CS286: AI for Science and Engineering

Lecture 4: Convolutional Neural Network (CNN) and Autoencoder (AE)

Jie Zheng (郑杰)

PhD, Associate Professor

School of Information Science and Technology (SIST), ShanghaiTech University

Fall, 2023

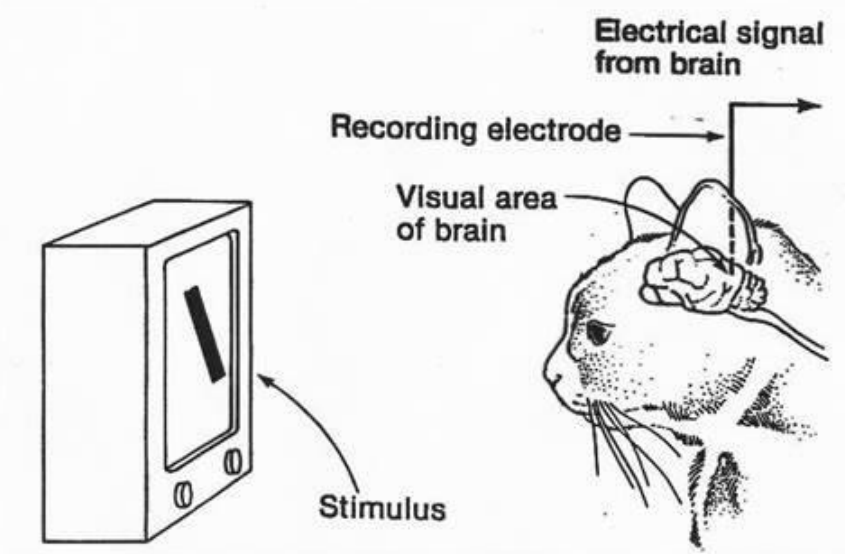
Some PPT on CNN courtesy from Prof. Xuming He, SIST, ShanghaiTech University

Learning objectives

- After taking this lecture, you should be able to:
  - Describe the biological background and rationale behind convolutional neural network (CNN)
  - Describe main components of a CNN
  - Explain how convolutional layers and pooling layers work
  - Explain the basic idea behind Autoencoders from perspective of efficient data representations
  - Describe the main ideas of a few variants of Autoencoders

Biological background: Visual Cortex

- David H. Hubel and Torsten Wiesel performed experiments on the visual system of cats in 1958 and 1959 (Nobel Prize in Physiology or Medicine, 1981)
- They gave insights into the structure of visual cortex:
  - Many neurons in the visual cortex have a small **local receptive field**, i.e. they react only to visual stimuli located in a limited region of the visual field; together they tile the whole visual field
  - Some neurons react only to lines of specific orientations (e.g. horizontal lines)
  - Some neurons have larger receptive fields and react to more complex patterns that are combinations of lower-level patterns
  - **Crucial idea**: Higher-level neurons are based on the outputs of neighboring lower-level neurons



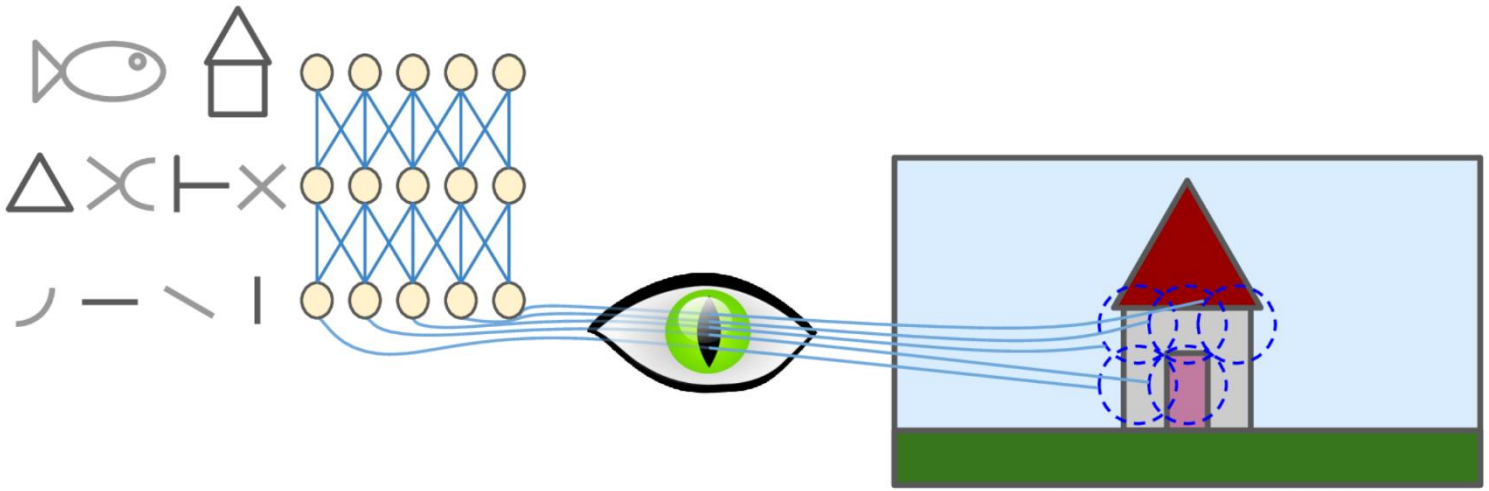
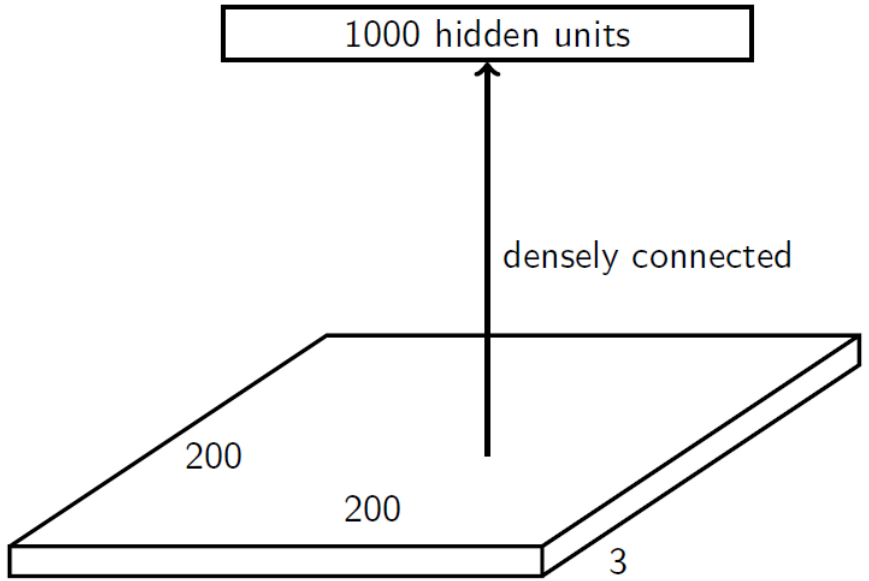


Figure from Fig. 14-1, page 447, Aurélien Géron's book "Hands-on Machine Learning" (Ed. 2), 2019

Motivation: Visual recognition

- Suppose we aim to train a network that takes a 200x200 RGB image as input
- What is the problem with having full connections in the first layer?
  - Too many parameters! 200x200x3x1000 = 120 million
  - What happens if the object in the image shifts a little?



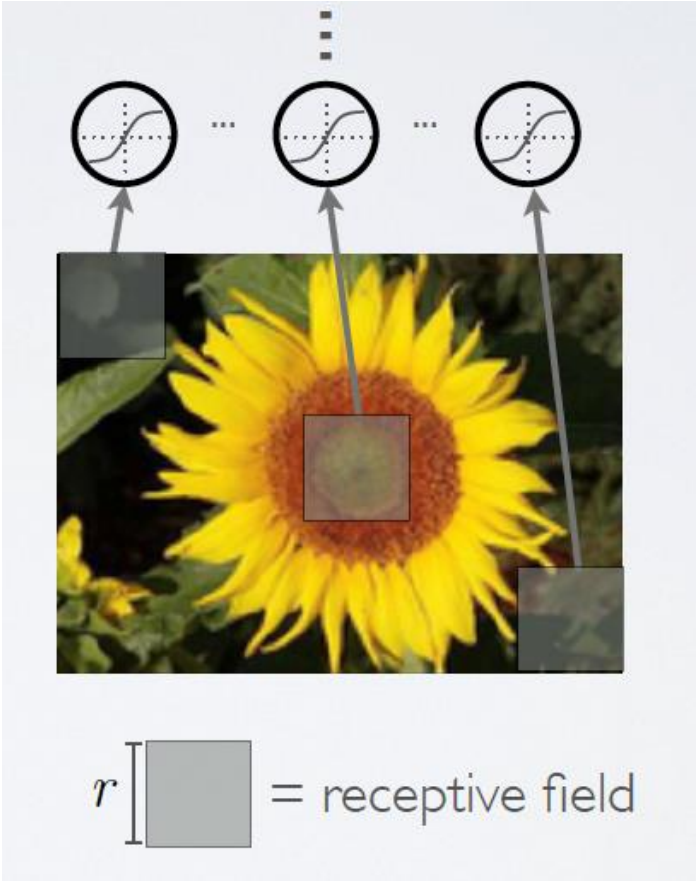
Goals

- Visual Recognition: Design a neural network that
  - Can deal with very high-dimensional inputs
  - Can exploit the 2D topology of pixels in images
  - Can build invariance to certain variations so that we can expect
    - Translation, small deformations, illumination, etc.
- Convolution networks leverage these ideas
  - Local connectivity
  - Parameter sharing
  - Pooling/subsampling hidden units

