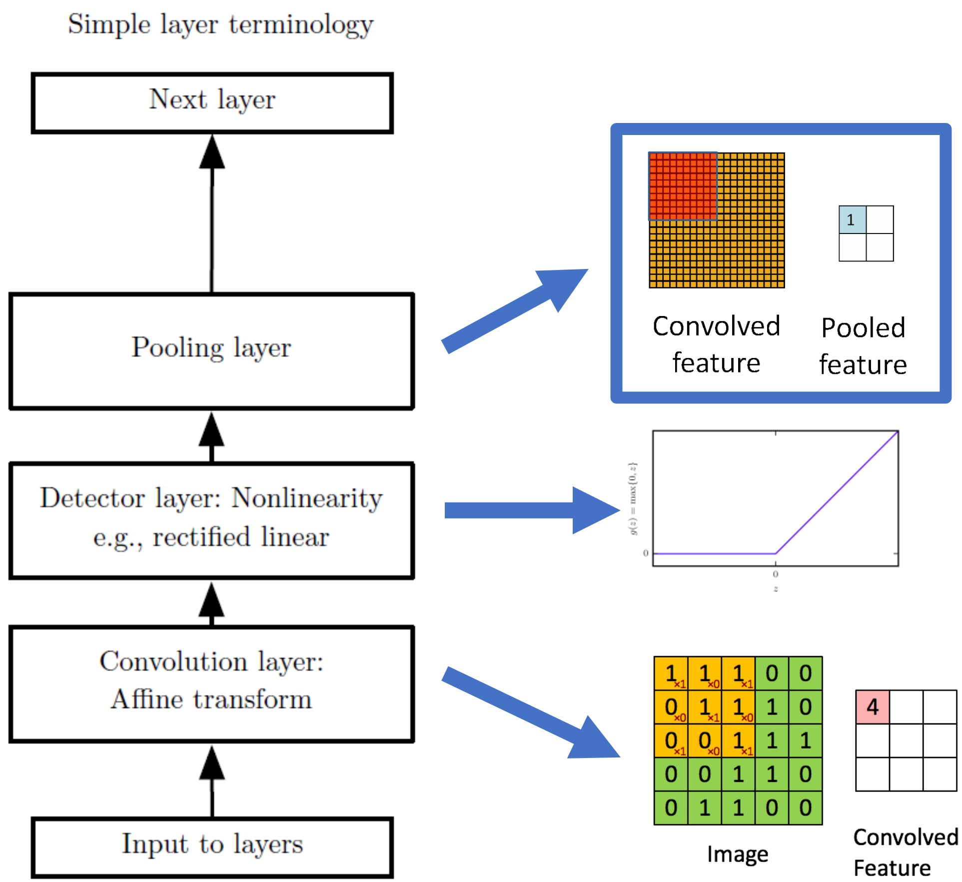


Figure 9.7

Pooling

- Goals of pooling:
  - Downsampling
  - Translation invariant
  - Feature extraction

