UNIVERSITÀ DEGLI STUDI DI PADOVA

Introduction to Deep Learning

Stefano Ghidoni

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

INTELLIGENT AUTONOMOUS SYSTEMS LAB

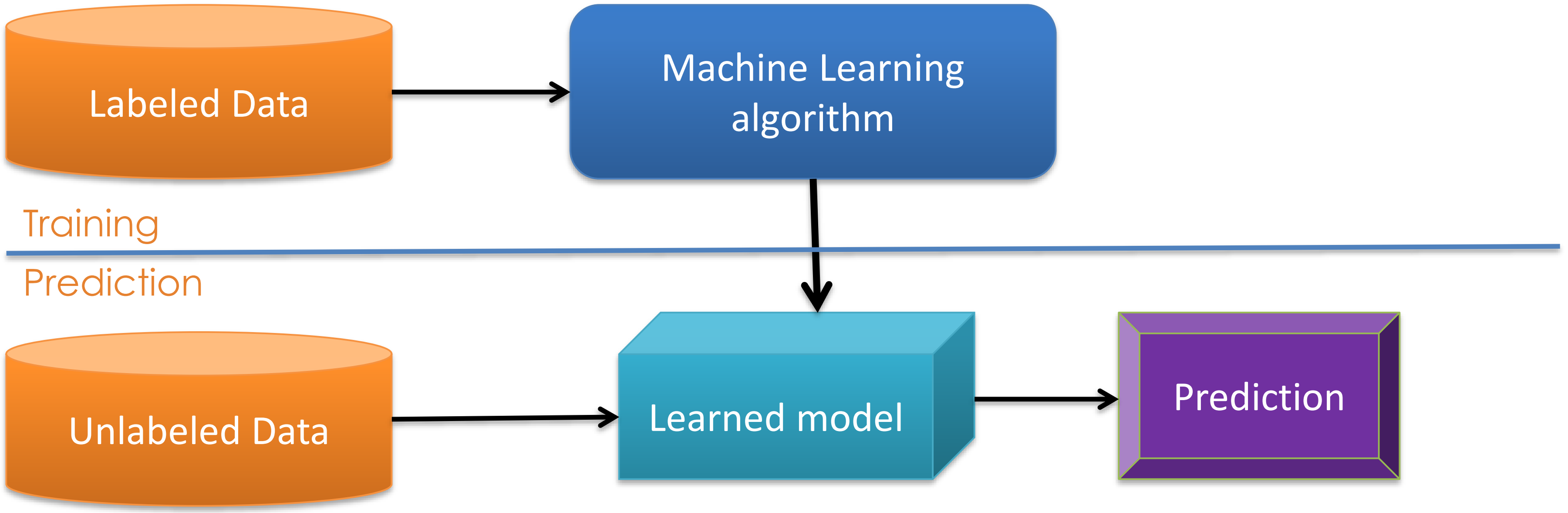
Agenda

- ML and DL
- Intro to deep learning
- Convolutional Neural Networks (CNNs)
- Image classification using DL

Slides by Pietro Zanuttigh and Stefano Ghidoni, Ross Girshick, Ismini Lourentzou, Sasank Chilamkurthy, Siddharth Das, Bharath Ramsundar, medium.com, Gonzalez book

Machine Learning

- A field of computer science that investigates how it is possible to learn from data
  - Goal: make predictions from data
- Learning an algorithm vs synthesizing an algorithm



Learning framework

- Input: dataset of input and target pairs
- Goal: find f* that best approximates the unknown function f
- f* is parametric, optimal parameters need to be found
- Supervised learning: desired outcome is available
- Unsupervised learning: desired outcome is not provided

A simple example: linear model

- A simple case – linear model

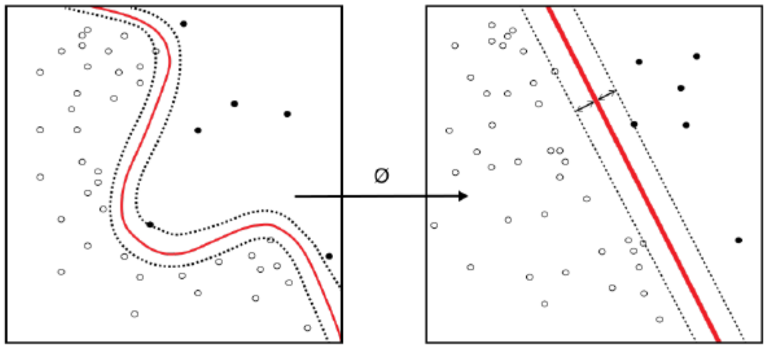
$$f(\boldsymbol{x_i}) = W\boldsymbol{x_i}$$

Where $W: R^n \to R^m$ is a linear operator mapping the observation space into the target space

- Simple and easy to understand, closed form solution may be available
- But: non-linear phenomena cannot be accurately described

Kernel function

- Non-linear phenomena can be linearized using a kernel function
- Non-linear mapping + linear model
- Easy to implement / relies on simple models
- Kernel is often application-dependent, hard to generalize



ML inputs

- The input of machine learning is data
  - Representation of a given phenomenon
  - CV: representation of an image, an object, a patch
- What kind of representation can be provided to a ML algorithm in the computer vision context?

ML inputs

- What kind of representation can be provided to a ML algorithm in the computer vision context?
- In general: domain-specific representations should be generated by means of CV algorithms
- Classification follows, based on such representation

ML inputs

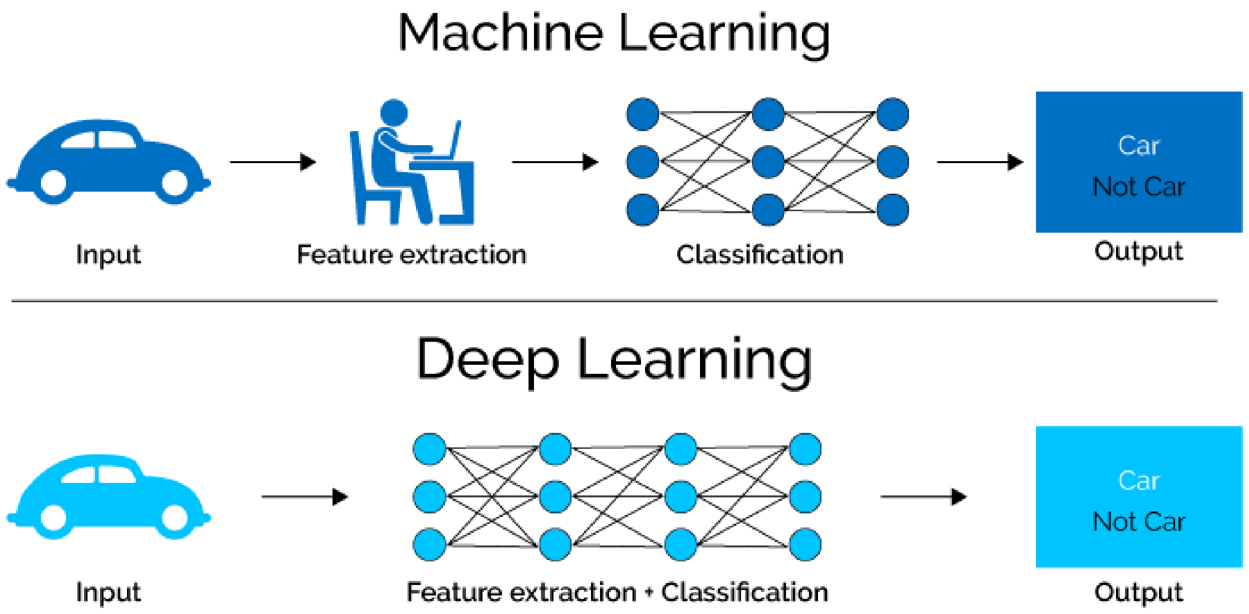
- Think of a CV-related representation to be presented to a ML algorithm
  - What is the first idea that comes to your mind?

ML inputs

- What is the first idea that comes to your mind?
- Curse of dimensionality: learning is not effective when the input space has too many dimensions
  - Too many depends on the ML method
  - The curse is due to many different effects
- Learning is no magic, it's similar to an optimization problem

Deep learning

- ML build on image representation provided by some algorithms
- DL also learns data representations
  - Very effective in learning patterns
  - Data is represented by means of a hierarchy of multiple layers – multiple levels of representation
  - Needs huge amount of data to be trained



https://www.xenonstack.com/blog/static/public/uploads/media/machine-learning-vs-deep-learning.png

DL – advantages

- Handcrafted features might be incomplete and require a lot of human work to be designed and tested
- Learned features are adapted to the input data
- Data representation and classification into one single network – **end-to-end learning**

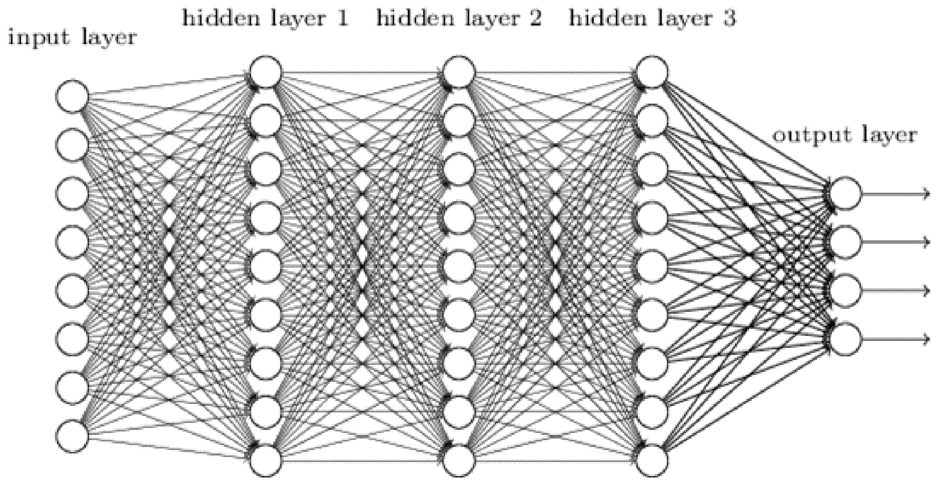
Deep Learning basics

Deep Neural Networks (DNNs)

- DNN: a composition of several simple functions, called layers

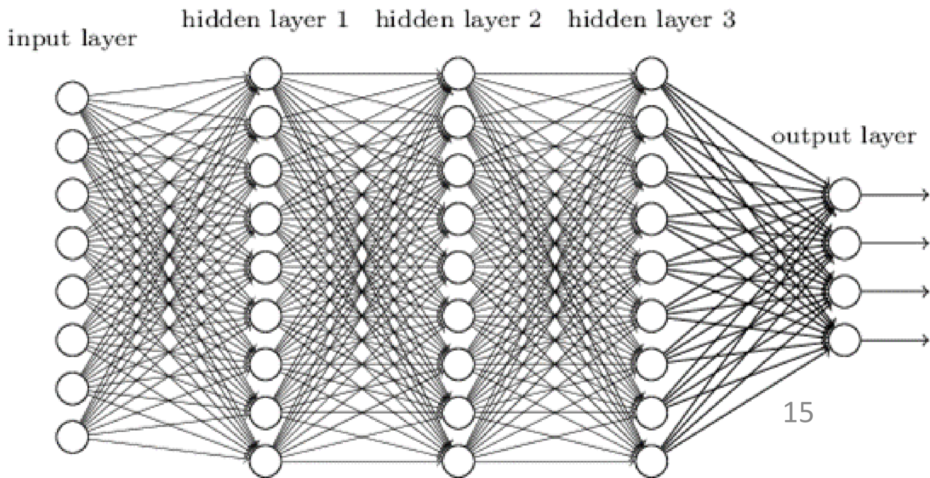
$$f(x_i) = f_{\theta_l} \circ h \circ f_{\theta_{l-1}} \circ h \circ \cdots \circ f_2 \circ h \circ f_1 \circ h$$