Overview of CNN

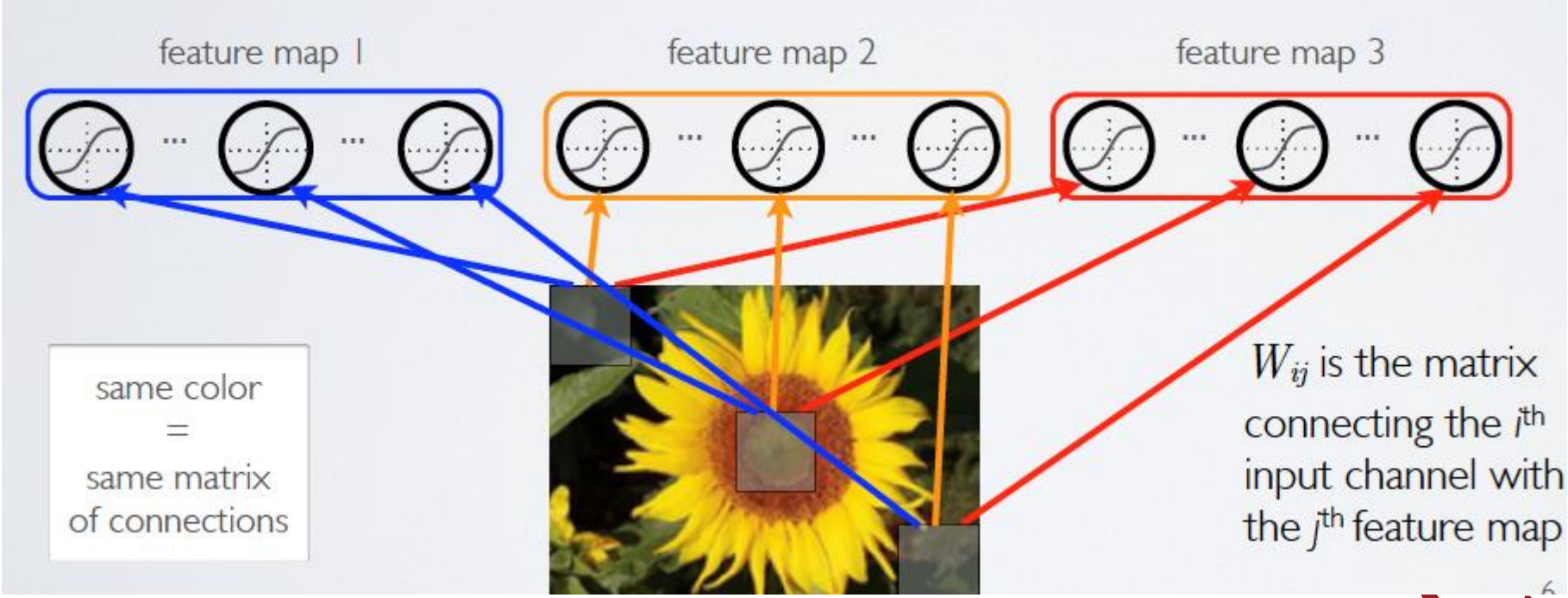
Overview of CNN

- **Idea 1**: Use a local connectivity of hidden units
  - Each hidden unit is connected only to a sub-region (patch) of the input image
  - Usually it is connected to all channels
  - Each neuron has a local receptive field



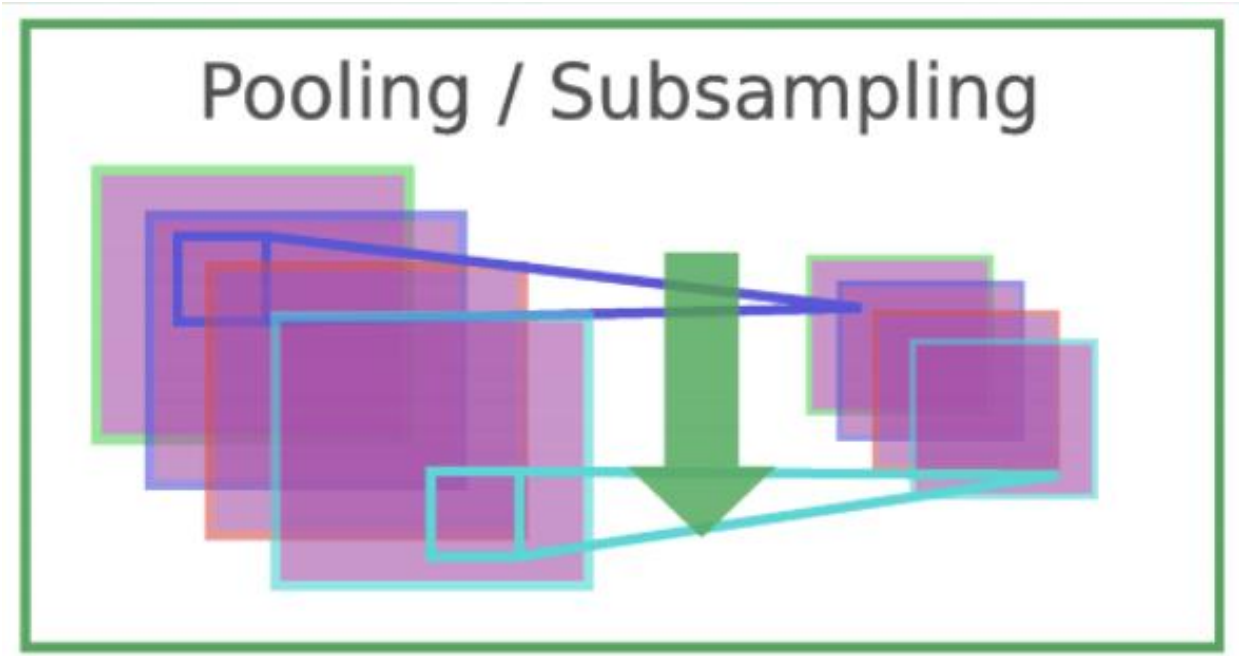
Overview of CNN

- **Idea 2**: Share weights across certain units
  - Units organized into the same "feature map" share weight parameters
  - Hidden units within a feature map cover different positions in the image



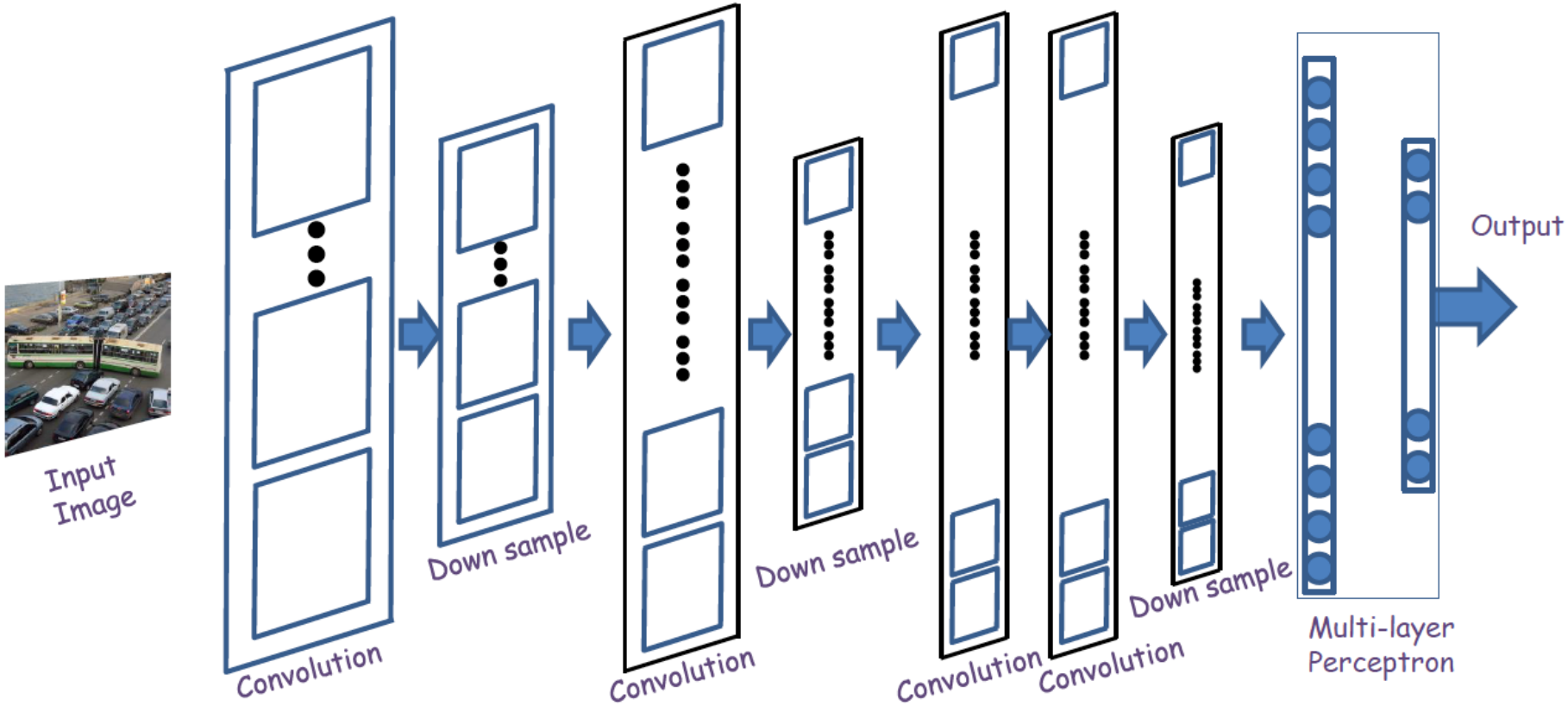
Overview of CNN

- **Idea 3**: Pool hidden units in the same neighborhood
  - Averaging or discarding location information in a small region
  - Robust toward small deformations in object shapes by ignoring details