- Pooling strategies:
  - Max pooling
  - Min pooling
  - Average pooling

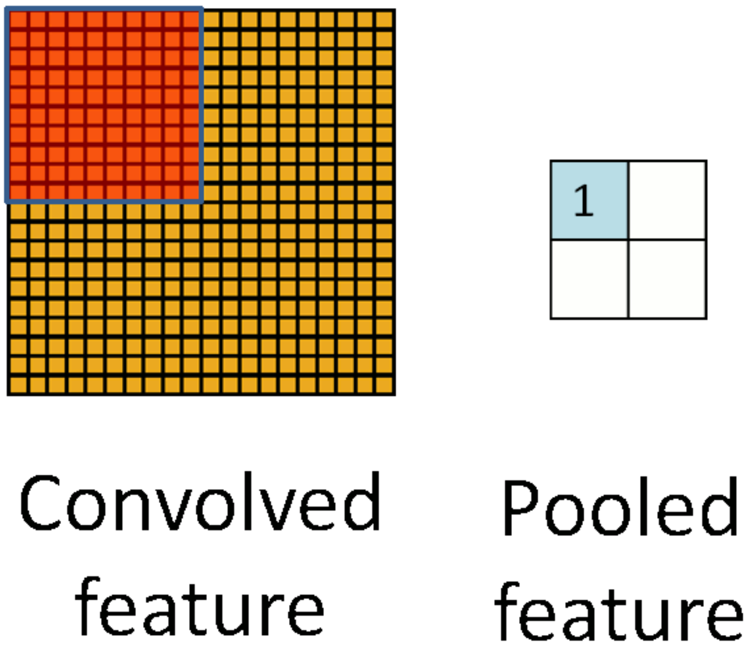


Figure 9.8

Convolutional Network Components

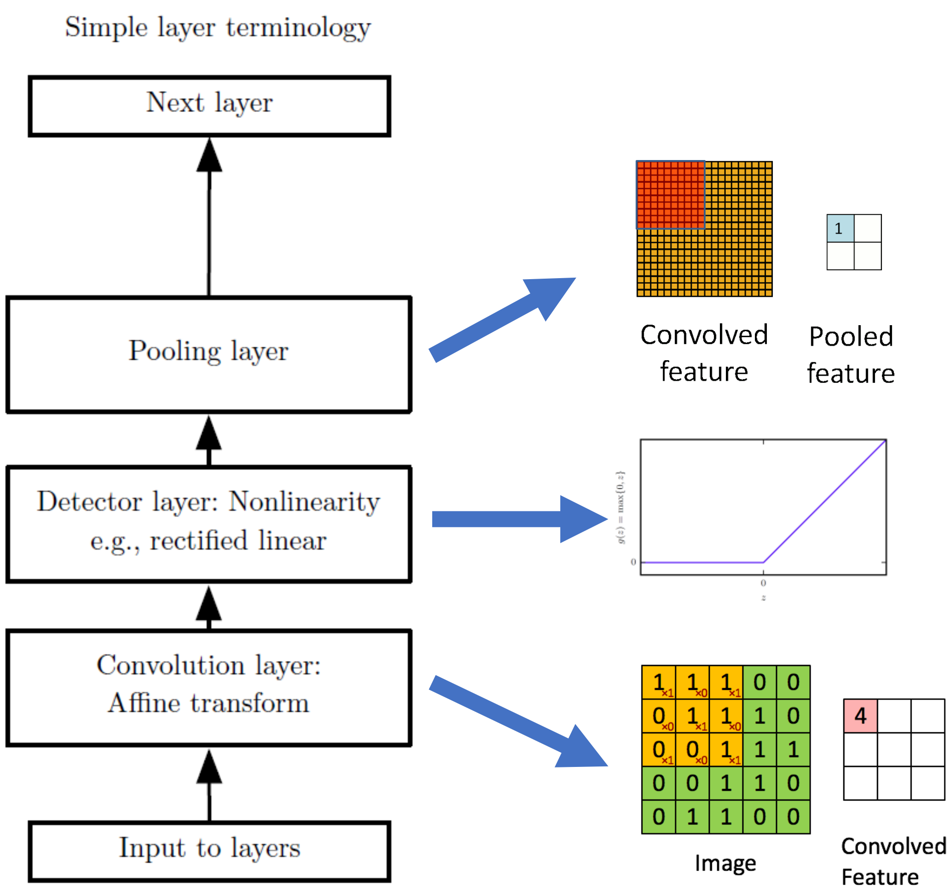
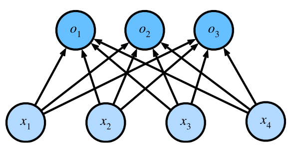
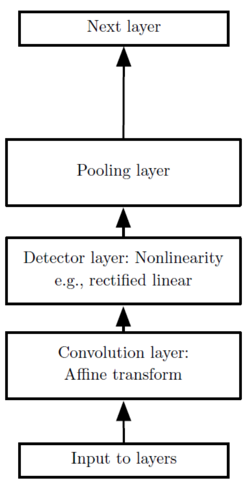


