西安交通大學

XI'AN JIAOTONG UNIVERSITY

Deep Learning Seminar

Session 1

Oct 13

00

入门介绍

Ill. Niklas Elmehed © Nobel Prize Outreach

John J. Hopfield

Prize share: 1/2

Ill. Niklas Elmehed © Nobel Prize Outreach

Geoffrey E. Hinton

Prize share: 1/2

The Nobel Prize in Physics 2024 was awarded jointly to John J. Hopfield and Geoffrey E. Hinton "for foundational discoveries and inventions that enable machine learning with artificial neural networks"

Ill. Niklas Elmehed © Nobel Prize Outreach

David Baker

Prize share: 1/2

Ill. Niklas Elmehed © Nobel Prize Outreach

Demis Hassabis

Prize share: 1/4

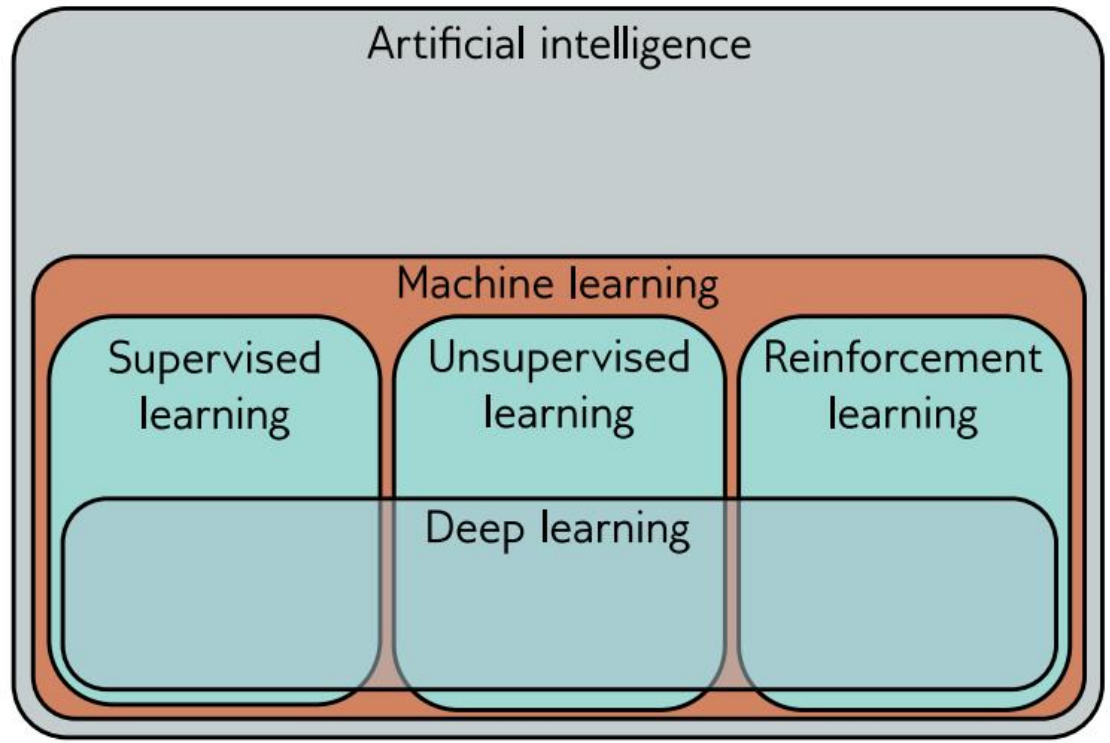
Ill. Niklas Elmehed © Nobel Prize Outreach

John M. Jumper

Prize share: 1/4

The Nobel Prize in Chemistry 2024 was divided, one half awarded to David Baker "for computational protein design", the other half jointly to Demis Hassabis and John M. Jumper "for protein structure prediction"