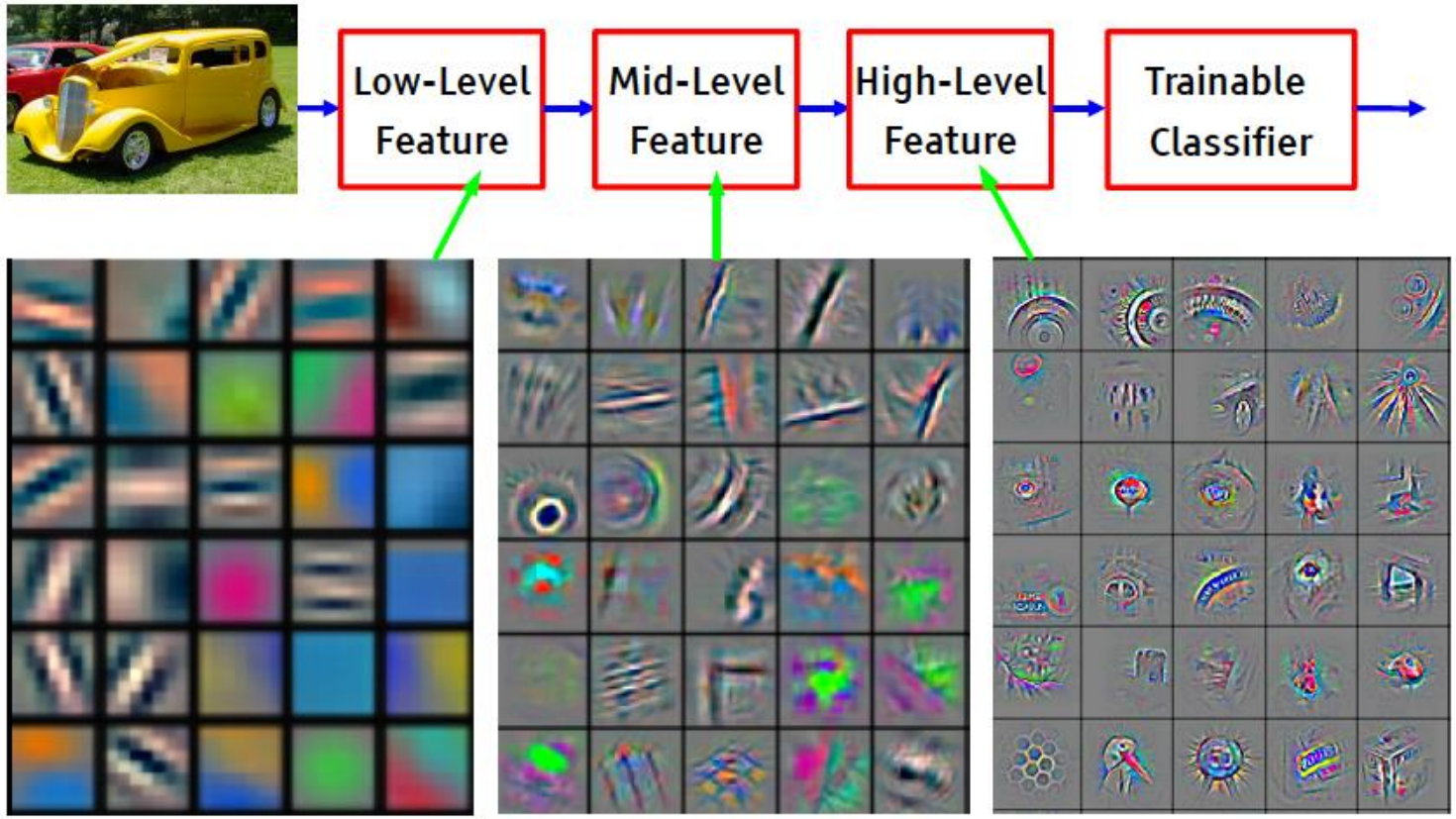
Overview of CNN

- **Idea 4**: Interleaving feature extraction and pooling operations
  - Extracting abstract, compositional features for representing semantic object classes



Overview of CNN

- Artificial visual pathway: from images to semantic concepts
  - Representation learning



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Convolutional Layers

2D Convolution

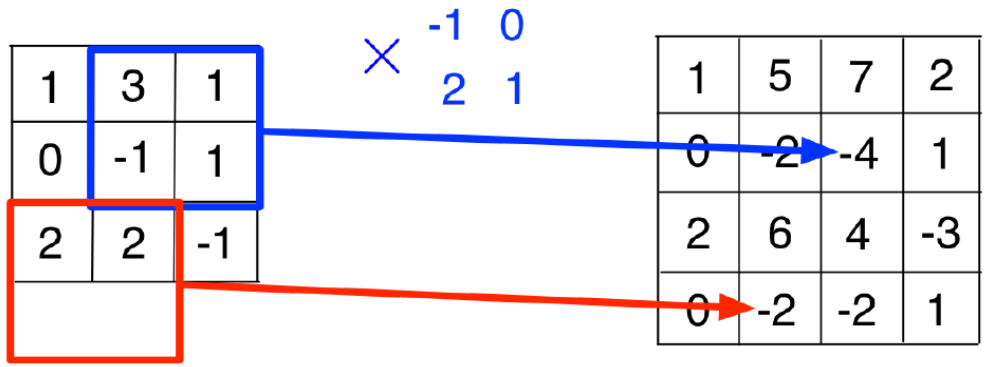
If A and B are two 2-D arrays, then:

$$(A * B)_{ij} = \sum_s \sum_t A_{st} B_{i-s,j-t}.$$

1	3	1
0	-1	1
2	2	-1

$*$

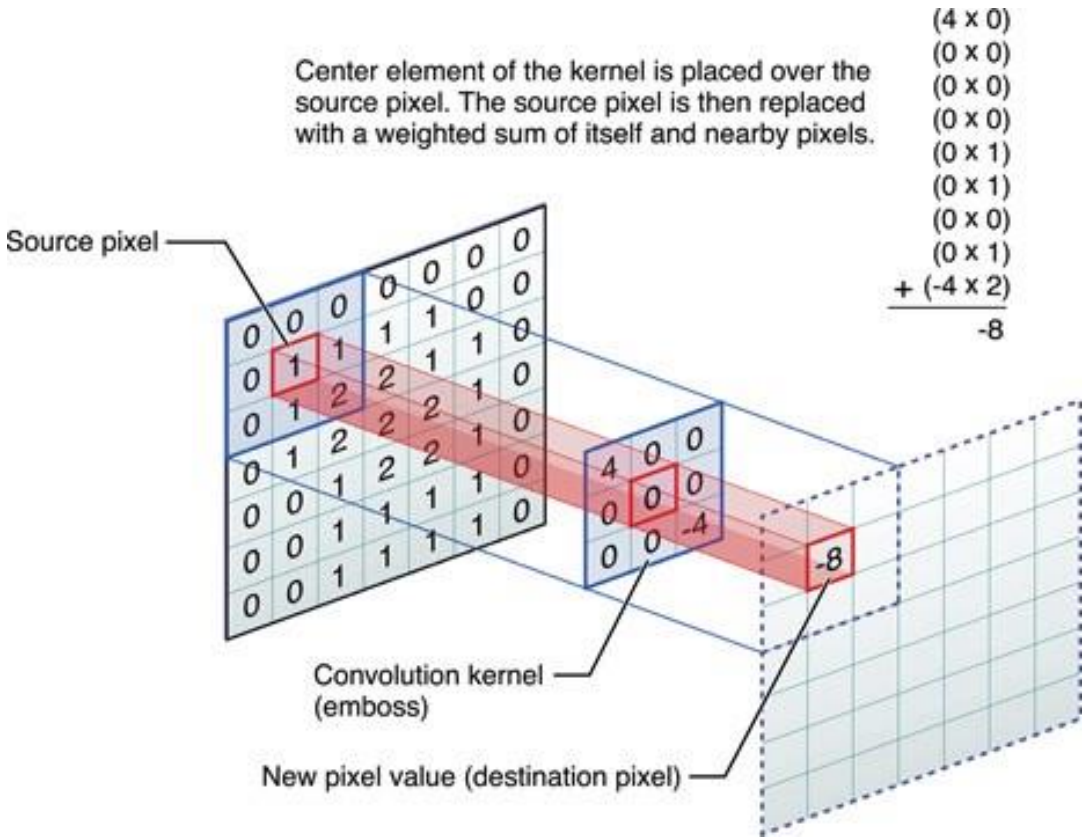
1	2
0	-1

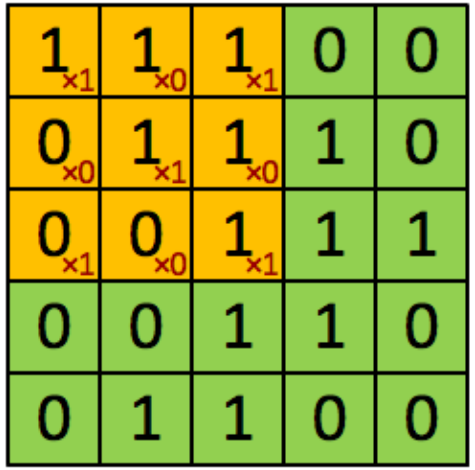


2D Convolution

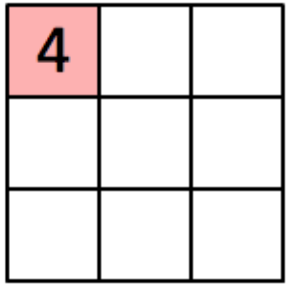
If A and B are two 2-D arrays, then:

$$(A * B)_{ij} = \sum_s \sum_t A_{st} B_{i-s,j-t}.$$





Image

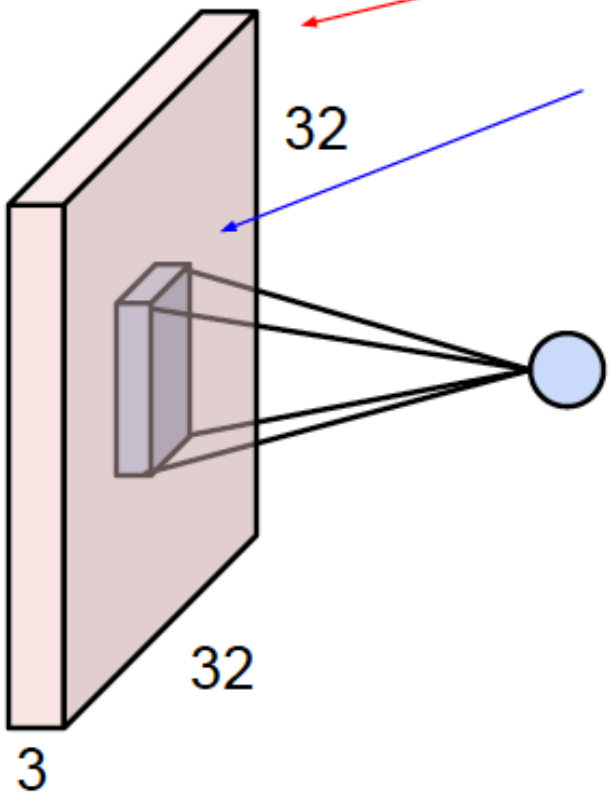


Convolved Feature

Picture Courtesy: developer.apple.com

Convolution layers

- Formal definition

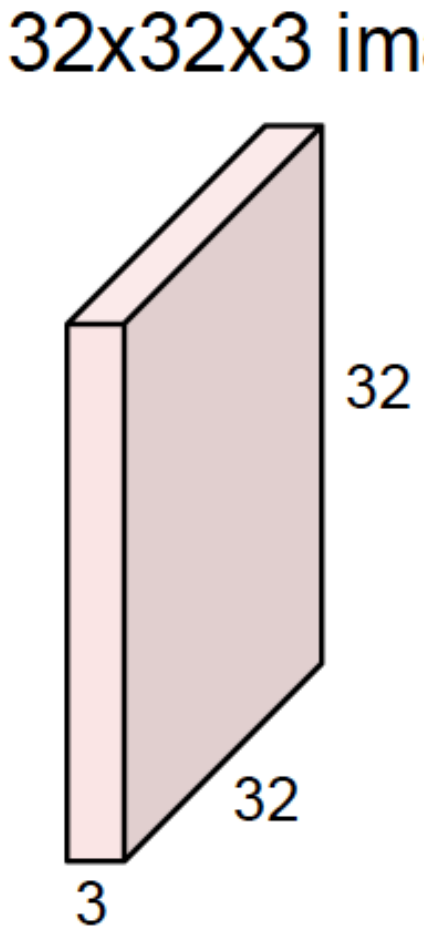


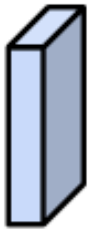
1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

Convolution layers

- Define a neuron corresponding to a 5x5x3 filter

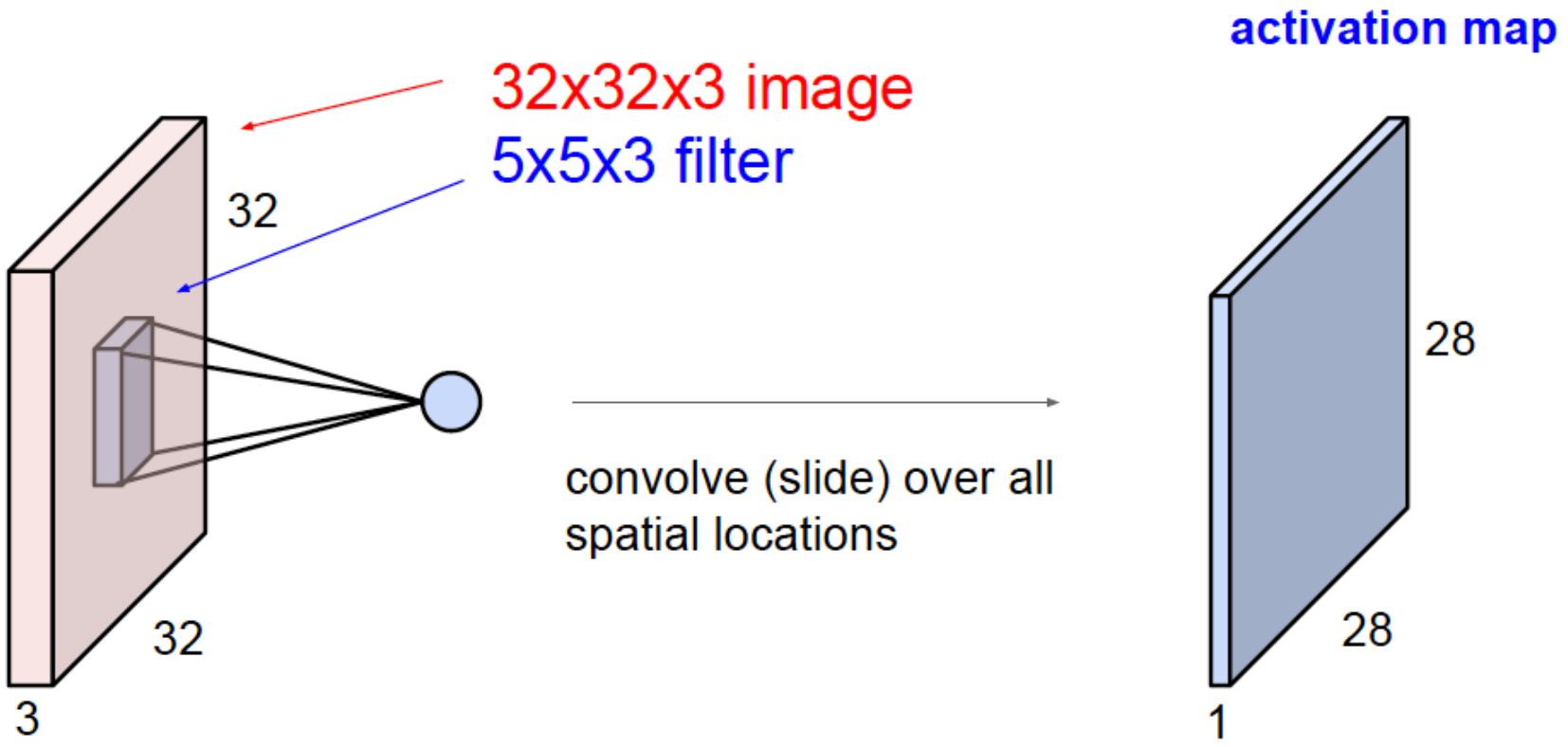




Convolve the filter with the image
i.e. “slide over the image spatially, computing dot products”

Convolution layers

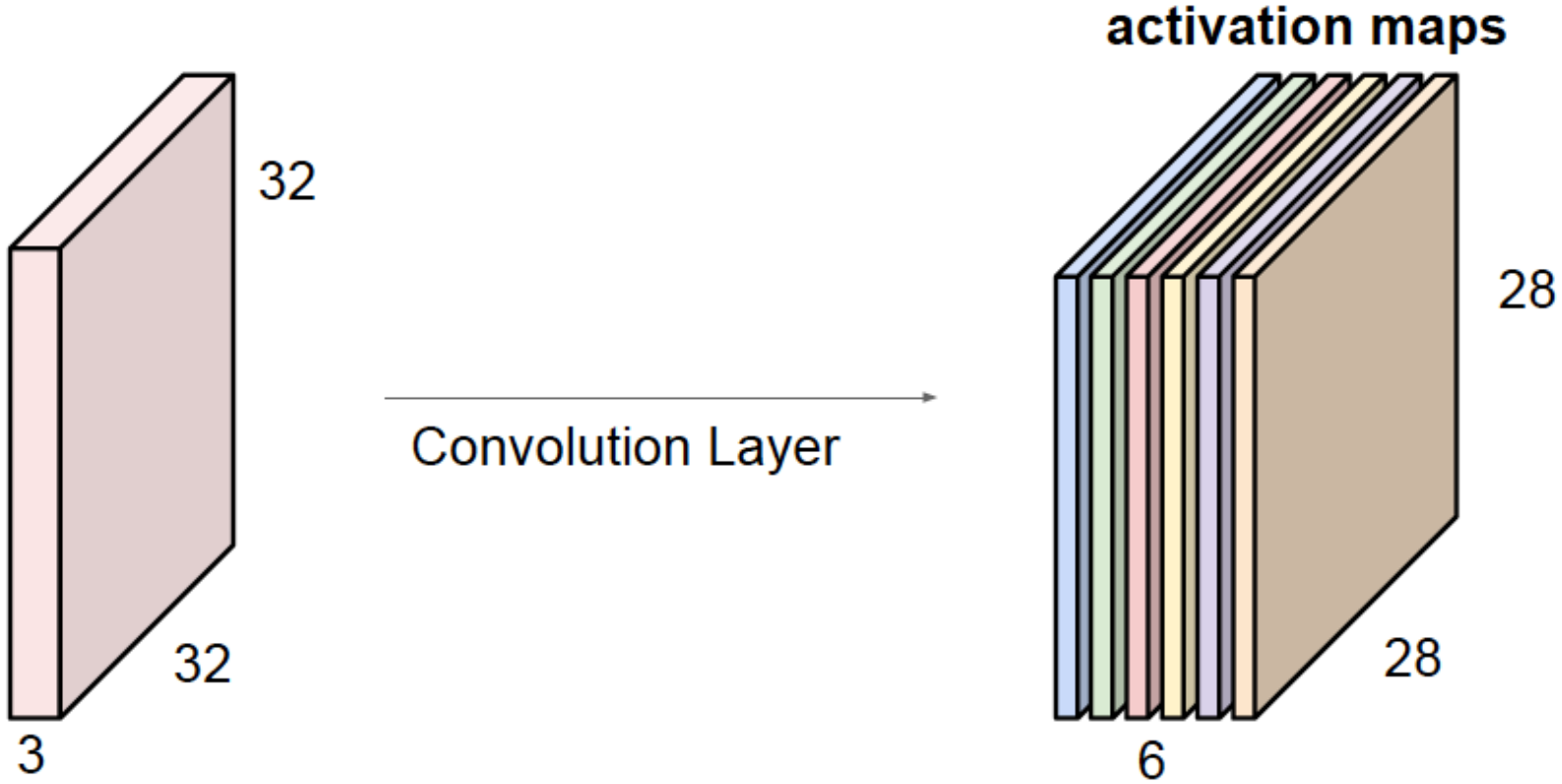
- Convolution operation
  - Parameter sharing
  - Spatial information