  - h is the activation function
  - f is a function depending on the specific layer
- Each layer is made of a set of neurons
- Includes non-linear elements
  - What happens if the layers are all linear?
- The hierarchical topology generates a hierarchical abstraction



Neurons

- Input to the neurons of one layer are the output of the neurons in the preceding layer – feedforward network
- Output of a neuron: activation function applied to a weighted average of the inputs plus a bias
- Several types of activation functions



Inside a neuron

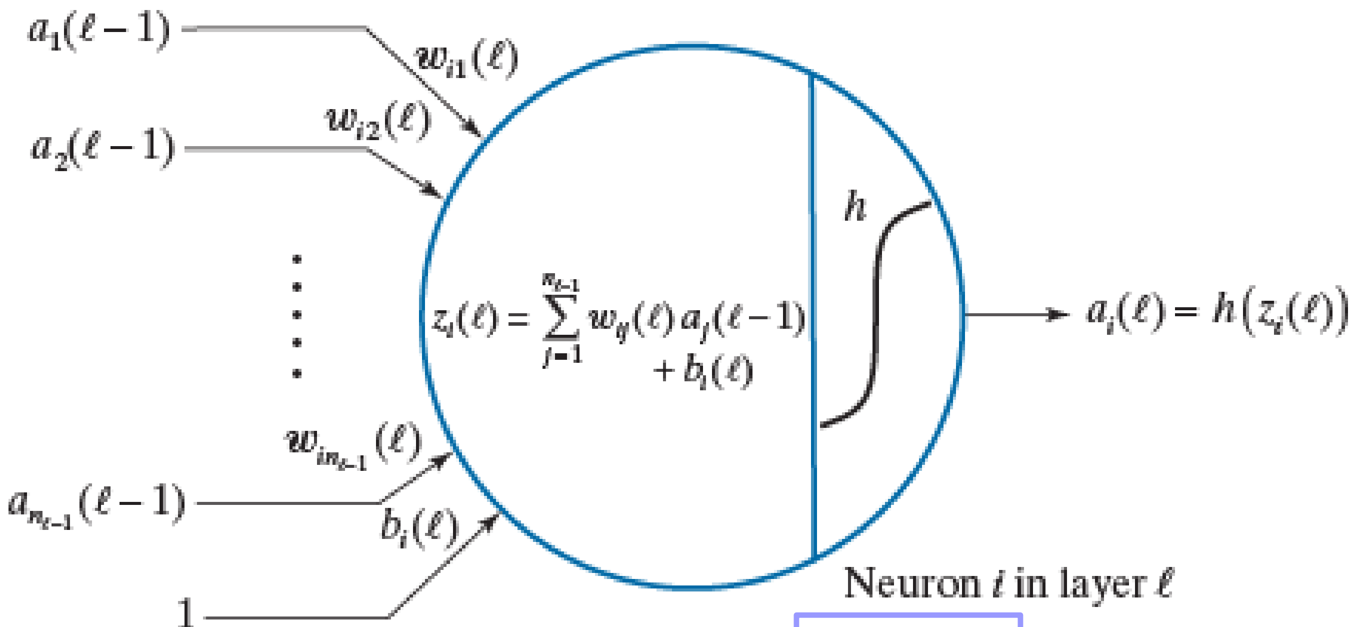
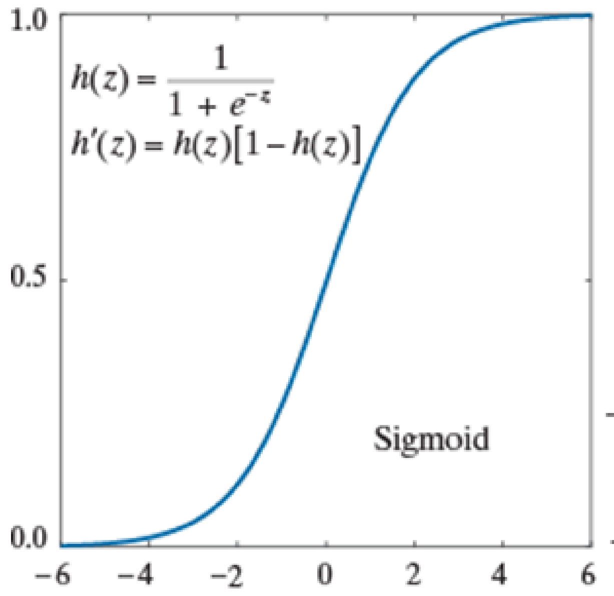

$a_1(\ell-1)$
$w_{i1}(\ell)$
$a_2(\ell-1)$
$w_{i2}(\ell)$
$w_{in_{\ell-1}}(\ell)$
$a_{n_{\ell-1}}(\ell-1)$
$b_i(\ell)$
1
$z_i(\ell) = \sum_{j=1}^{n_{\ell-1}} w_{ij}(\ell)\, a_j(\ell-1) + b_i(\ell)$
h
$a_i(\ell) = h(z_i(\ell))$
Neuron i in layer ℓ

FIGURE 12.29 Model of an artificial neuron, showing all the operations it performs. The "ℓ" is used to denote a particular layer in a layered network.

$$a_i(l) = \underbrace{h}_{\text{Activation function}}\left(\sum_{j=1}^{n_{l-1}} \overbrace{w_{ij}}^{\text{Weights}}(l)\,a_j(l-1) + \underbrace{b_i}_{\text{Bias}}(l)\right)$$

Activation: sigmoid

- Sigmoid function: $R^n \rightarrow [0,1]$
- Smooth output
- Representation of the firing rate of a neuron
- Sigmoid neurons saturate and kill gradients
  - Poor learning close to 0 or 1



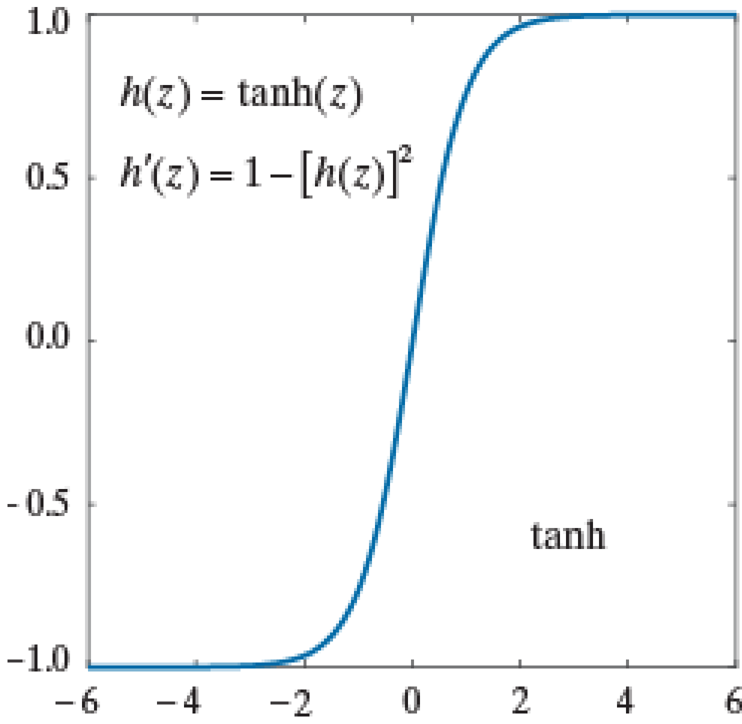
Activation: tanh

- Tanh function:

$$R^n \to [-1,1]$$

- Saturation: similar to sigmoid
- Output is zero-centered
- Tanh is a scaled sigmoid:

$$\tanh(x) = 2sigm(2x) - 1$$

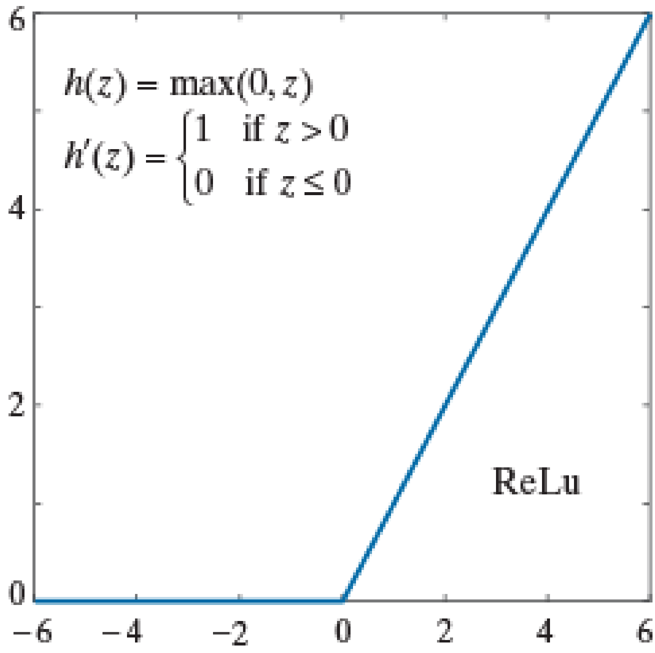


Activation: ReLU

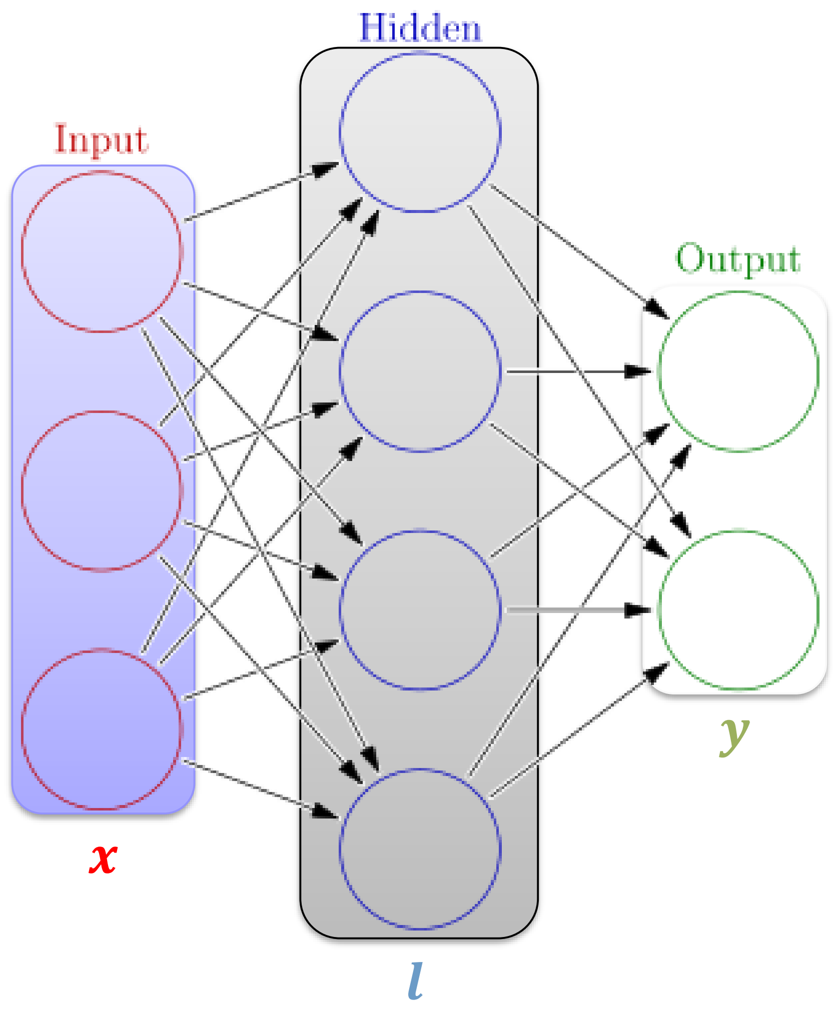
- Simpler function:

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

- Range: $R^n \rightarrow R^n_+$
- Faster training thanks to its non-saturating nature
  - Prevents the gradient vanishing problem
- Lighter computation



A NN layer



Weights

$$l = h(W_1 x + b_1)$$

$$y = h(W_2 l + b_2)$$

Activation functions

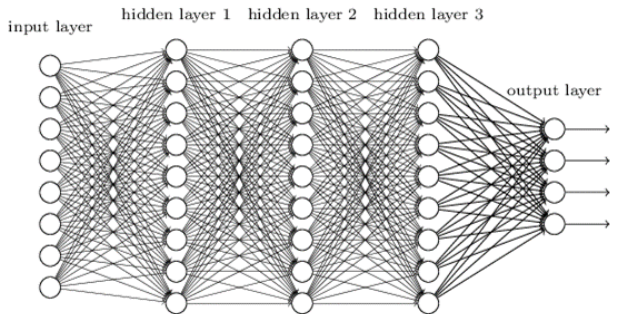
Bias

4 + 2 = 6 neurons (not counting inputs)
[3 x 4] + [4 x 2] = 20 weights
4 + 2 = 6 biases

26 learnable **parameters**

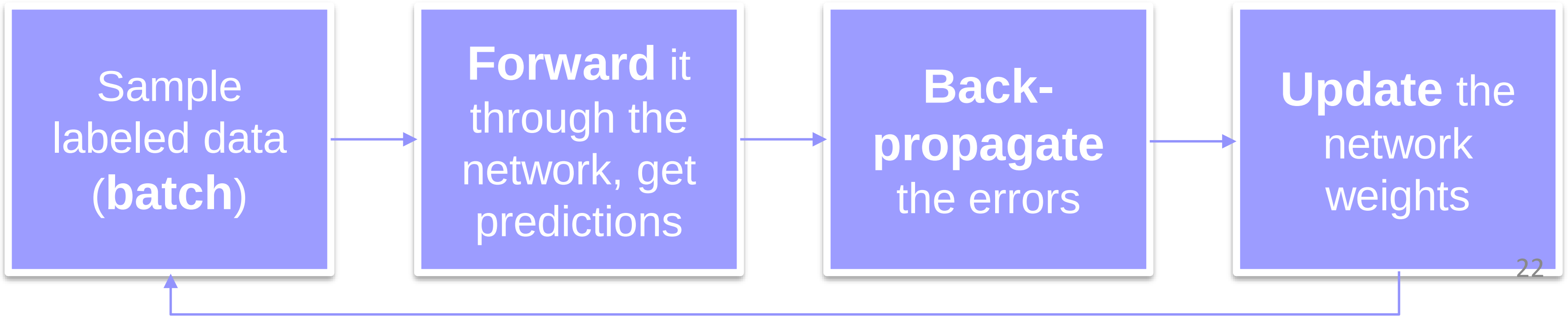
Fully connected network

- Fully connected multilayer feedforward network
  - Several layers (deep)
  - No loops (feedforward)
  - High number of connections
- A typical usage: classification network
  - # output neurons is # classes
  - Max value: class selection



Training

- Definition of a cost function
- Propagate data through the network
- Measure the error between desired and actual values using the cost function
- Update the weights using backpropagation


Sample labeled data (**batch**)
Forward it through the network, get predictions
Back-propagate the errors
Update the network weights

The output layer

- The output layer depends on the classification problem we are solving
- If we have 2 classes, several representations are possible:
  - One value describing the class
  - Two values describing the score of each class
- Can be generalized to N classes

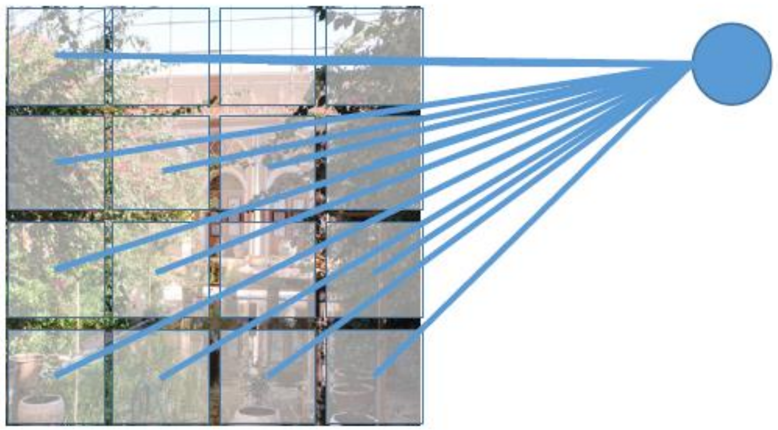
Objective function: 2 classes

→ ALFA ROMEO → FIAT	→ **1,98m** → **2,16m**
Classification	Regression

Convolutional neural networks – CNNs

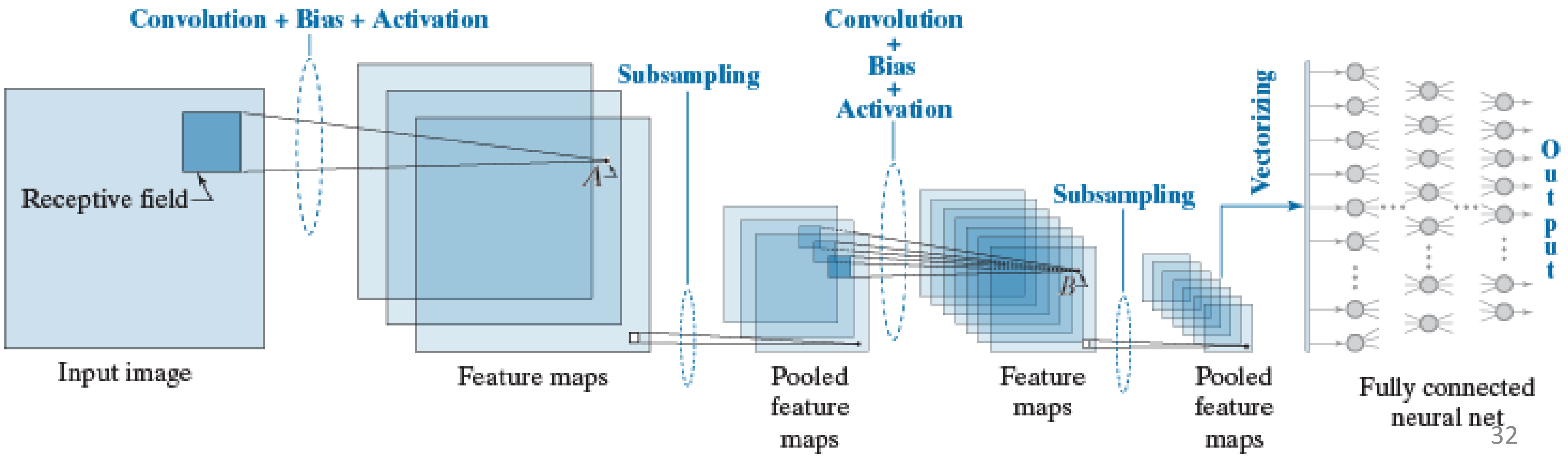
DL on images

- Focus on the input data structure
- An image is a 2D matrix
  - Spatial neighborhood is important
- Interesting features are local, shift- and deformation-invariant
  - Should be taken into account in the formulation of f



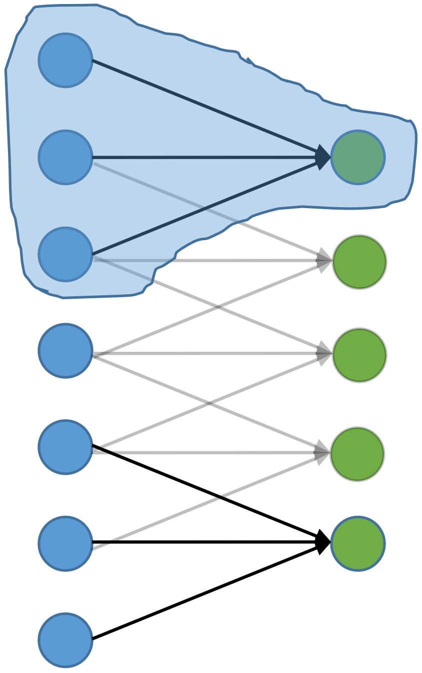
From NNs to CNNs

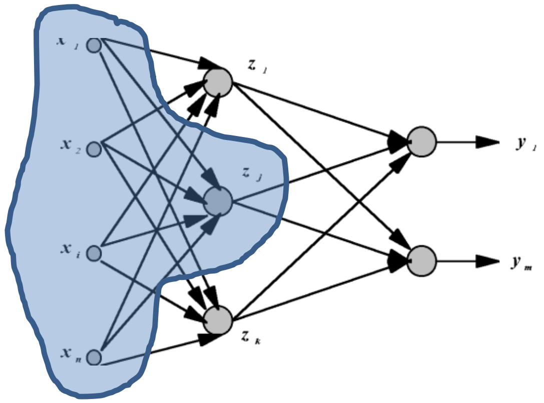
- Local connectivity: receptive field for each pixel
- Shared weights: spatially invariant response
- Multiple feature maps
- Subsampling (pooling)



Local connectivity

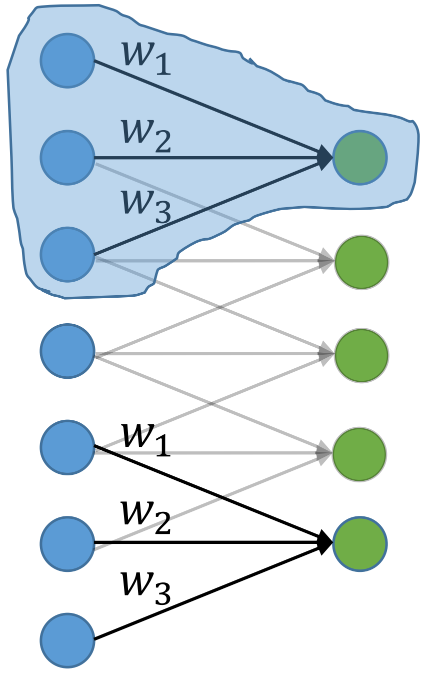
- Local connectivity
  - Only neighboring nodes are connected
- Lower number of weights to train





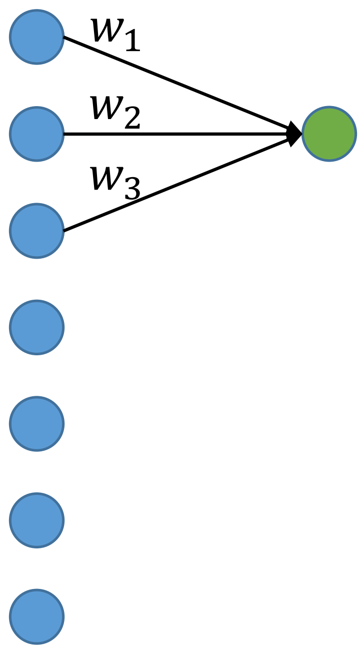
Shared (tied) weights

- All green units share the same parameters (tied weights), but operate on a different input window