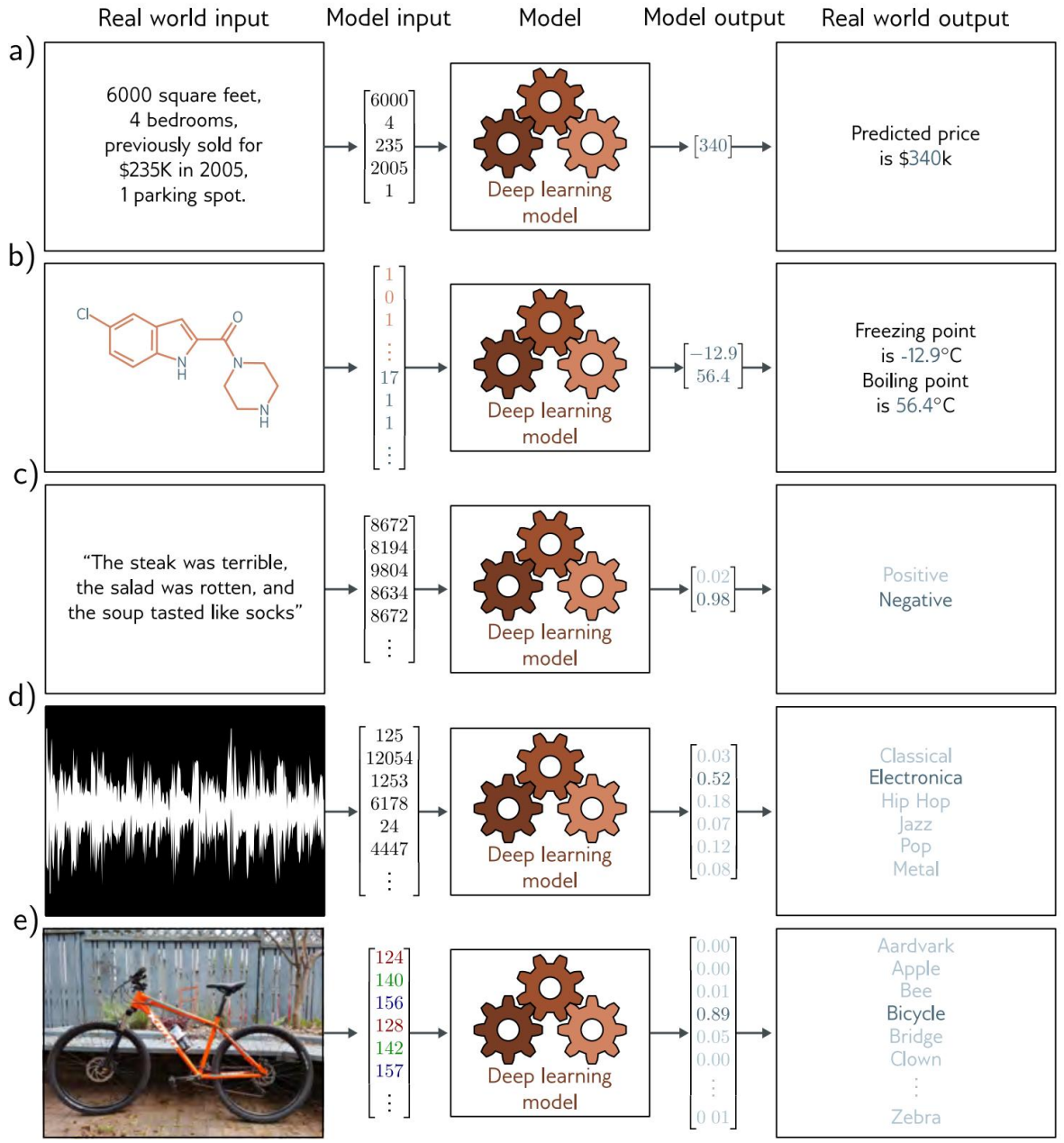
Figure 1.1 Machine learning is an area of artificial intelligence that fits mathematical models to observed data. It can coarsely be divided into supervised learning, unsupervised learning, and reinforcement learning. Deep neural networks contribute to each of these areas.