Convolution layers

- Multiple kernels/filters

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:

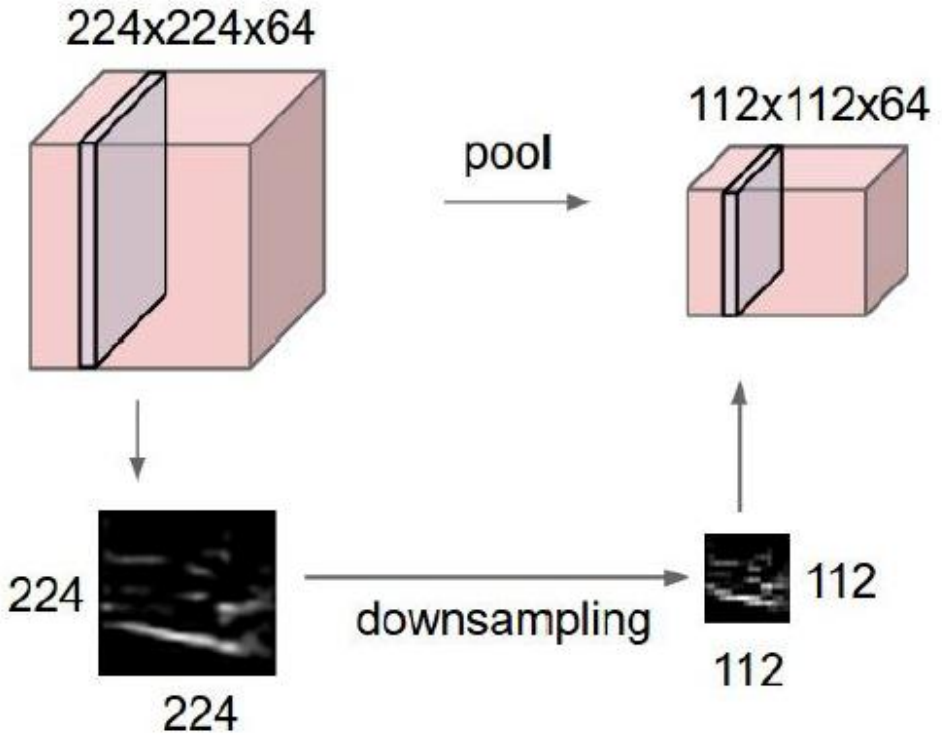


We stack these up to get a "new image" of size 28x28x6!

Pooling Layers

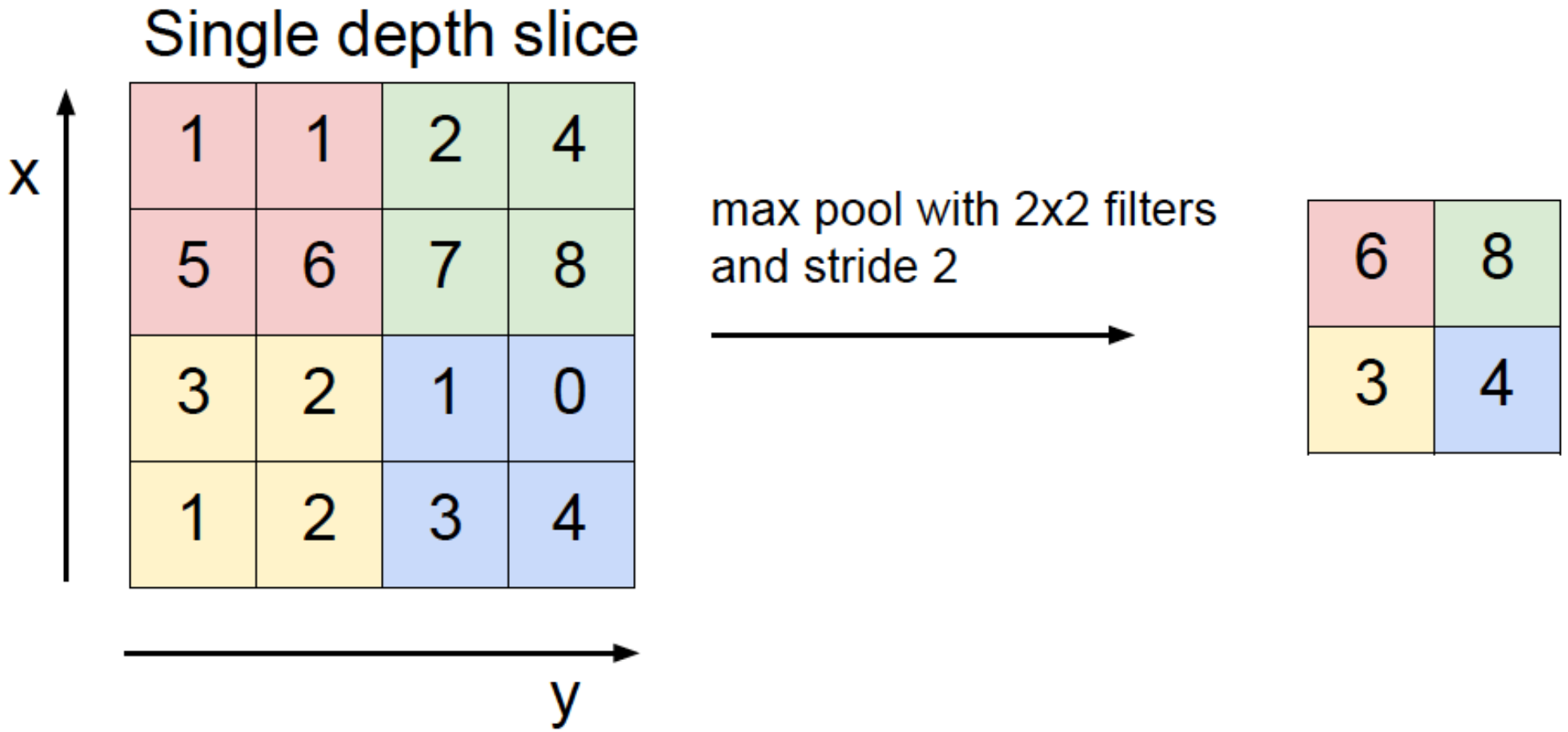
Pooling layers

- Reducing the spatial sizes of the feature maps
  - Smaller representations
  - On each activation map independently
  - Low resolution means fewer details



Pooling layers

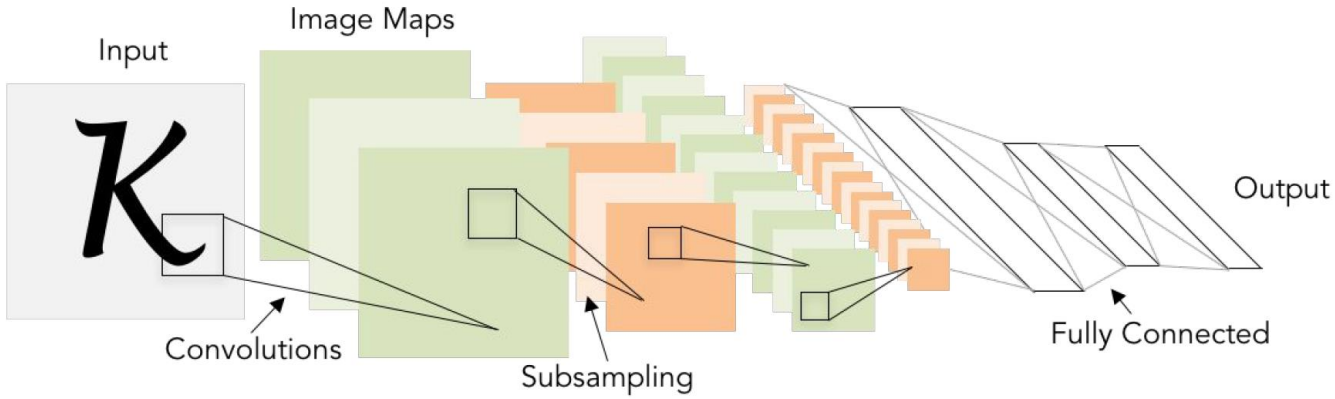
- Example: max pooling



Example CNNs

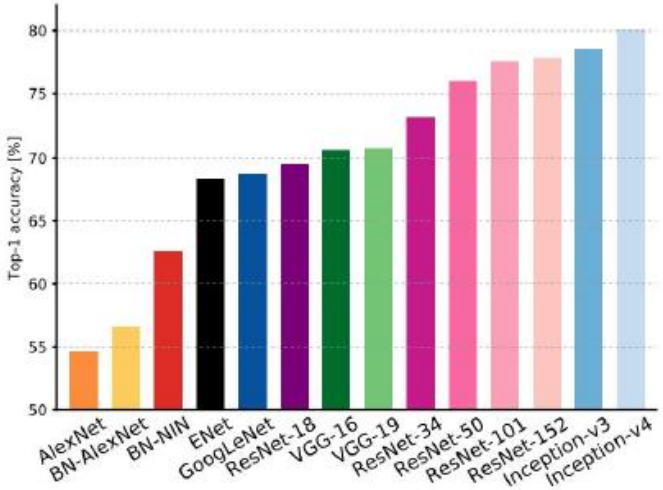
- LeNet-5 (Handwritten digit recognition)
- Sequential structure: AlexNet / VGGNet
- Parallel branches: GoogLeNet
- Residual structure: ResNet / DenseNet