Figure 9.7

Linear layer and softmax layer

- The output of the pooling layer is flattened into a vector
- The output of the linear layer is then fed into a softmax function

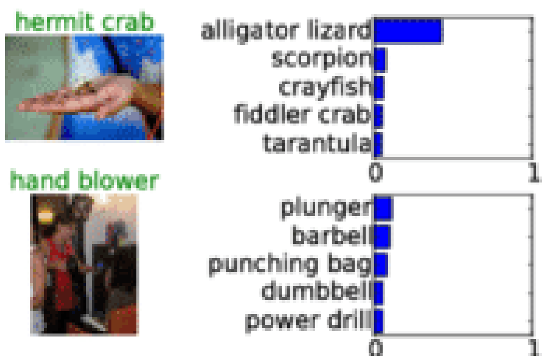




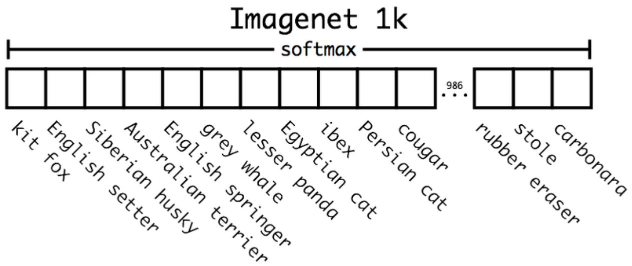
$$\text{softmax}([z_1, z_2, \dots, z_n]) = \left[\frac{e^{z_1}}{\sum_i e^{z_i}}, \frac{e^{z_2}}{\sum_i e^{z_i}}, \dots \frac{e^{z_n}}{\sum_i e^{z_n}},\right]$$

Learning a keyword probability distribution

- The softmax will return a probability distribution over all keywords for each image

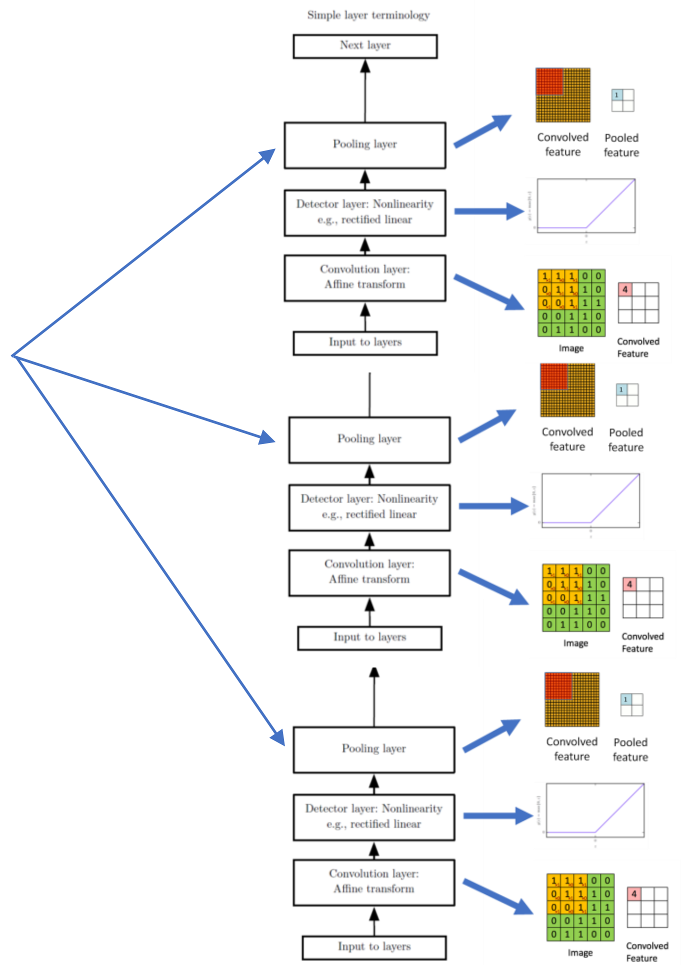


$$\text{softmax}([dog, cat, \dots, bird]\,) = \left[\frac{e^{dog}}{\sum_i e^{z_i}}, \frac{e^{cat}}{\sum_i e^{z_i}}, \dots \frac{e^{bird}}{\sum_i e^{z_n}},\right]$$