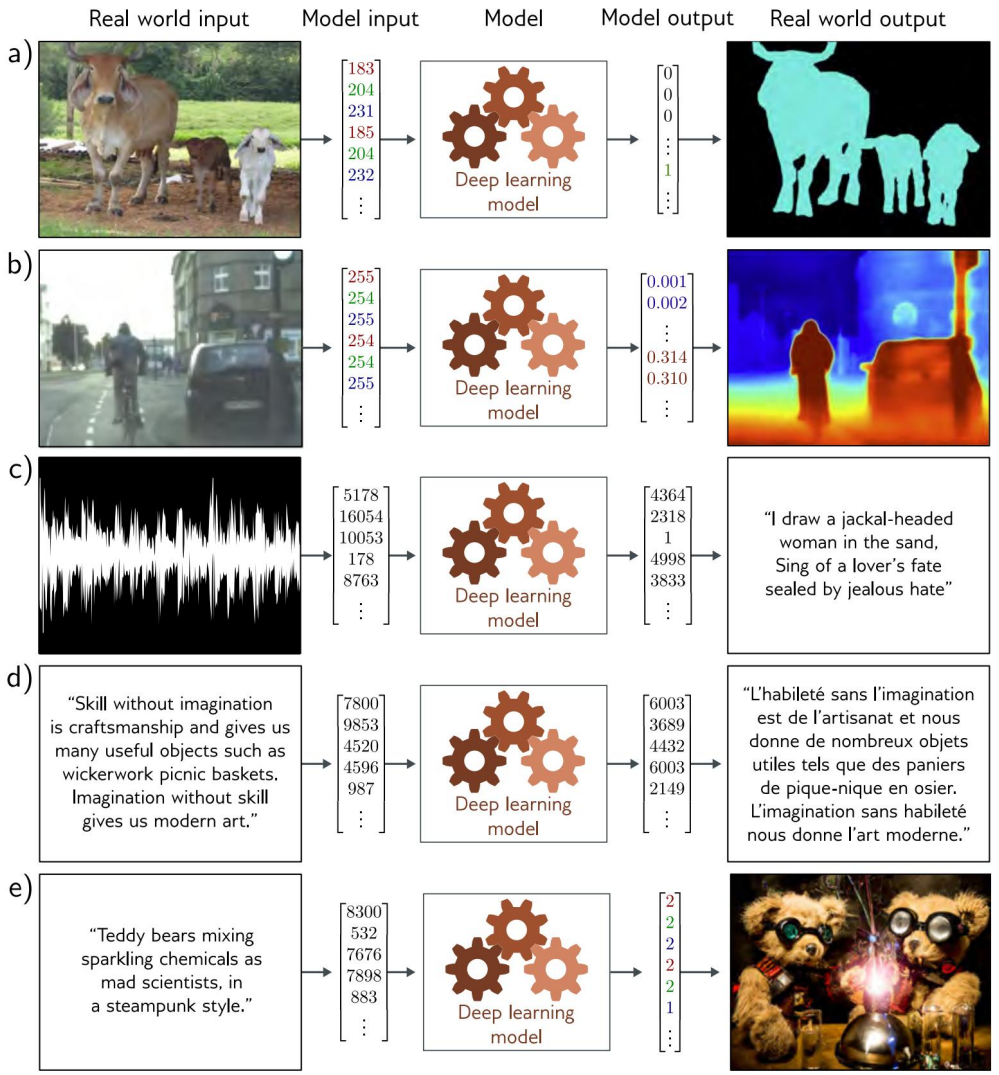
Function describe the world

Regression

Classification

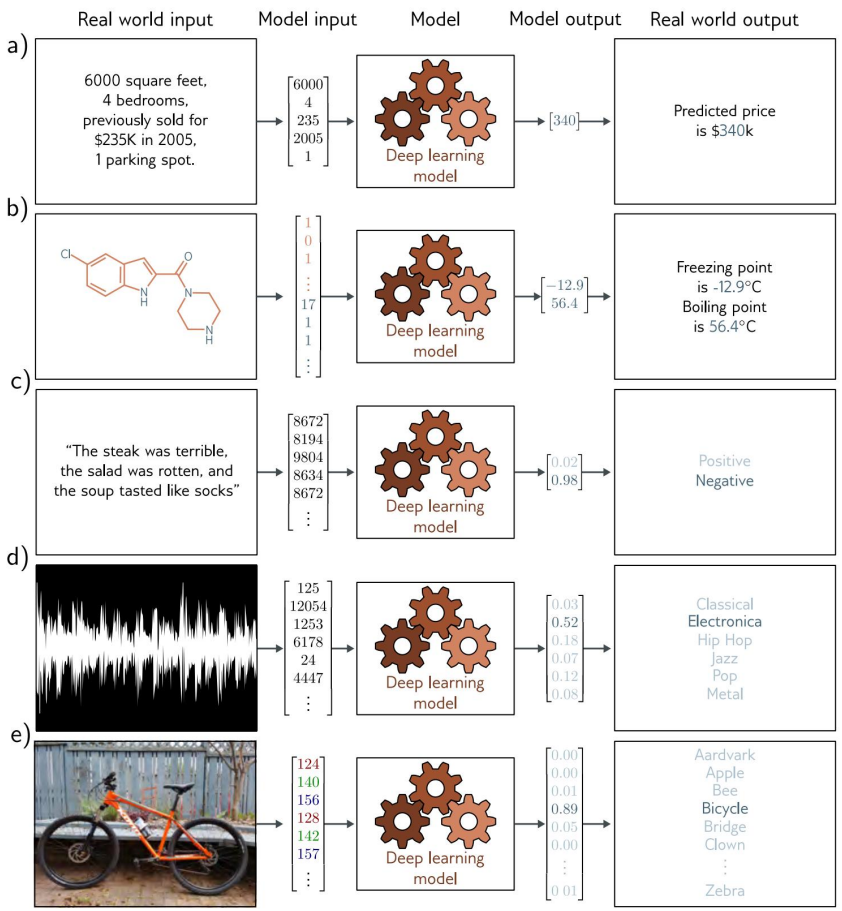


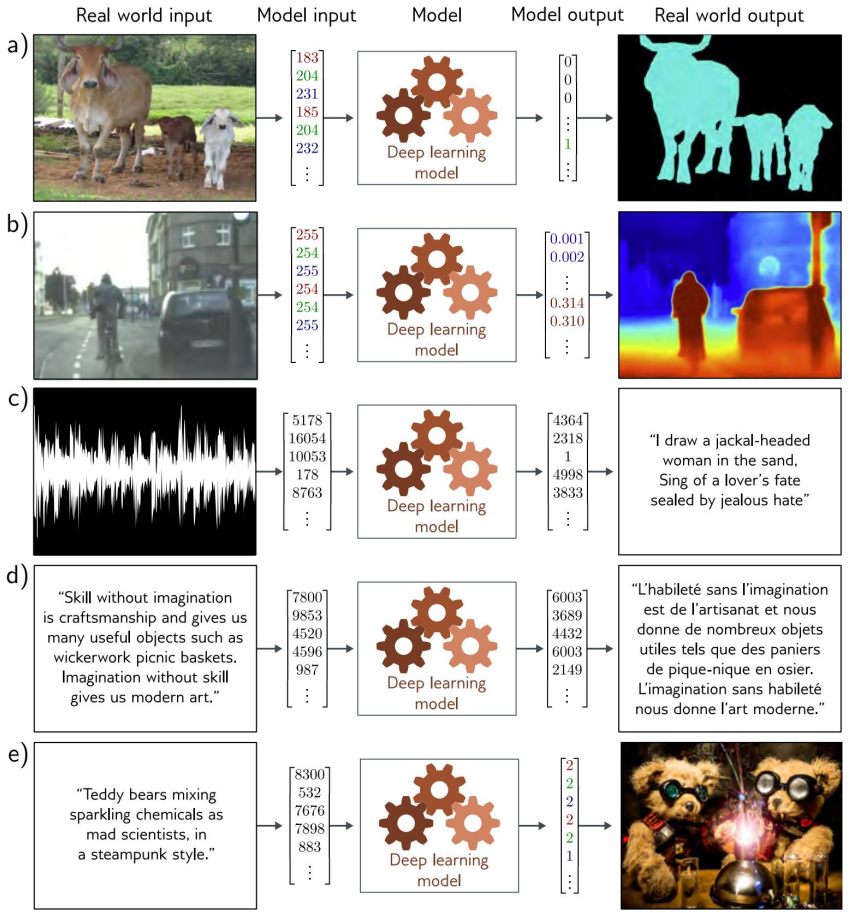
Function describe the world



Supervised learning VS Unsupervised learning

Deep learning is representation learning





Raw data

Representation space

~ Latent space

How to Represent An Image?

→ class

- Domain knowledge required
- But what's next?