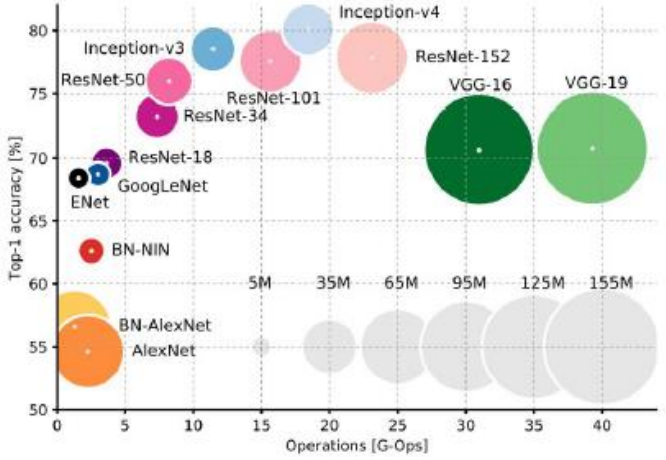
[LeCun et al., 1998]



Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-FC-FC]

Increasing model complexity





An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Autoencoder (AE)

Efficient Data Representations

Question: Which sequence is easier to remember?

- 40, 27, 25, 36, 81, 57, 10, 73, 19, 68

The shorter sequence seems easier?

- 50, 25, 76, 38, 19, 58, 29, 88, 44, 22, 11, 34, 17, 52, 26, 13, 40, 20

What if there are some rules that we can follow?

Efficient Data Representations

Question: Which sequence is easier to remember?

- 40, 27, 25, 36, 81, 57, 10, 73, 19, 68
- 50, 25, 76, 38, 19, 58, 29, 88, 44, 22, 11, 34, 17, 52, 26, 13, 40, 20

If you can remember the start of sequence 50, the length of sequence and two rules, then it is easy to reconstruct the whole sequence

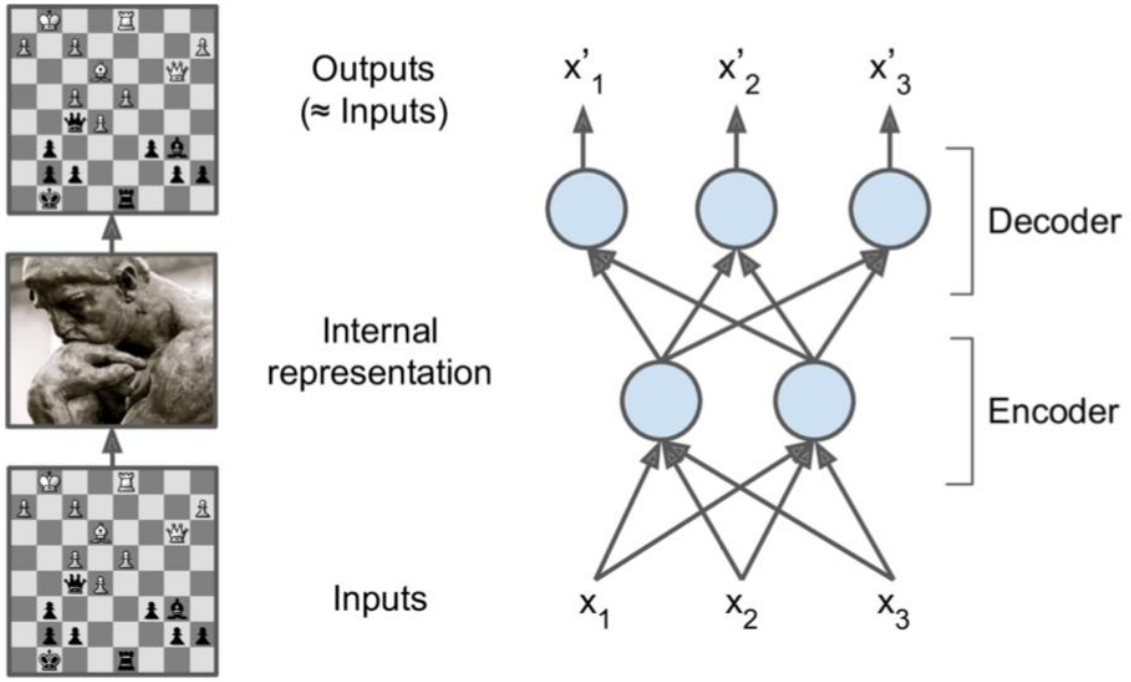
Chess memory experiment

Professional chess players can memorize positions of all pieces on a chessboard in 5 seconds. How can they do that?

Step 1: Look at the chessboard

Step 2: Analyze (encode) the chessboard

Step 3: Recover (decode) the chessboard

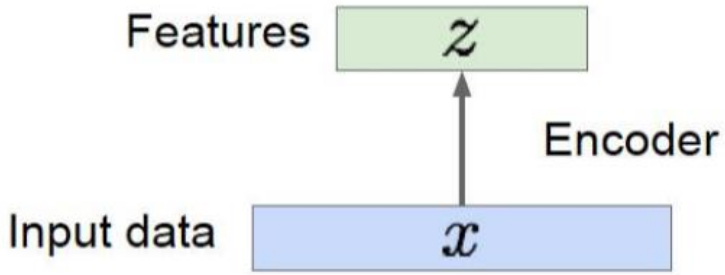


Internal representation has a lower dimensionality than the input data

Autoencoders

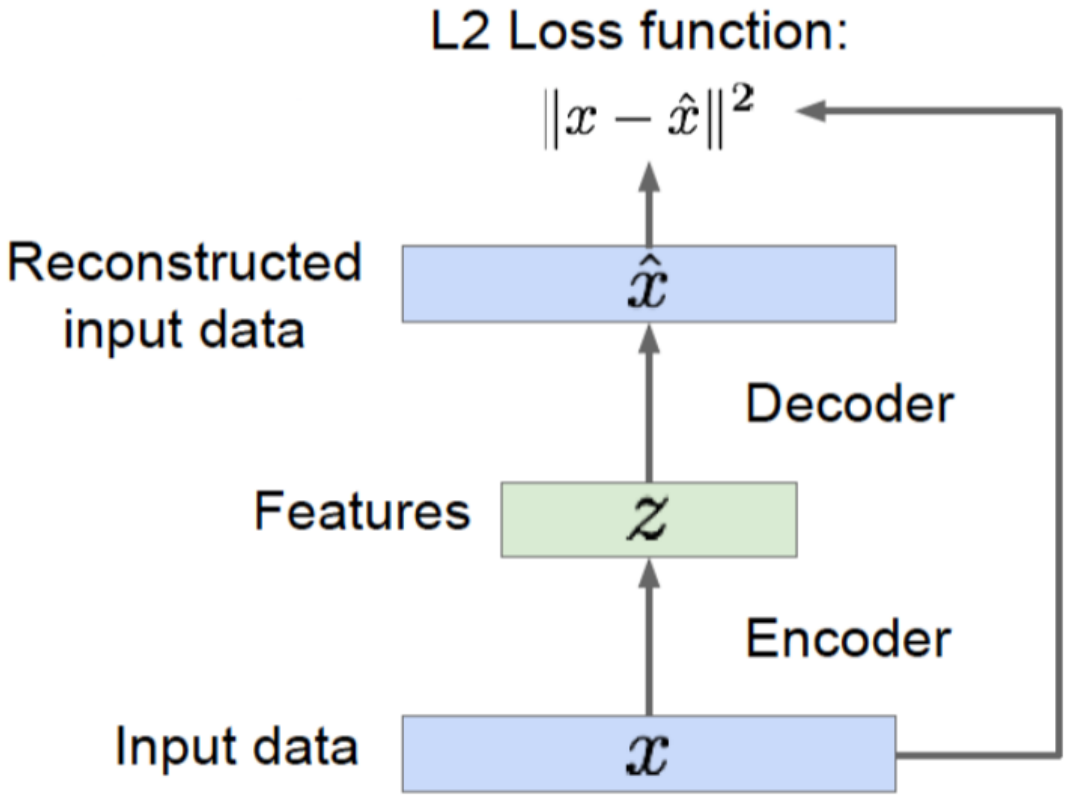
- The above two examples show that, by finding internal representations with lower dimensionality than the input data, we can find important patterns in the data

- **Autoencoder (AE)** is an unsupervised neural network for learning a lower-dimensional feature representation from unlabeled training data
  - It tries to reconstruct the input with lower dimensions (less neurons)



Training an Autoencoder

- How to train an autoencoder?
  - x : input data
  - z : internal representation or feature
  - $\hat{x}$: reconstructed data
  - $\|x - \hat{x}\|^2$: reconstruction error (or reconstruction loss)
- Then, an optimizer (e.g. Adam) can be used to minimize the reconstruction loss

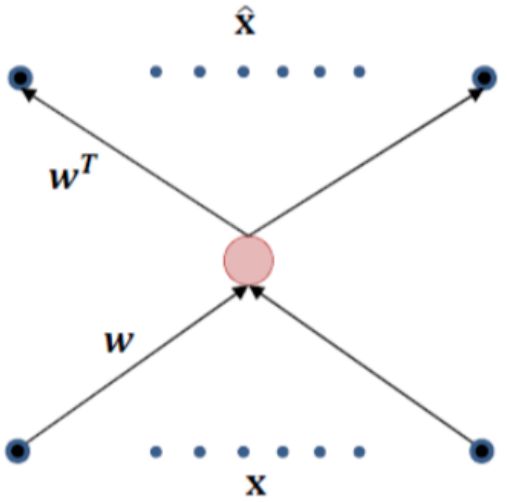


Principal Components Analysis (PCA)