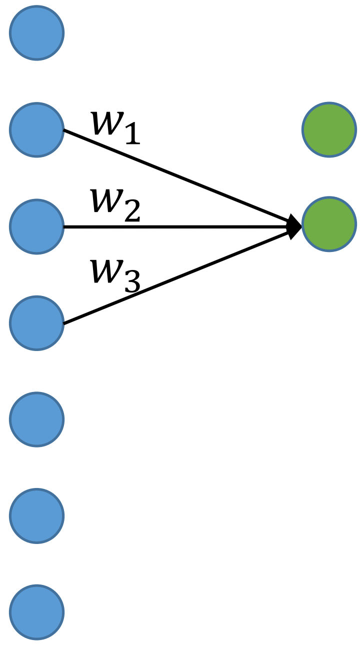
Convolution

- Convolution with 1-D filter $[w_3, w_2, w_1]$



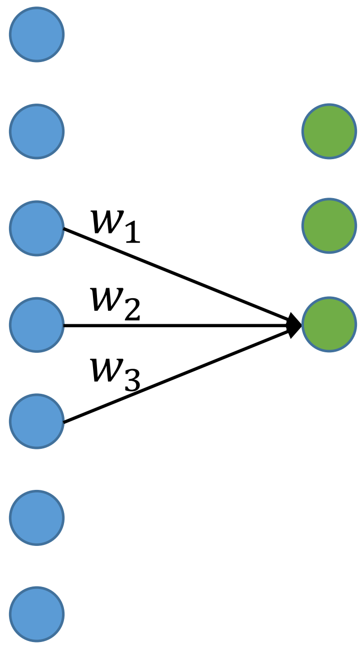
Convolution

- Convolution with 1-D filter $[w_3, w_2, w_1]$



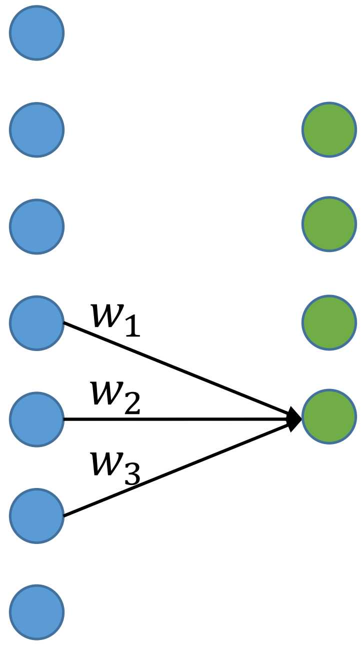
Convolution

- Convolution with 1-D filter $[w_3, w_2, w_1]$



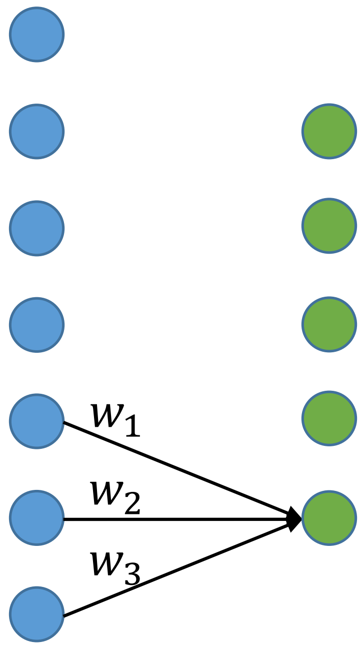
Convolution

- Convolution with 1-D filter $[w_3, w_2, w_1]$



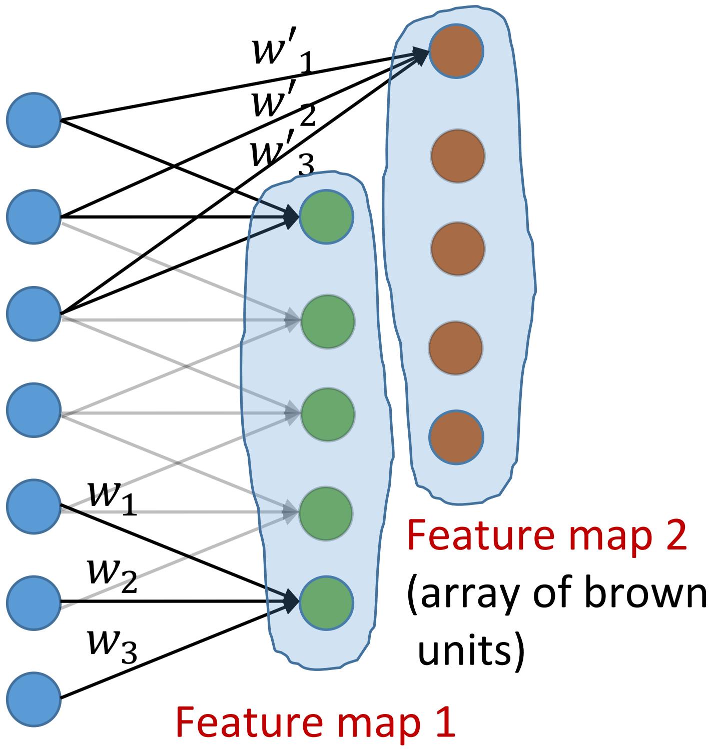
Convolution

- Convolution with 1-D filter $[w_3, w_2, w_1]$



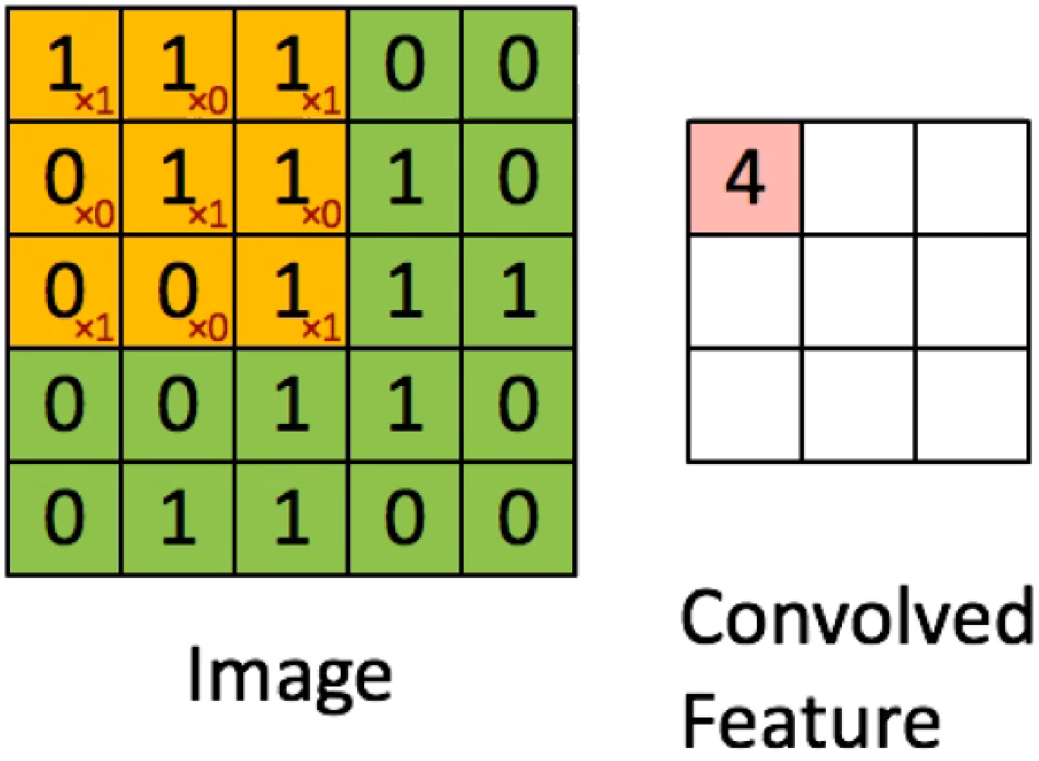
Feature maps

- Multiple feature maps are learned
- Brown and green units compute different functions
  - Both operate on the same input data windows



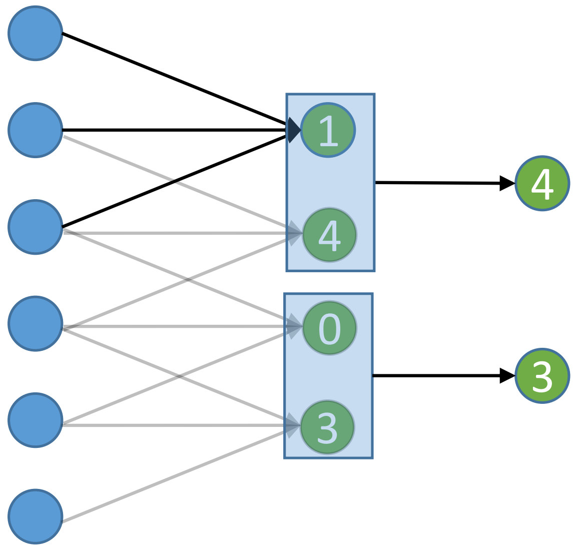
Feature map 1
(array of green units)

Convolutional layer



Pooling

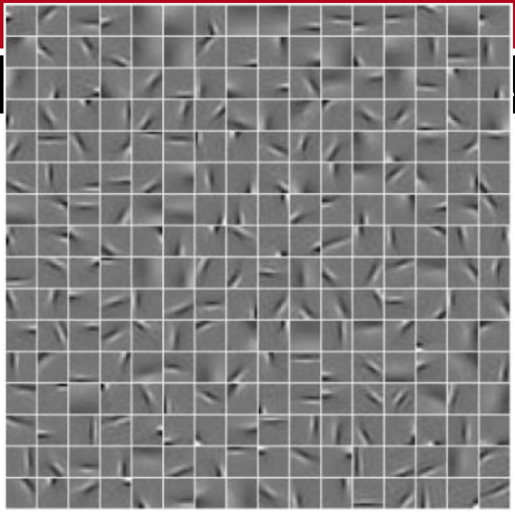
- Max and average pooling
- Feature map subsampling
  - Pooling area: 2 units
  - Pooling stride: 2 units

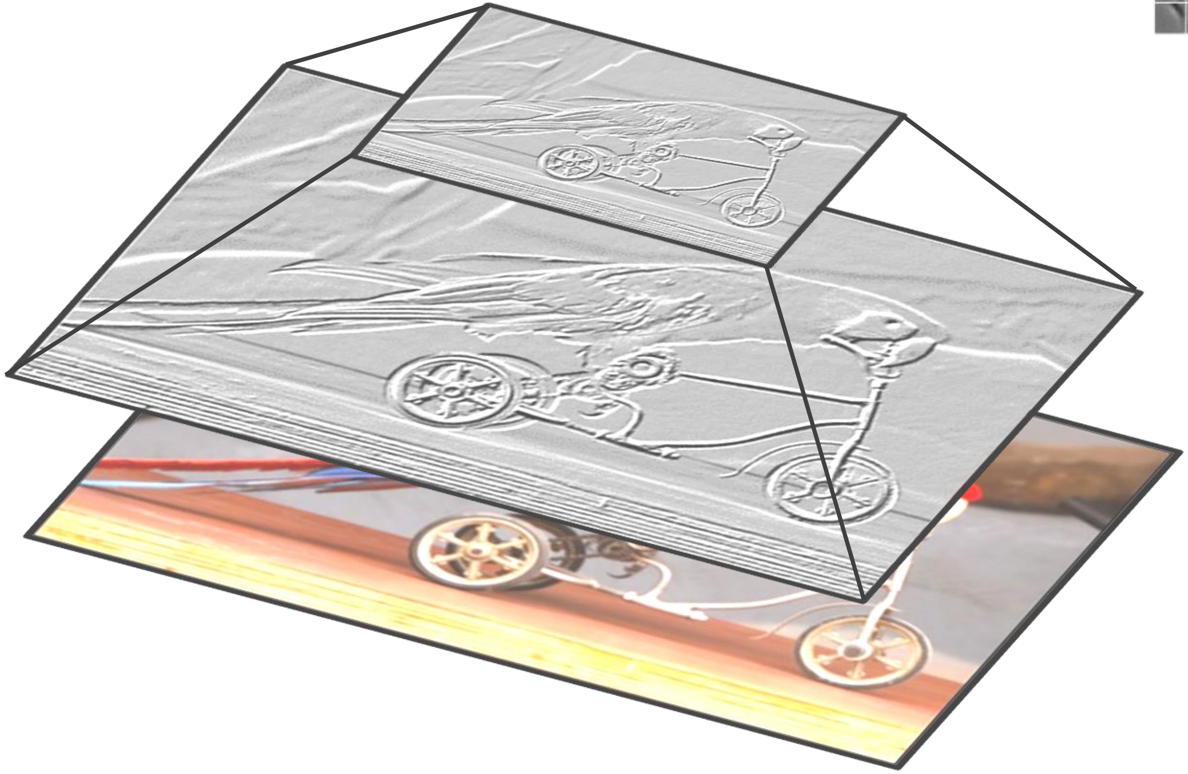


Pooling layer

- Reduce the resolution
  - Next conv layer applied at a larger scale
- Originally introduced to reduce computational complexity, but it is crucial
- Also adds some deformation invariance
- Max pooling usually employed
  - Fast and efficient, strong results

Convolution + pooling





The MNIST dataset

- MNIST database (Modified National Institute of Standards and Technology database)
- Database of handwritten digits
- 60000 training samples and 10000 testing samples (28x28 grayscale images)
- Used in many CNN examples and tutorials !