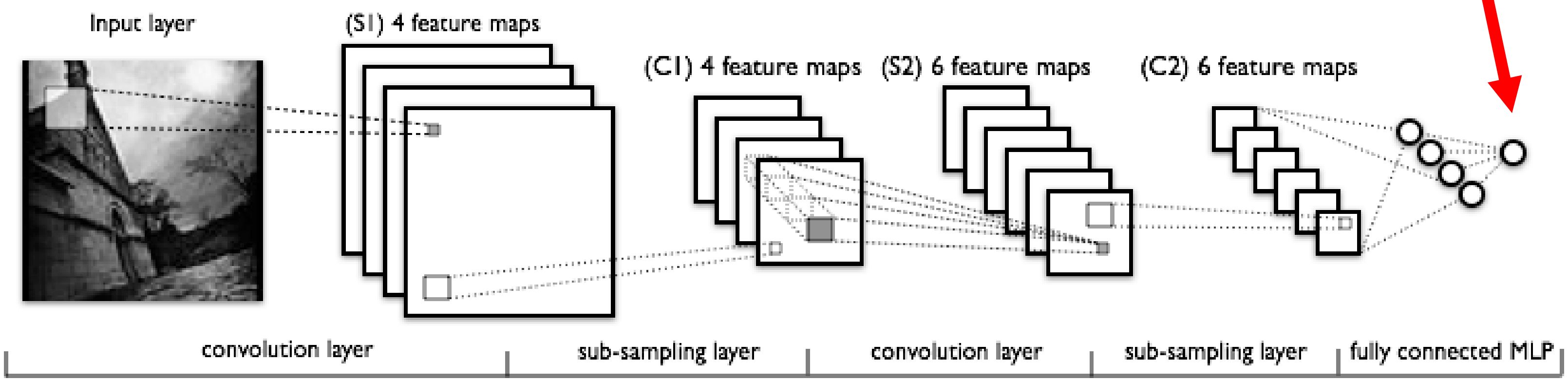
Extracting image features for visual search

- Deep learning architectures learn hierarchies of data representations
- On each layer, we can extract the data, do a greedy pooling and then flatten the data.
  - This creates an image feature vector that we can use for searching.