- What is Principal Components Analysis (PCA)?
  - Reduce the data dimensionality by linear transformation without losing much information
- Formulation:
  - Suppose there are n samples in an m-dimension space $X = \{x_1, x_2, \dots, x_n\}$
  - Project the sample points into k dimension ($k < m$):
    - the projection matrix is $W = \{w_1, w_2, \dots, w_n\}$, and W is orthogonal
    - the projected points is $Z = WX$

Relation between AE and PCA

- What if the encoding of AE is linear ?
  - Encoding process becomes a linear projection $W \in \mathbb{R}^{n \times m}$ from a high-dimensional space $x \in \mathbb{R}^n$ to a low-dimensional space $z = Wx$
  - Decoding process becomes $\hat{x} = W^T W x$
  - Reconstruction error is $\|x - W^T W x\|^2$, the same as PCA
  - Therefore, PCA is a special (linear) form of AE



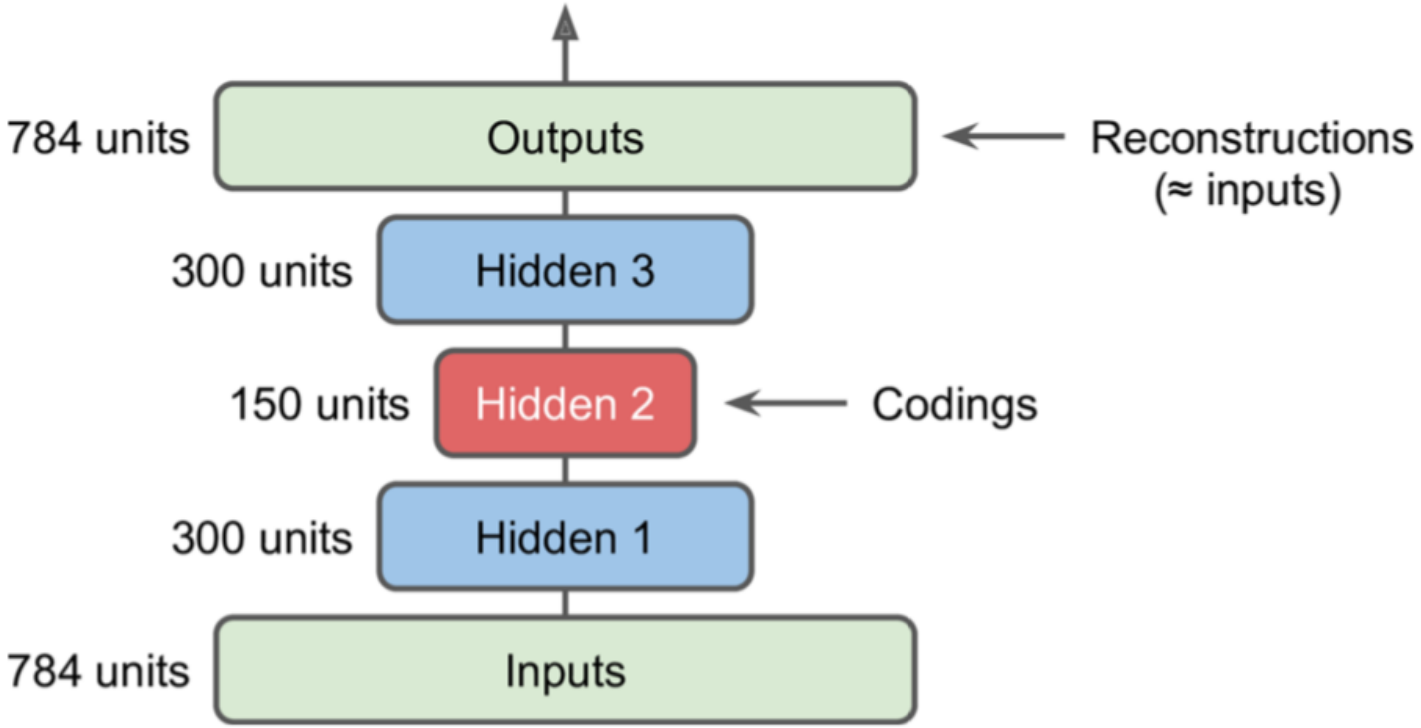
Example Autoencoders

Example AEs

- **Stacked (or deep) Autoencoder**: more hidden layers than standard AEs
- **Denoising Autoencoder (DAE)**: reconstruct the input from a corrupted (by adding noise) version of input
- **Sparse Autoencoder**: limit the amount of total activation for a neuron throughout training
- **Variation Autoencoder (VAE)**: An AE whose training is regularized
  - To avoid overfitting
  - To ensure that the latent space has good properties that enable generative process
  - Instead of encoding an input as a single point, VAE encodes it as a distribution over the latent space

Stacked (or deep) autoencoder

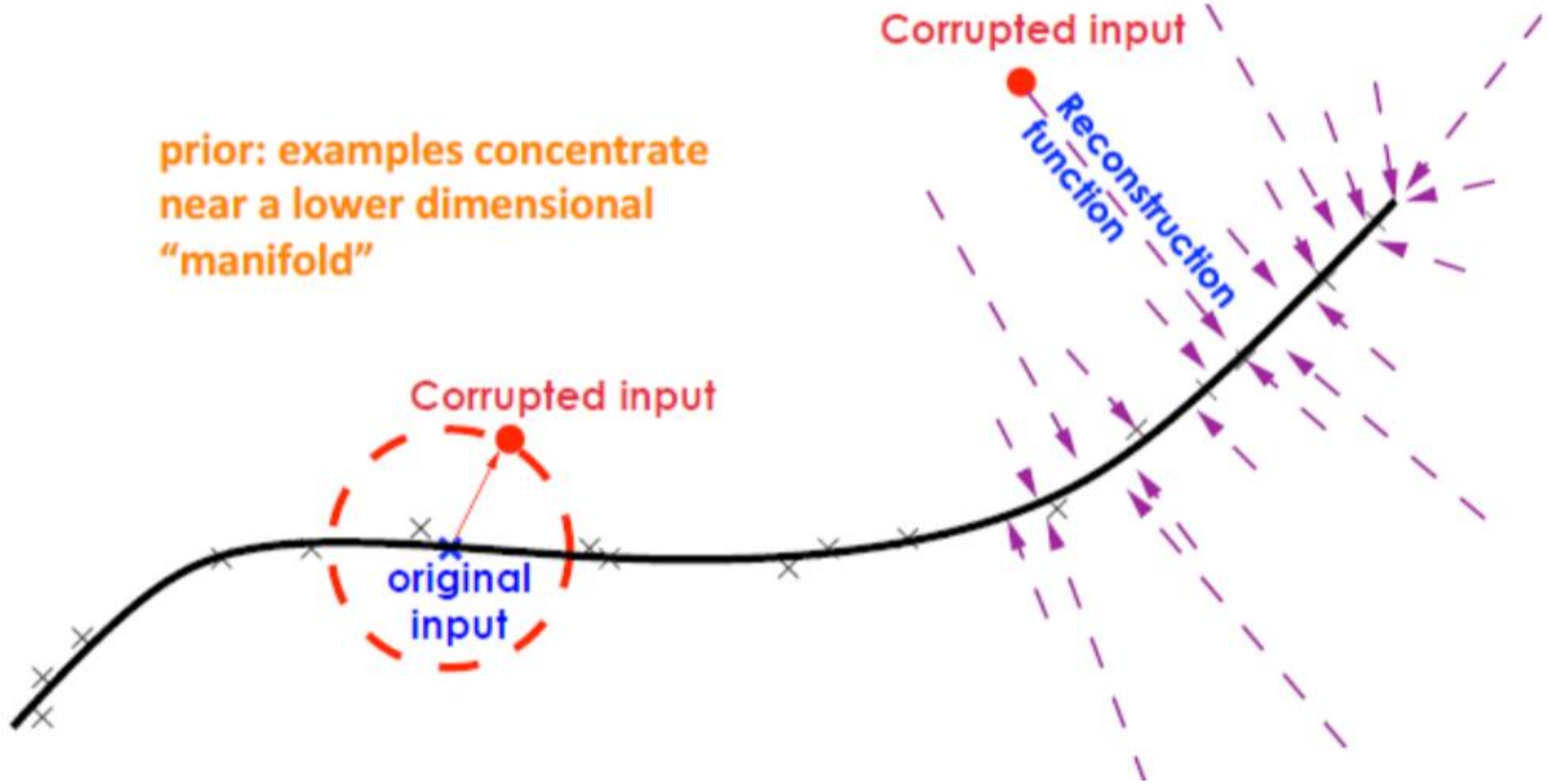
- More hidden layers than standard Autoencoders



Denoising Autoencoder (DAE)

- What is DAE?
  - Randomly corrupt some of the inputs
  - Train the autoencoder to reconstruct the input from a corrupted version of it
  - The autoencoder is forced to predict the original inputs
  - This requires to capture the joint distribution between a set of variables
- **Rationale**: To increase the difficulty of reconstructing the output by adding noise