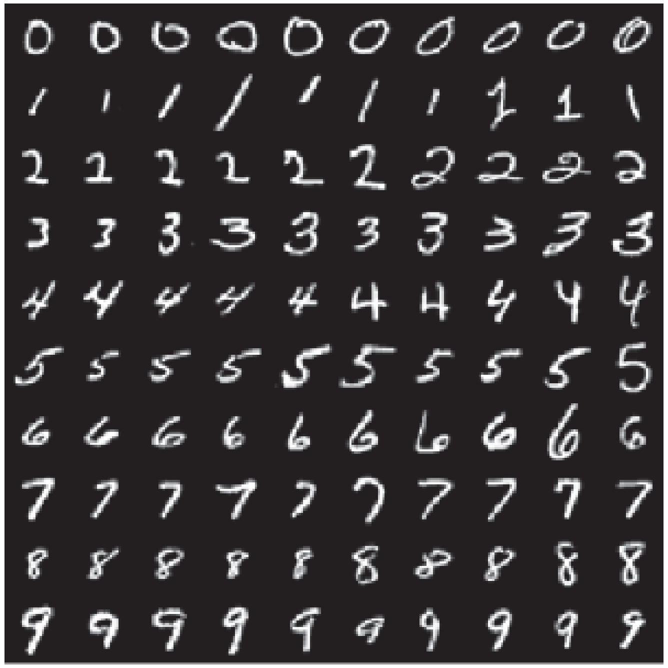
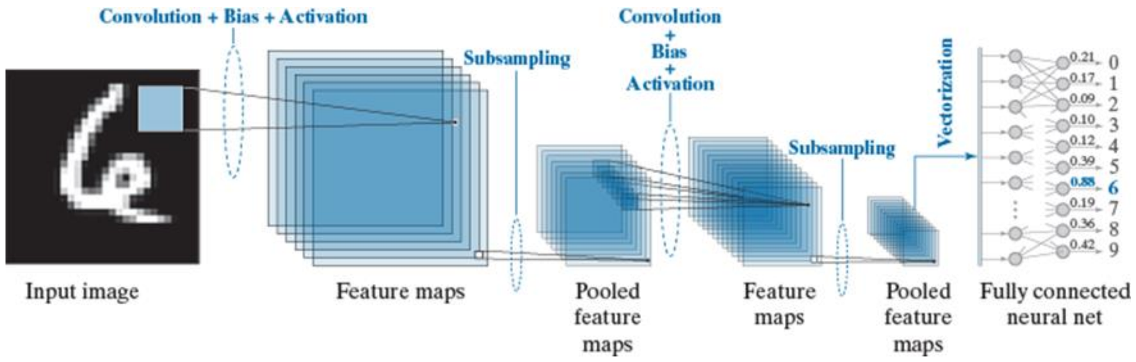


FIGURE 12.48 Samples similar to those available in the NIST and MNIST databases. Each character subimage is of size 28 × 28 pixels.(Individual images courtesy of NIST.)

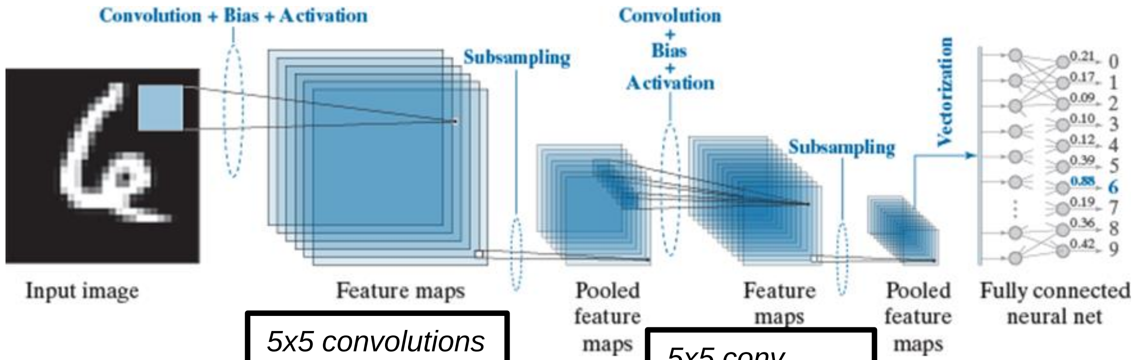
Type	Classifier	Distortion	Preprocessing	Error rate (%)
Linear classifier	Pairwise linear classifier	None	Deskewing	7.6[9]
K-Nearest Neighbors	K-NN with non-linear deformation (P2DHMDM)	None	Shiftable edges	0.52[19]
Boosted Stumps	Product of stumps on Haar features	None	Haar features	0.87[20]
Non-linear classifier	40 PCA + quadratic classifier	None	None	3.3[9]
Support vector machine	Virtual SVM, deg-9 poly, 2-pixel jittered	None	Deskewing	0.56[21]
Neural network	2-layer 784-800-10	None	None	1.6[22]
Neural network	2-layer 784-800-10	elastic distortions	None	0.7[22]
Deep neural network	6-layer 784-2500-2000-1500-1000-500-10	elastic distortions	None	0.35[23]
Convolutional neural network	6-layer 784-40-80-500-1000-2000-10	None	Expansion of the training data	0.31[15]
Convolutional neural network	6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.27[16]
Convolutional neural network	Committee of 35 CNNs, 1-20-P-40-P-150-10	elastic distortions	Width normalizations	0.23[8]
Convolutional neural network	Committee of 5 CNNs, 6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.21[17]
Random Multimodel Deep Learning (RMDL)[24]	30 Random Deep Leaning (RDL) models (10 CNNs, 10 RNNs, and 10 DNN)	None	None	0.18[18]

CNN working on MNIST

- Consider a CNN to solve the MNIST problem
  - Need to choose a network structure
  - The structure (or topology) determines the number of trainable parameters



CNN working on MNIST



5x5 convolutions
6 feature maps
2x2 pooling

*(25+1)*6=156 params*

Bias

5x5 conv.
12 feat. maps
2x2 pooling

*(6*25+1)*12=1812 params*

Receptive field size

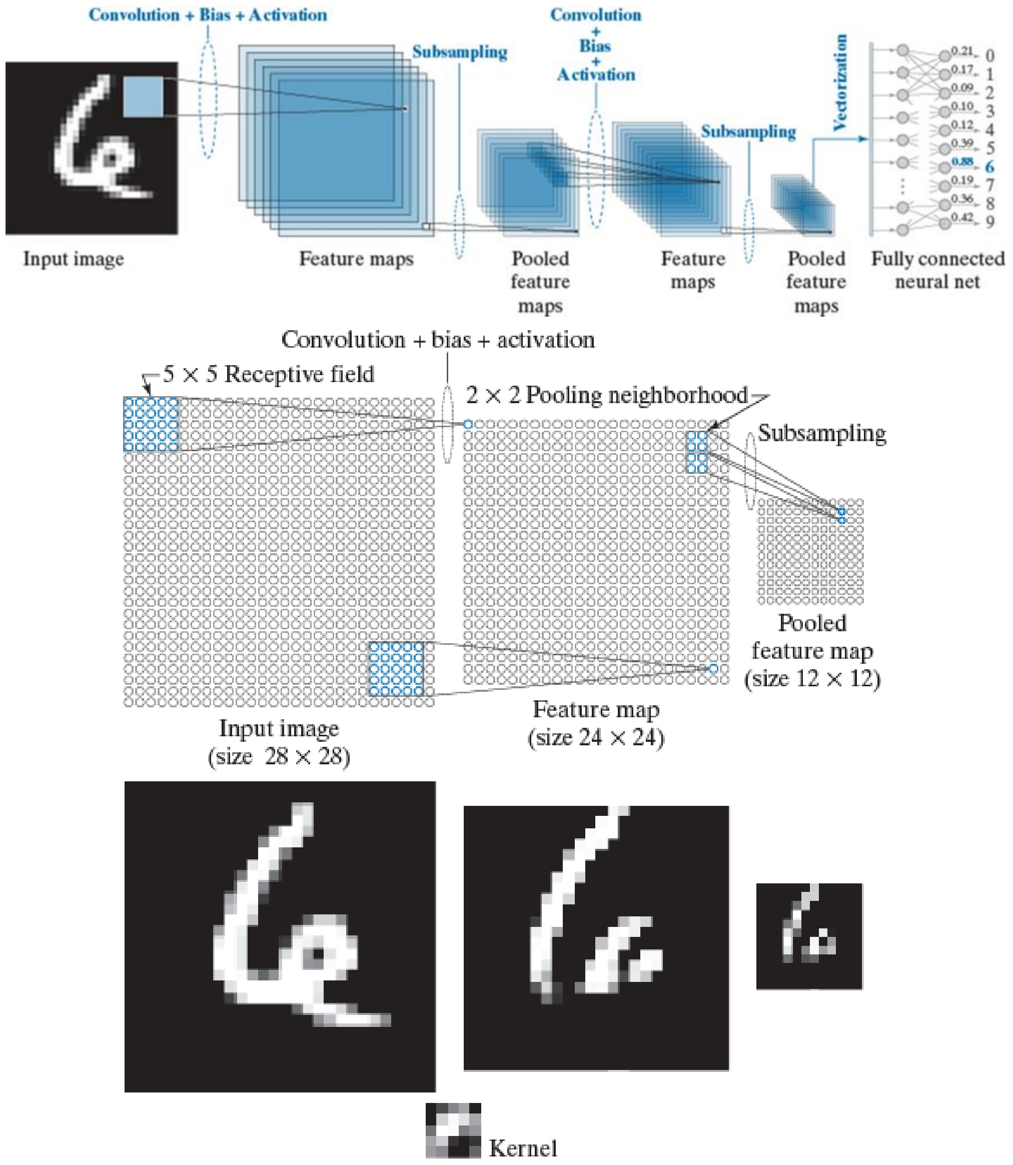
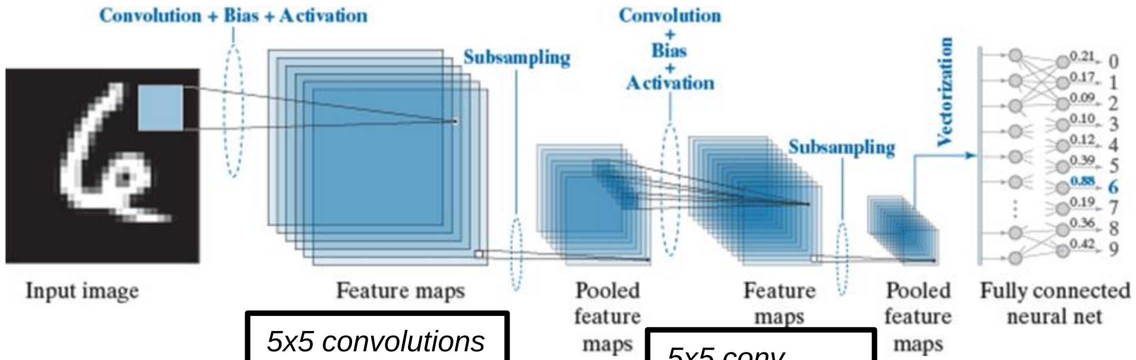