Examples of CNN architectures

Output neurons correspond to ImageNet concepts



Seminal CNN architecture - 1998

PROC. OF THE IEEE, NOVEMBER 1998 7

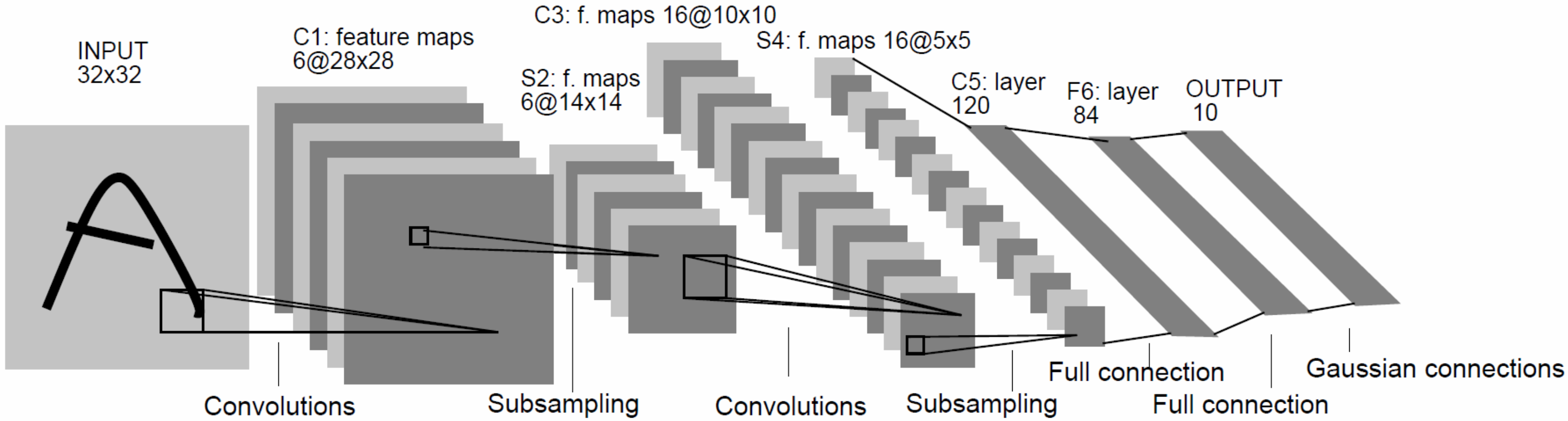
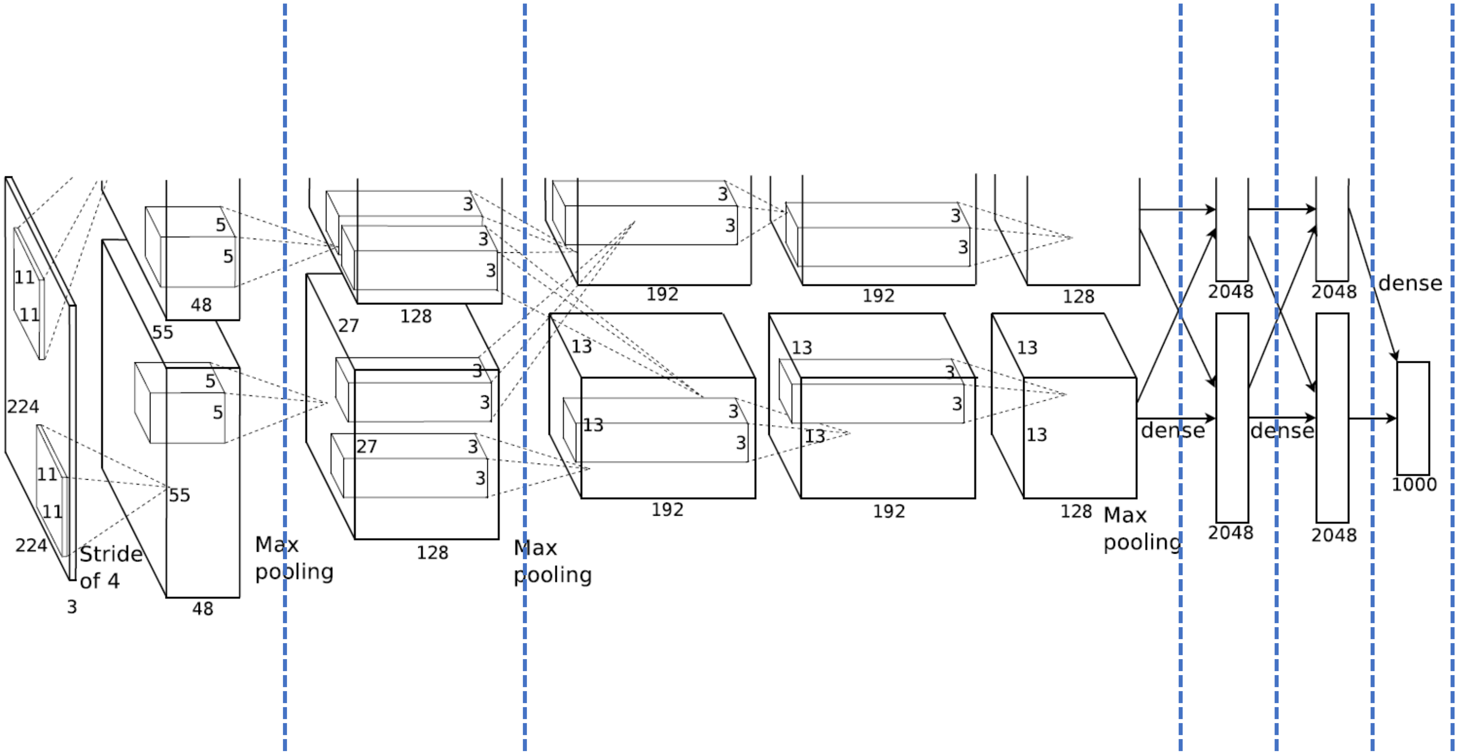