→ edge → class

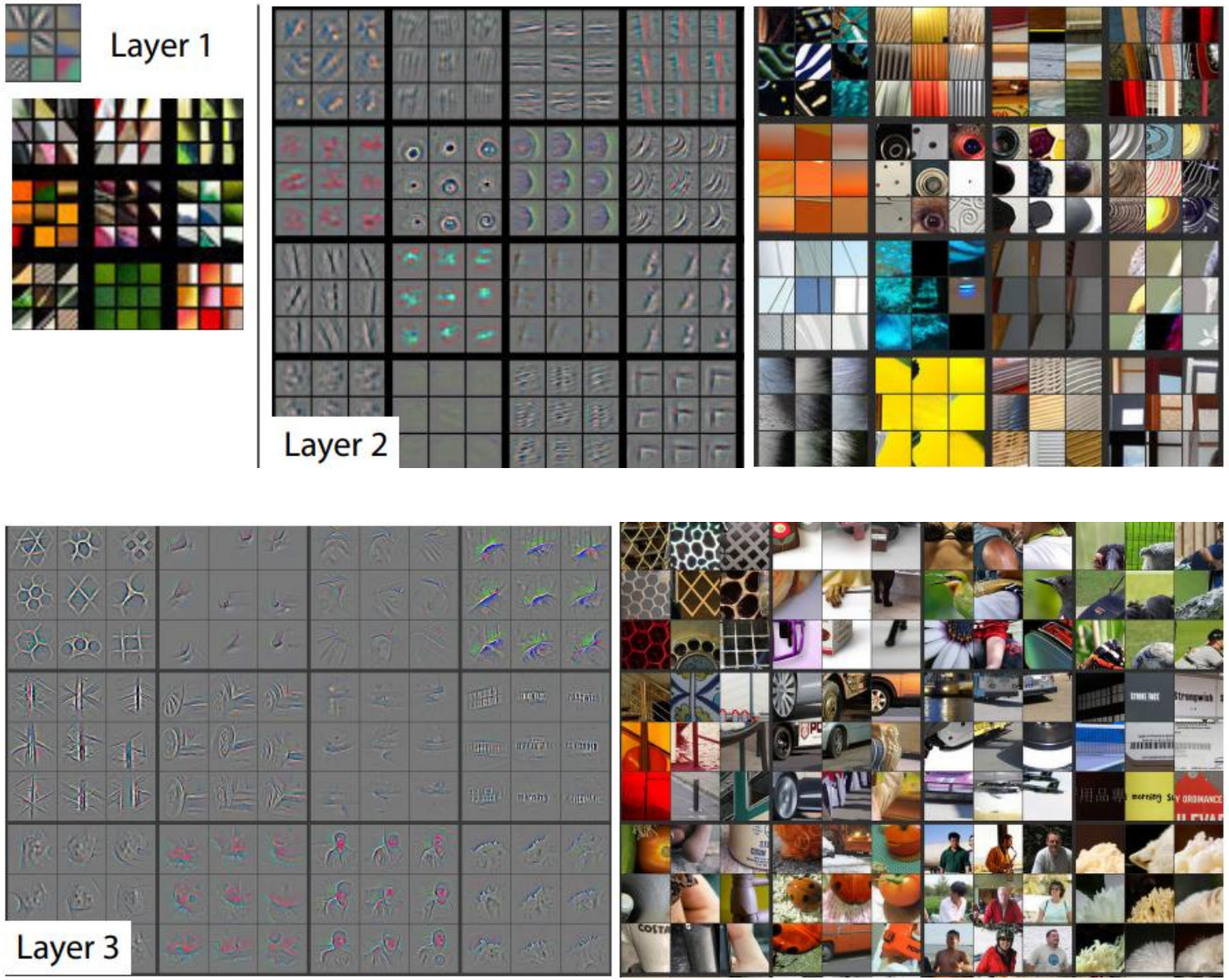
→ edge → orientation → class