Convolution + Bias + Activation
Subsampling
Convolution + Bias + Activation
Subsampling
Vectorization
Input image
Feature maps
Pooled feature maps
Feature maps
Pooled feature maps
Fully connected neural net
0.21 0
0.17 1
0.09 2
0.10 3
0.12 4
0.39 5
0.88 6
0.19 7
0.36 8
0.42 9
Convolution + bias + activation
5 × 5 Receptive field
2 × 2 Pooling neighborhood
Subsampling
Pooled feature map (size 12 × 12)
Input image (size 28 × 28)
Feature map (size 24 × 24)
Kernel

FIGURE 12.41
Top row: How the sizes of receptive fields and pooling neighborhoods affect the sizes of feature maps and pooled feature maps.
Bottom row: An image example. This figure is explained in more detail in Example 12.17. (Image courtesy of NIST.)

Trained CNN layers



5x5 convolutions
6 feature maps
2x2 pooling

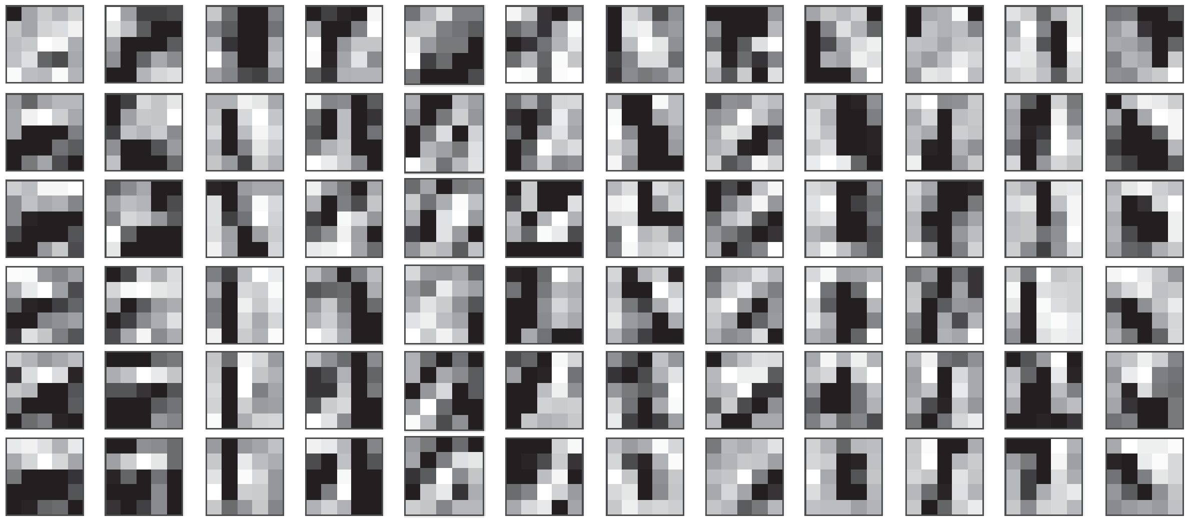
*(25+1)*6=156 params*

5x5 conv.
12 feat. maps
2x2 pooling

*(6*25+1)*12=1812 params*

Layer 1

Layer 2



Computing feature maps

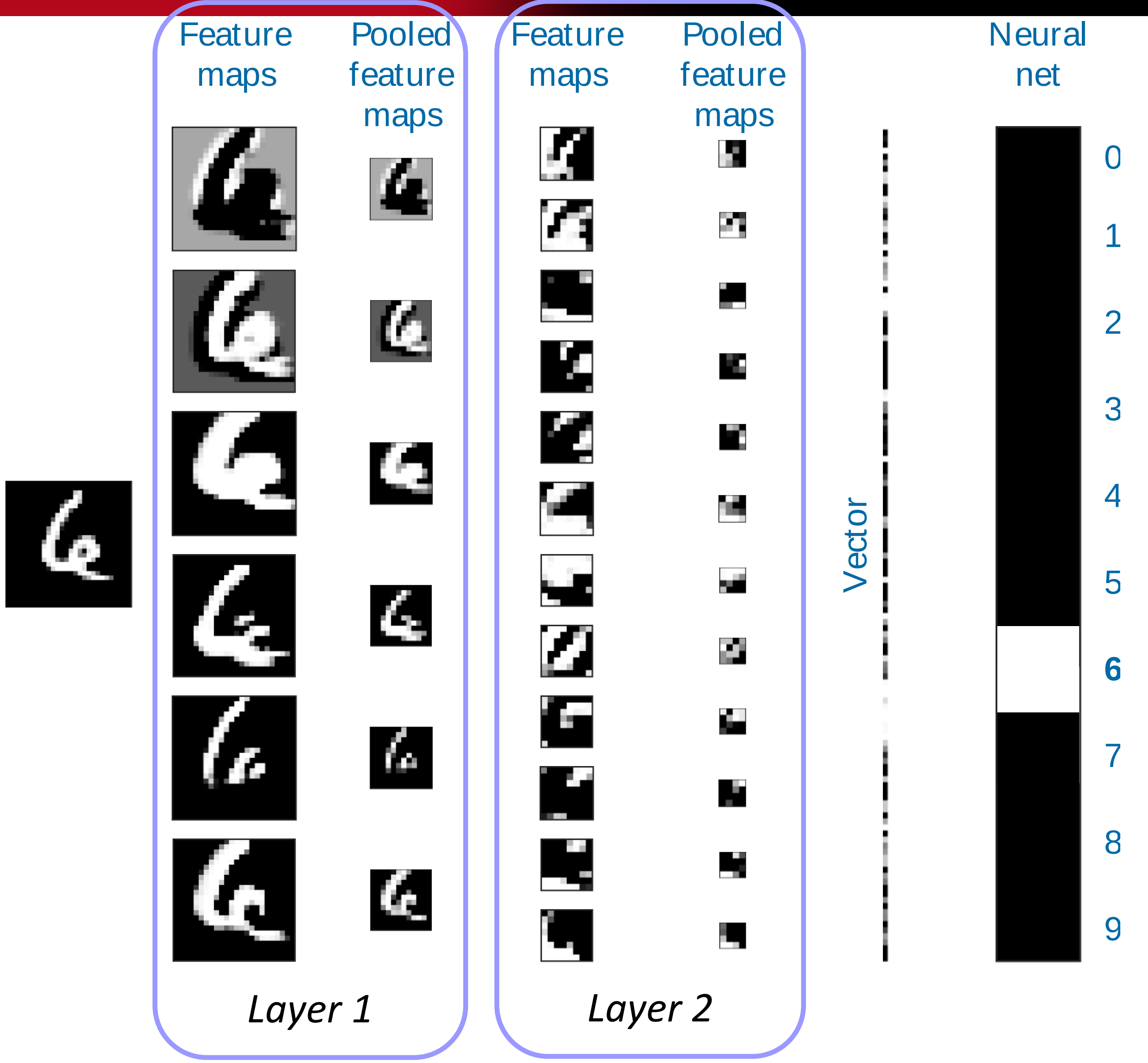


FIGURE 12.44 Visual summary of an input image propagating through the CNN in Fig. 12.42. Shown as images are all the results of convolution (feature maps) and pooling (pooled feature maps) for both layers of the network. (Example 12.17 contains more details about this figure.)

Performance on MNIST recognition

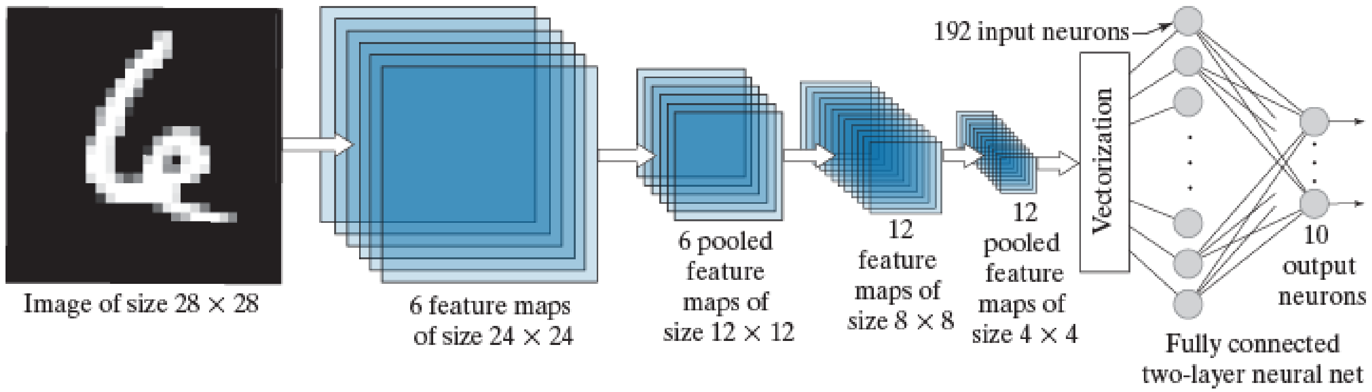


Image of size 28 × 28
6 feature maps of size 24 × 24
6 pooled feature maps of size 12 × 12
12 feature maps of size 8 × 8
12 pooled feature maps of size 4 × 4
Vectorization
192 input neurons
10 output neurons
Fully connected two-layer neural net

FIGURE 12.49 CNN used to recognize the ten digits in the MNIST database. The system was trained with 60,000 numerical character images of the same size as the image shown on the left. This architecture is the same as the architecture we used in Fig. 12.42. (Image courtesy of NIST.)