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

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

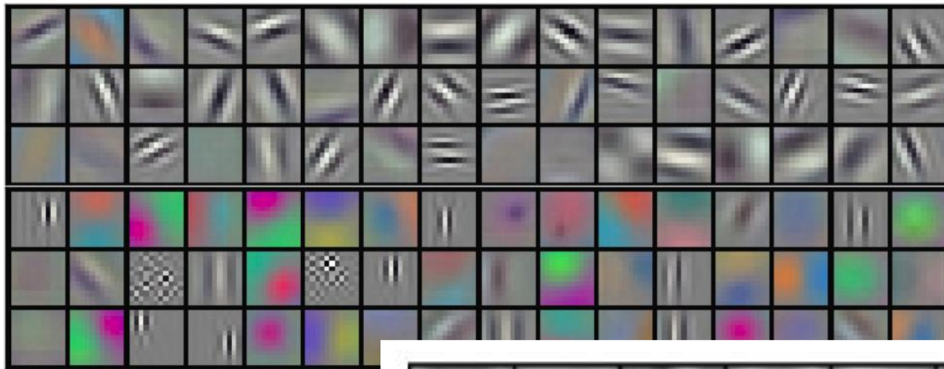
AlexNet 2012



Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

Example for face detection

Low level CNN kernels

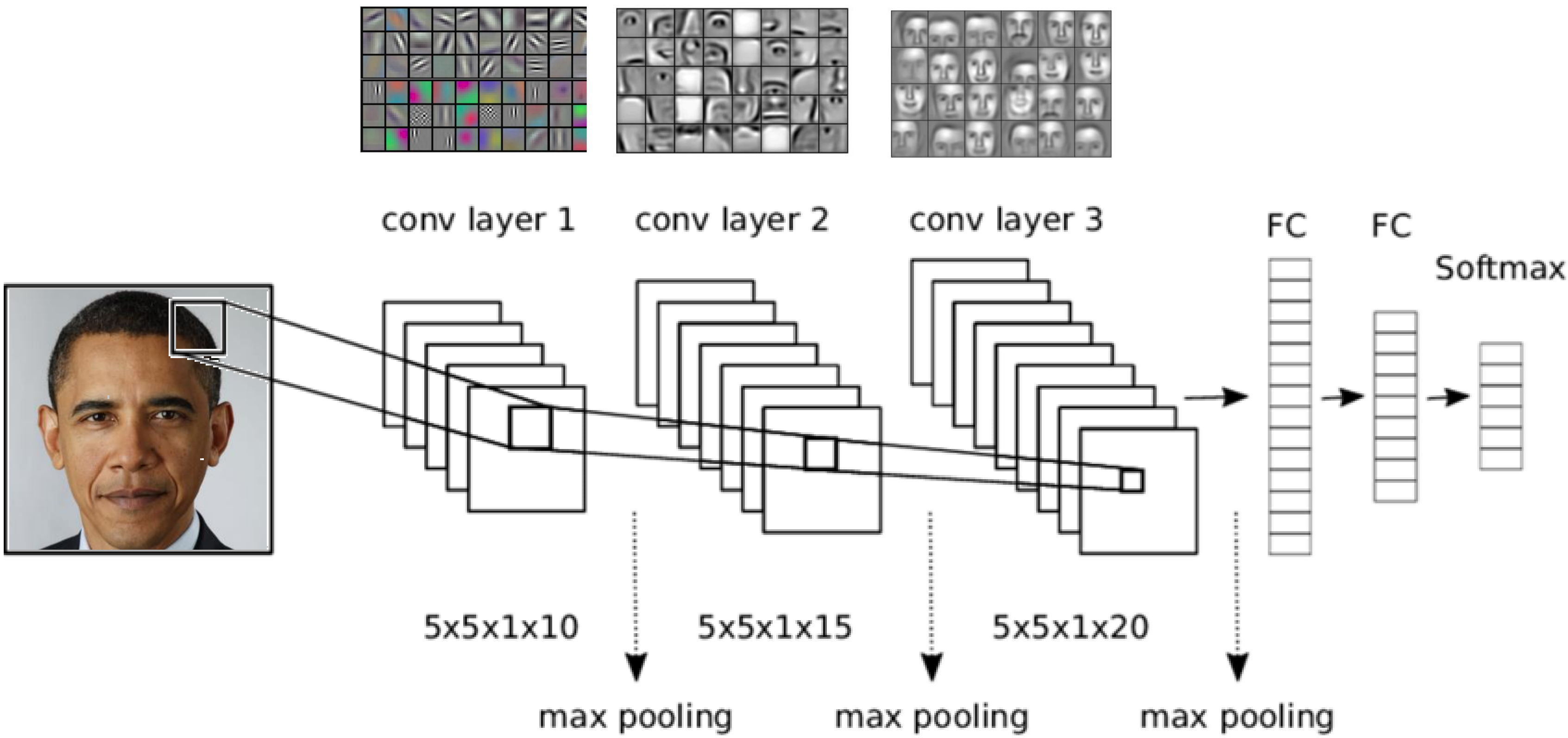