deeper

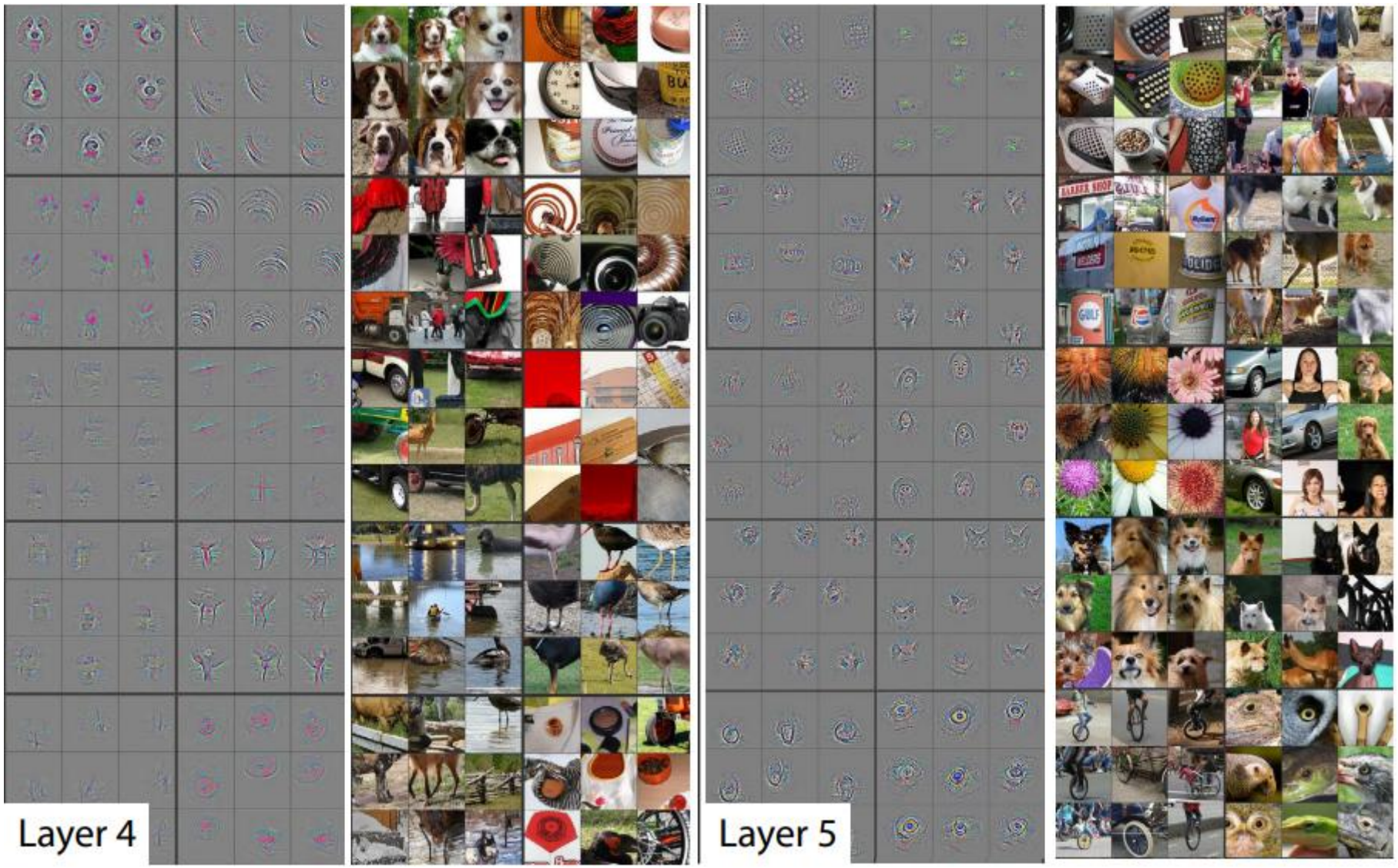
→ edge → orientation → histogram → class