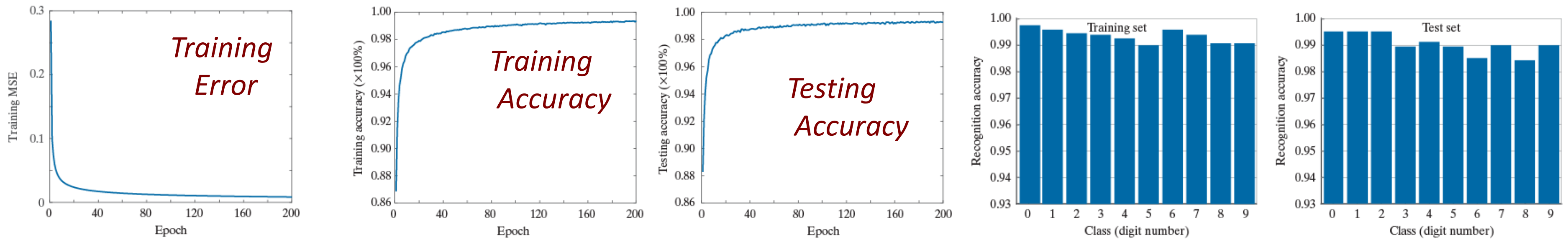
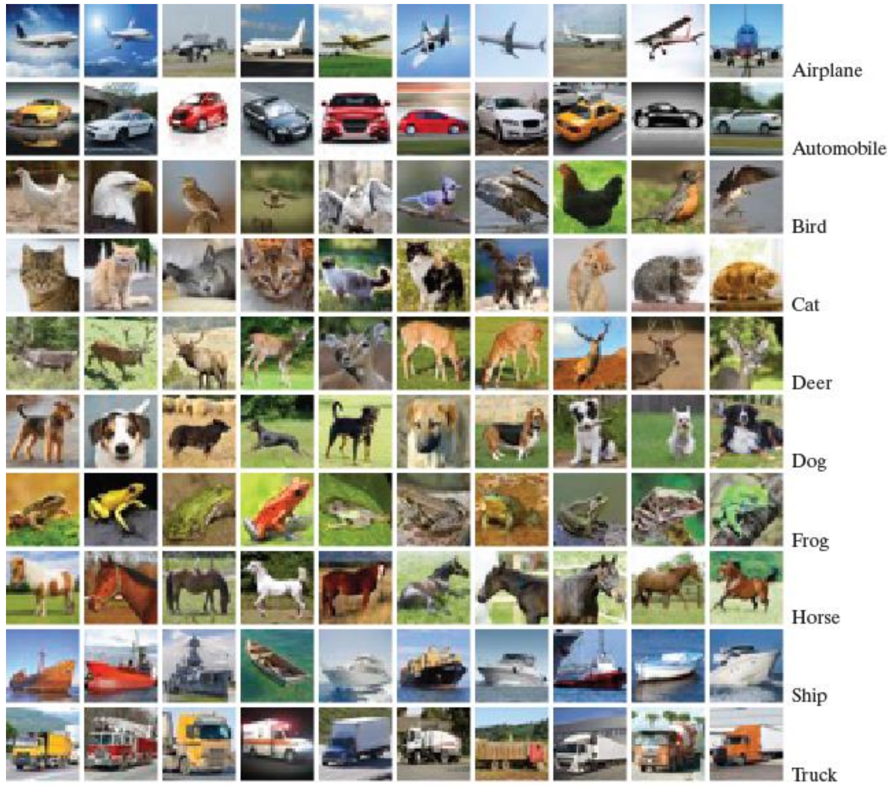
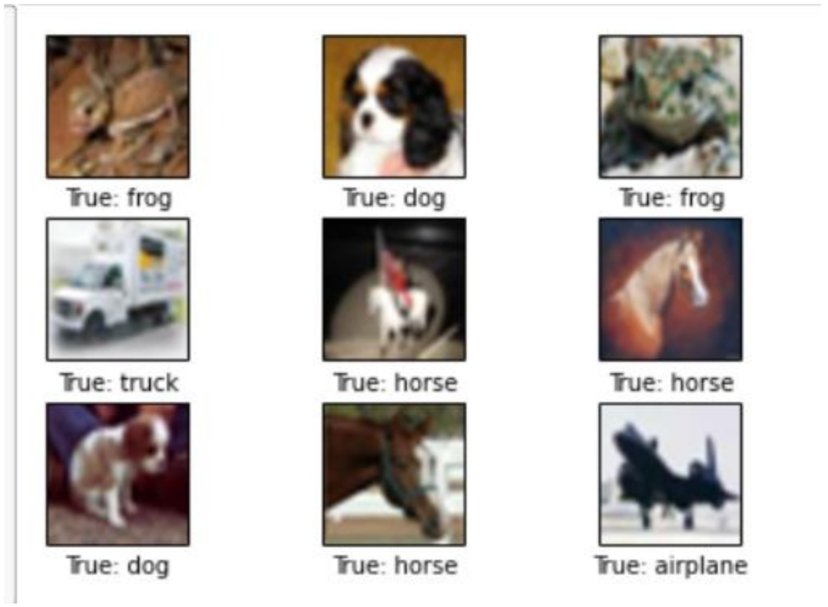


FIGURE 12.51 (a) Training accuracy (percent correct recognition of the training set) as a function of epoch for the 60,000 training images in the MNIST database. The maximum achieved was 99.36% correct recognition. (b) Accuracy as a function of epoch for the 10,000 test images in the MNIST database. The maximum correct recognition rate was 99.13%.

CIFAR-10

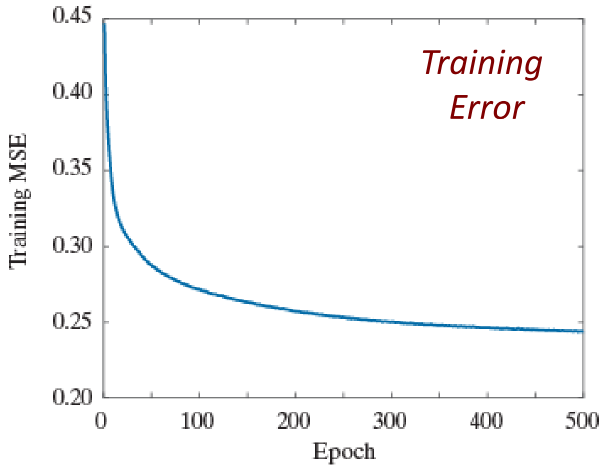
- 60k images (50k training, 10k testing), 32×32 pixels
- 10 classes, 6k images per class
- High-quality images, clearly separable classes
- Color images: 3 input channels

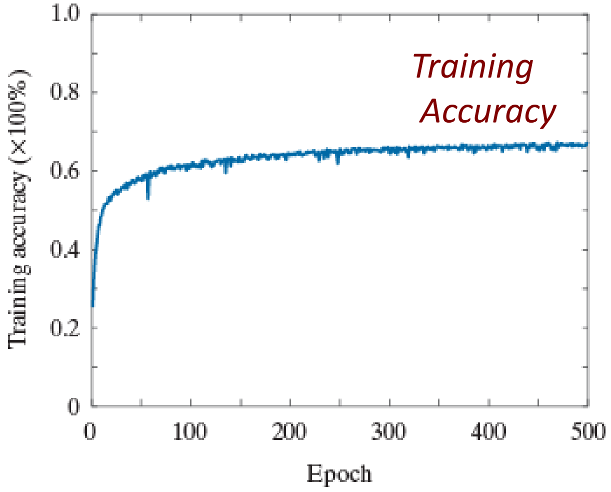


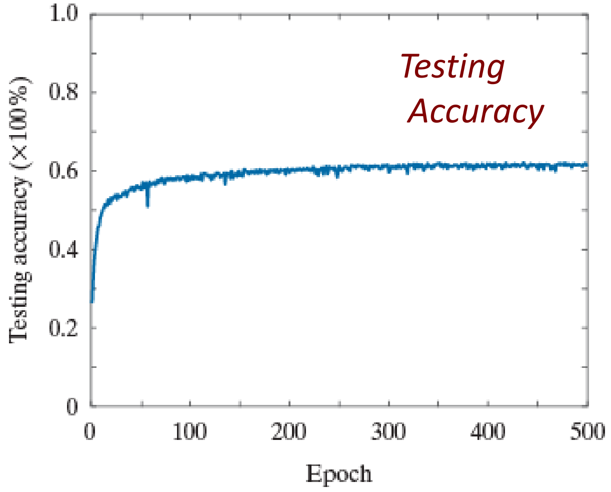


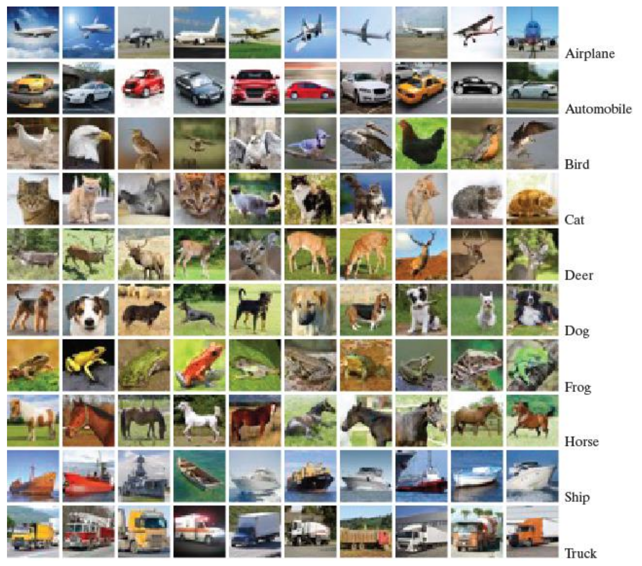
UNIVERSITÀ DEGLI STUDI DI PADOVA

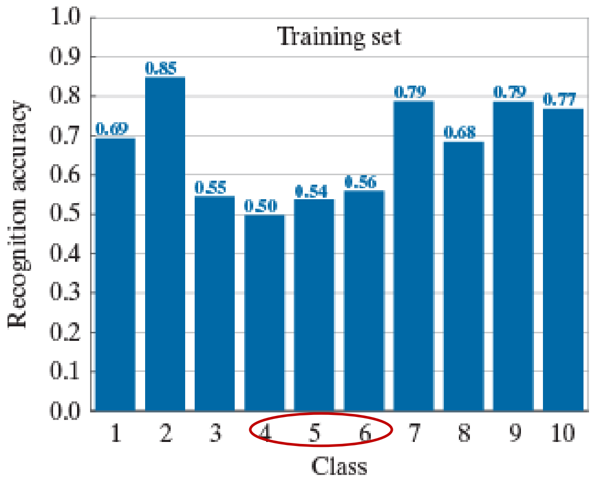
Training on CIFAR-10

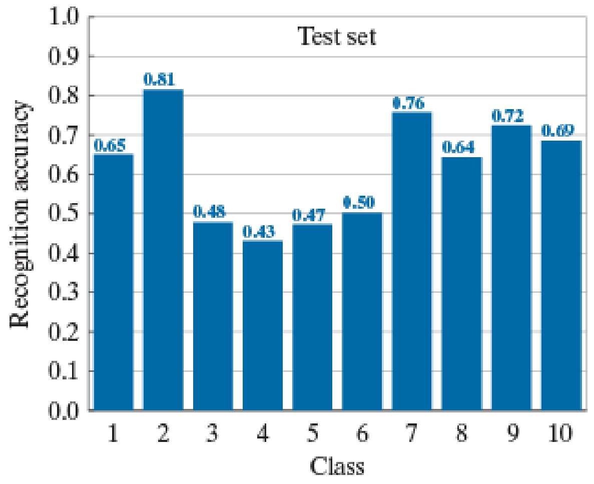












Weights

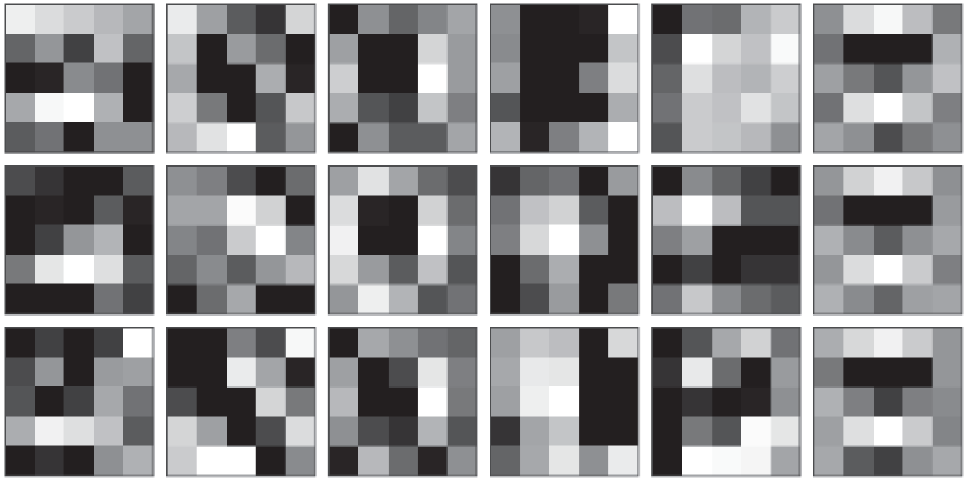


FIGURE 12.60 Weights of the kernels of the first convolution layer after 500 epochs of training.