Mid level CNN kernels

High level CNN kernels

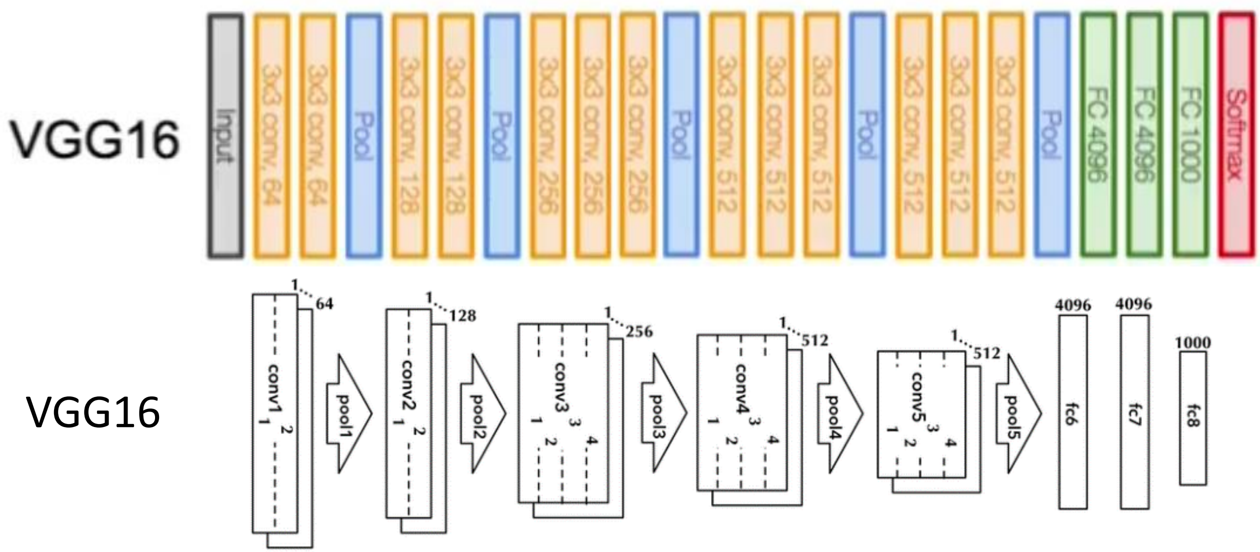
Lee, H., Grosse, R., Ranganath, R., & Ng, A. Y. (2011). Unsupervised learning of hierarchical representations with convolutional deep belief networks. *Communications of the ACM*, *54*(10), 95-103.