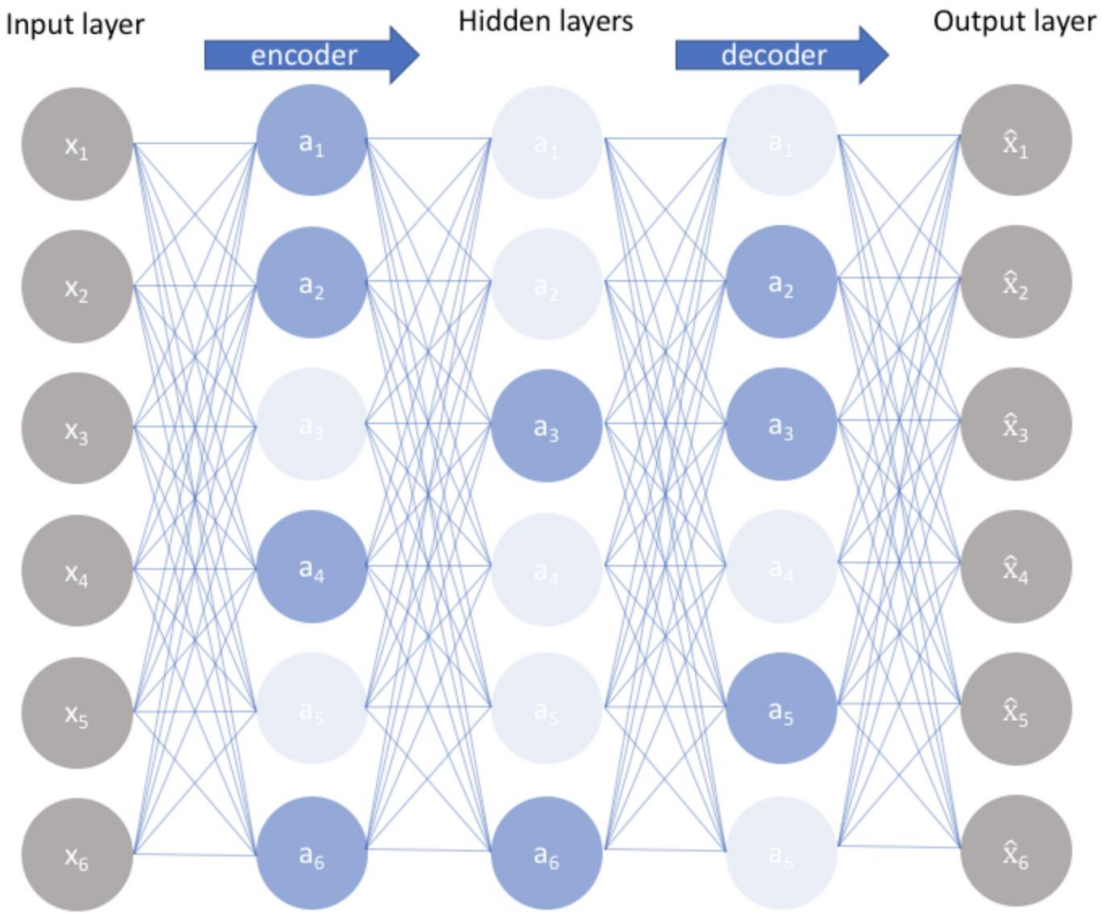
Geometric view of DAE



Sparse Autoencoder

- Usually, the dimension of hidden layers in an AE is small

- What if the dimension of hidden layers k in the AE is large ?
  - We call it over-completeness
  - This can be solved by enforcing sparsity

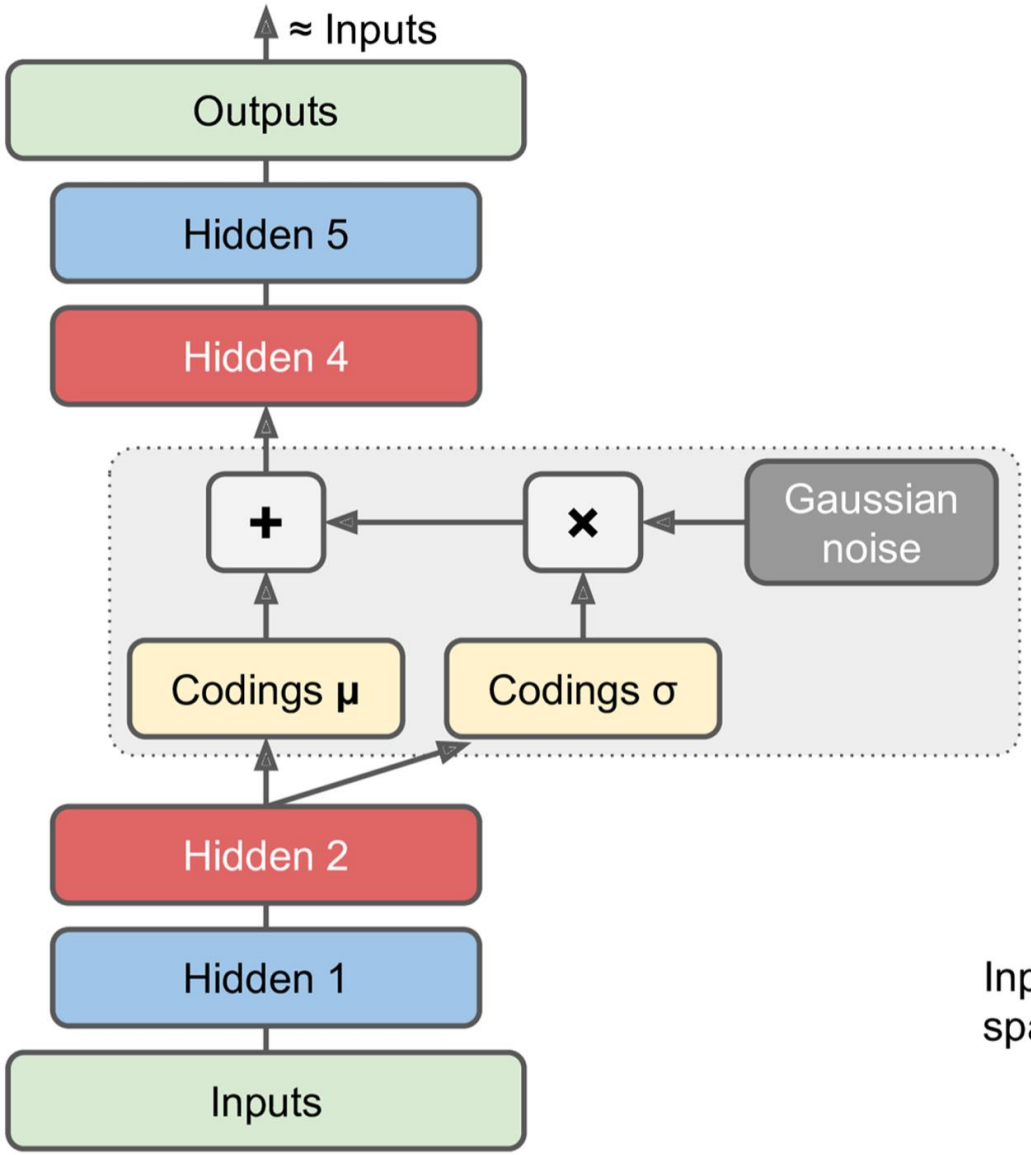


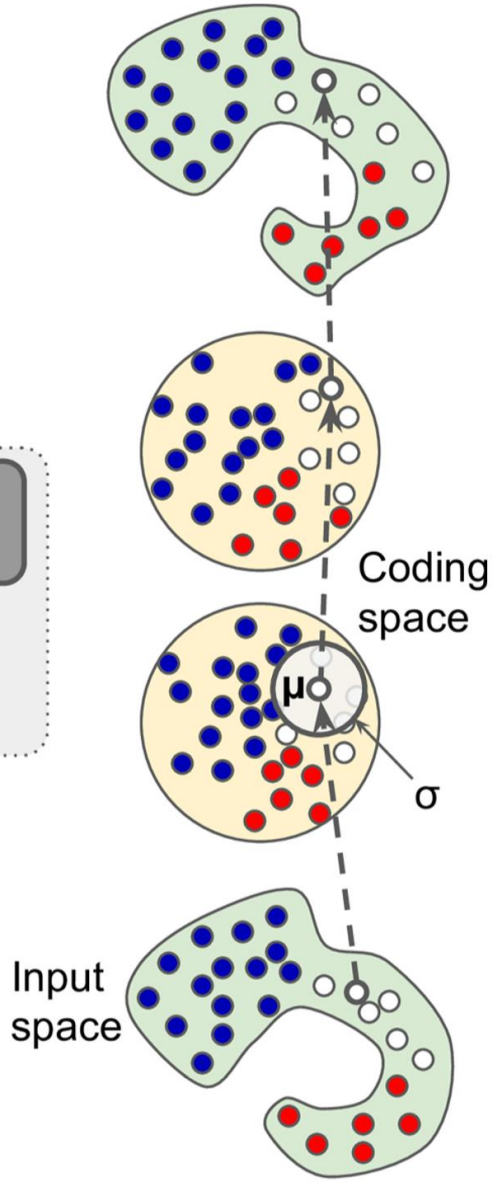
Variational Autoencoder (VAE)

- A **variational autoencoder (VAE)** can be defined as being an autoencoder whose training is regularized
  - To avoid overfitting, and
  - To ensure that the latent space has good properties that enable generative process

- Instead of encoding an input as a single point, VAE encodes it as a distribution over the latent space

Variational Autoencoder (VAE)





Recap

- In this lecture, we have learned:
  - Goals and main ideas of CNN
  - Key components of a CNN model (convolutional layers, pooling layers)
  - The rationale of autoencoders with efficient data representations (i.e. to reconstruct inputs in a lower-dimensionality space)
  - Relation between PCA and autoencoders
  - Several variants of autoencoders
    - Stacked (or deep) AE
    - Denoising AE
    - Sparse AE
    - Variational AE

References

- Book “Dive into Deep Learning” (http://d2l.ai/), Chapters 7 and 8.
- Yann LeCun et al. “Gradient-based learning applied to document recognition”, Proceedings of the IEEE 86, no. 11 (1998): 2278-2324.
- Aurélien Géron’s book “Hands-On Machine Learning with SciKit-Learn & TensorFlow” (Ed. 2), O’Reilly, 2019, Chapters 14 and 17.
- Ian Goodfellow et al. “Deep Learning”, MIT Press, 2017, Chapters 9 and 14.