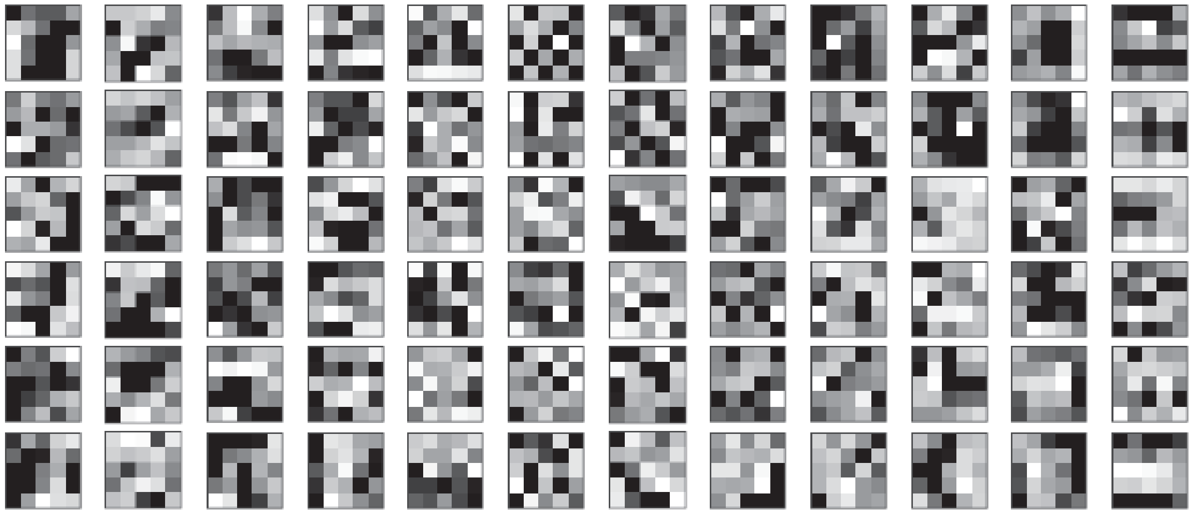


FIGURE 12.61 Weights of the kernels of the second convolution layer after 500 epochs of training. The interpretation of these kernels is the same as in Fig. 12.54.

Forward pass

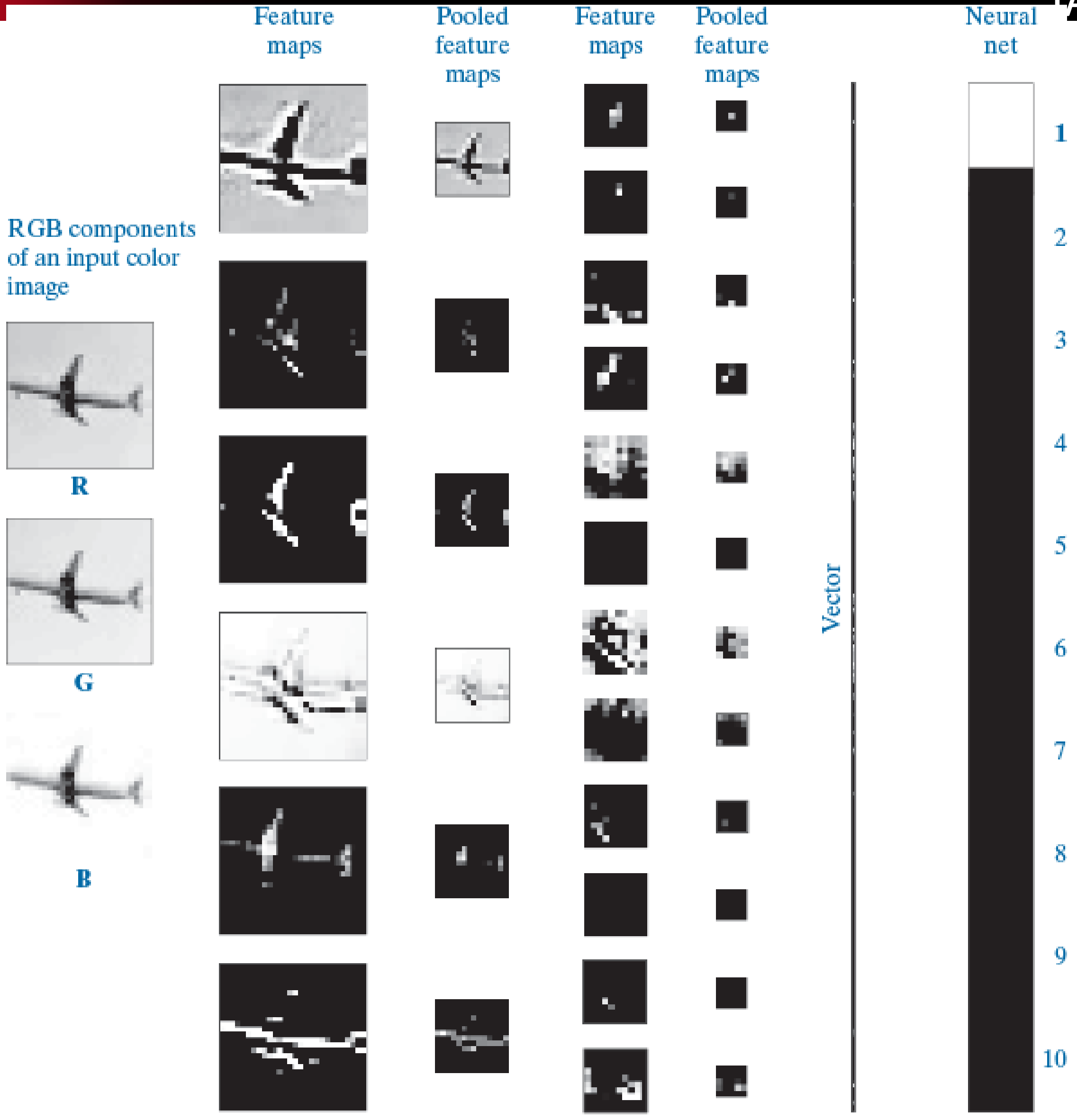
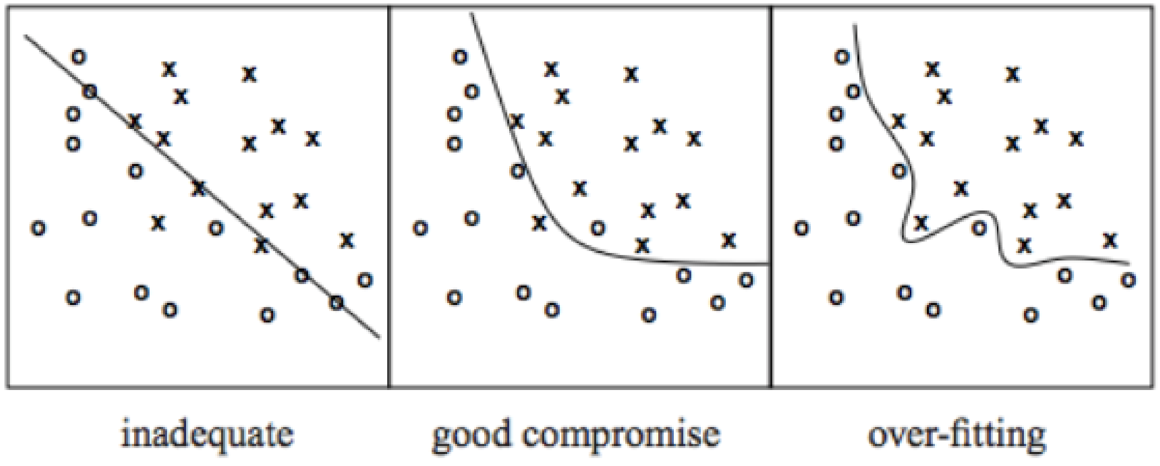
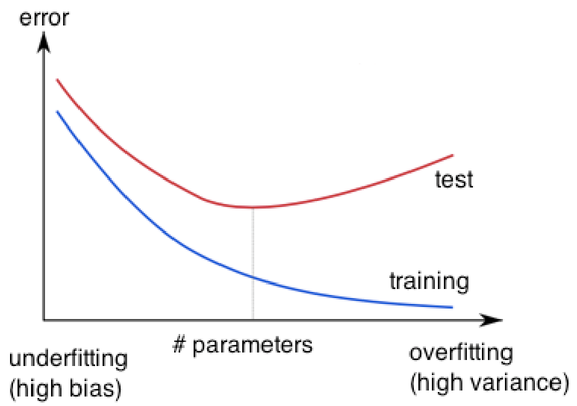


FIGURE 12.62 Graphical illustration of a forward pass through the trained CNN. The purpose was to recognize one input image from the set in Fig. 12.56. As the output shows, the image was recognized correctly as belonging to class 1, the class of airplanes. (Original image courtesy of Pearson Education.)

Insights: overfitting

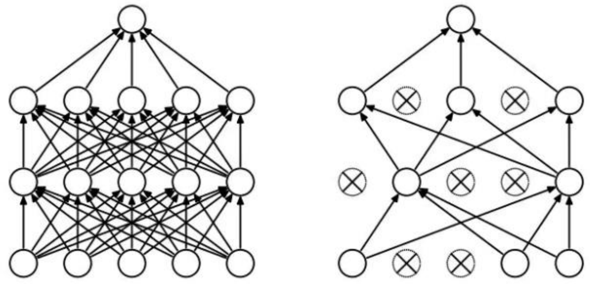
- Nearly perfect fitting on the training data
- No generalization, leading to high errors on test data
- Typically: model too complex WRT training data





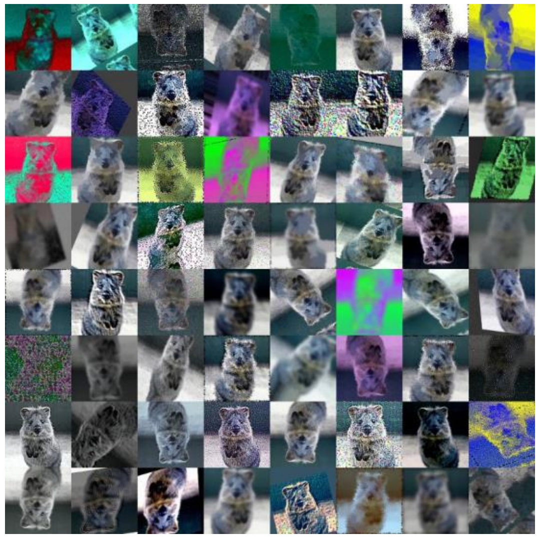
Insights: regularization

- Dropout
  - Randomly drop units during training
  - Each unit retained with fixed probability p



- Early stopping
  - Use validation error to decide when training should be stopped
  - Stop when monitored quantity is not improved after n subsequent epochs (n is called patience)

Insights: pre-processing



- Add some variance to the data
  - Noise
  - Rotation
  - Flip
  - Hue, color, saturation
- This is often called data augmentation

UNIVERSITÀ DEGLI STUDI DI PADOVA

Introduction to Deep Learning

Stefano Ghidoni

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

INTELLIGENT AUTONOMOUS SYSTEMS LAB