Each CNN filter kernel locates a pattern



VGG 16 architecture

Output neurons correspond to ImageNet concepts

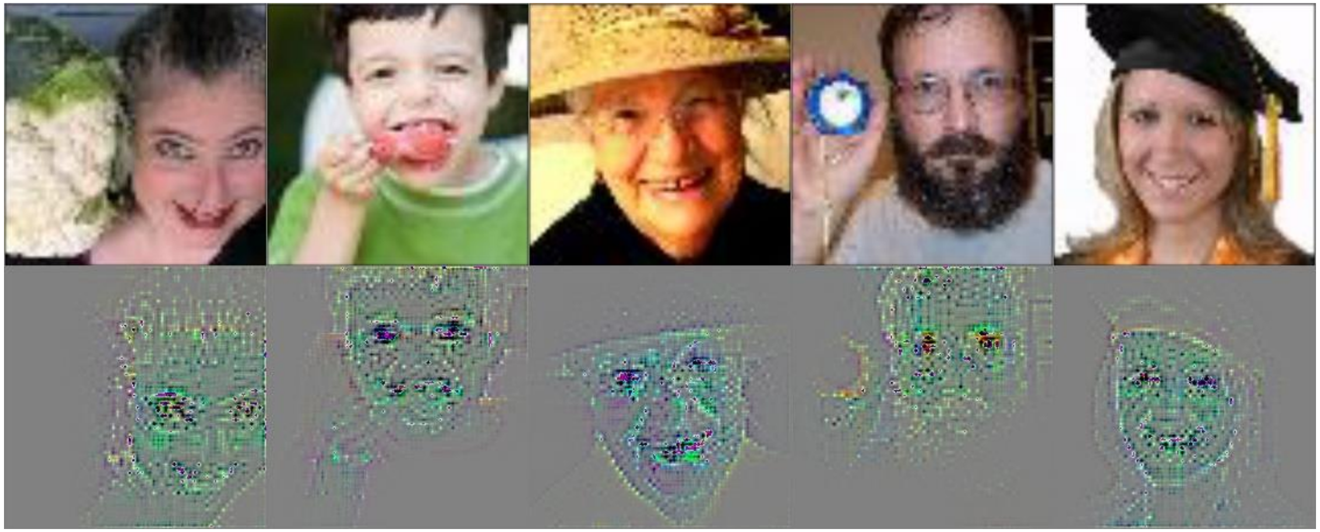


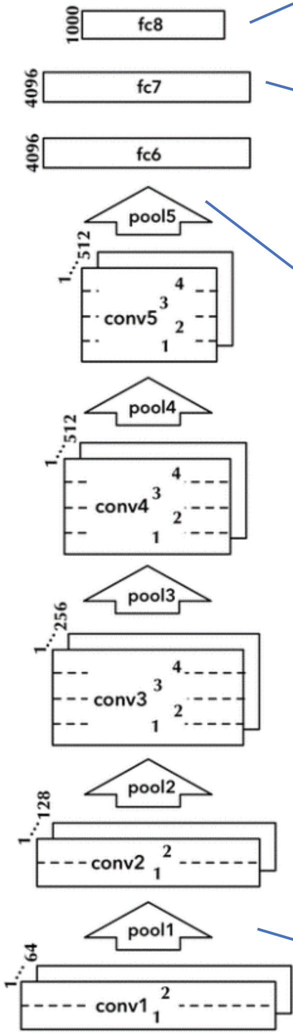
Output neurons correspond to ImageNet concepts

Visualizing VGG16

https://github.com/yosuah/vgg_deconv_vis

High level neuron from the fifth convolution block





VGG16: Softmax output – 500 imagens

VGG16-fc7: **metric="euclidean" – distâncias superiores a 0,98**:

VGG16-pool5: **metric="euclidean" – distâncias superiores a 1**:

VGG16-pool1: **metric="euclidean"**:

Summary and readings

- Learning data representations
  - Convolution operation
  - ReLU activation
  - Pooling
  - Residual Networks

- Understand visual data representations:
  - low-level layers, mid-level layers and high-level layers

- Bibliography:
  - http://d2l.ai/chapter_convolutional-neural-networks/index.html
  - http://d2l.ai/chapter_convolutional-modern/index.html