→ edge → orientation → histogram → clusters → class

Lowe 1999, Sivic & Zisserman 2003, Dalal & Triggs 2005, Grauman & Darrell 2005, Lazebnik et al 2006, Perronnin & Dance 2007, Yang et al 2009, Jégou et al 2010, …



"Visualizing and Understanding Convolutional Networks", Zeiler and Fergus, arXiv 2013, ECCV 2014



“Visualizing and Understanding Convolutional Networks”, Zeiler and Fergus, arXiv 2013, ECCV 2014

01

数据拟合

- 数据拟合旨在从数据中学习到一个数学模型，使其能很好的逼近产生数据的客观规律。

数据集:

$$S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n} \subset \mathcal{X} \times \mathcal{Y},$$

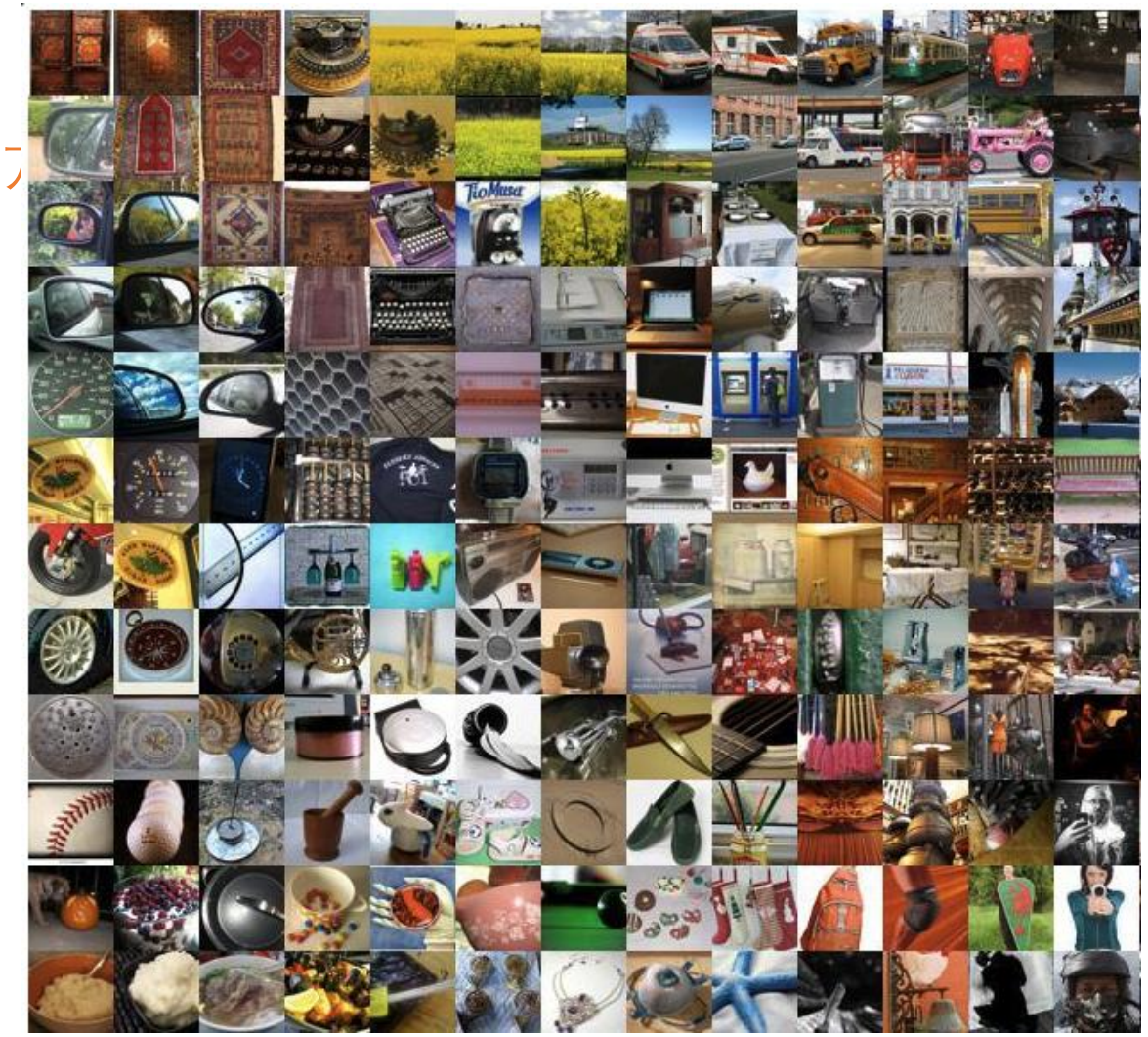
目标函数:

$$f^* : \mathcal{X} \to \mathcal{Y} \qquad\qquad y_i = f^*(\boldsymbol{x}_i)$$

注:

1. f^*通常未知，但可以通过采样得到数据集S.
2. 机器学习：使用带有参数θ的模型f_θ逼近目标函数。

- ImageNet数据集
  - 该项目已手动标注了1400多万张图像，包含2万
  - 常见ILSVRC 2012 （ImageNet-1k）
- 数据$S = \{(x_i, y_i)\}_{i=1}^{n}$?
  - $n \approx 1.28 \times 10^6$
  - $\boldsymbol{x}_i \in [0,255]^{256\times256}$
  - $\boldsymbol{y}_i \in \{0,1\}^{1000}$
- 该数据集背后的目标函数是什么?
  - 人类大脑的视觉识别函数



https://cs.stanford.edu/people/karpathy/cnnembed/

当给定一组数据的时候，我们希望从这个**函数空间**中找到一个最佳函数来逼近这组数据背后的**目标函数**。为了实现这一点，我们通常会定义一个**损失函数** (loss function) 来衡量模型预测值和真实值之间的差距，然后通过优化算法调整 θ ，以最小化损失函数。这个过程就是我们通常说的**模型训练**。通过这种方法，我们希望最终得到的模型 f_θ 能够在整个输入空间，特别是在未见输入上，与目标函数f^*足够接近。

损失函数 L，常用的均方差：

$$\ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}), f^*(\boldsymbol{x})) = (f_{\boldsymbol{\theta}}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2.$$

训练误差L_s：

$$L_S = \frac{1}{n}\sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), y_i).$$

最小化训练误差，得到在训练数据上误差最小的模型：

$$\min_{\boldsymbol{\theta}} L_S.$$

我们关心的其实是泛化误差L_D(*generalization error*)，也就是那些没有被用来训练的测试集上的误差，

$$L_{\mathcal{D}} = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}), f^*(\boldsymbol{x})),$$

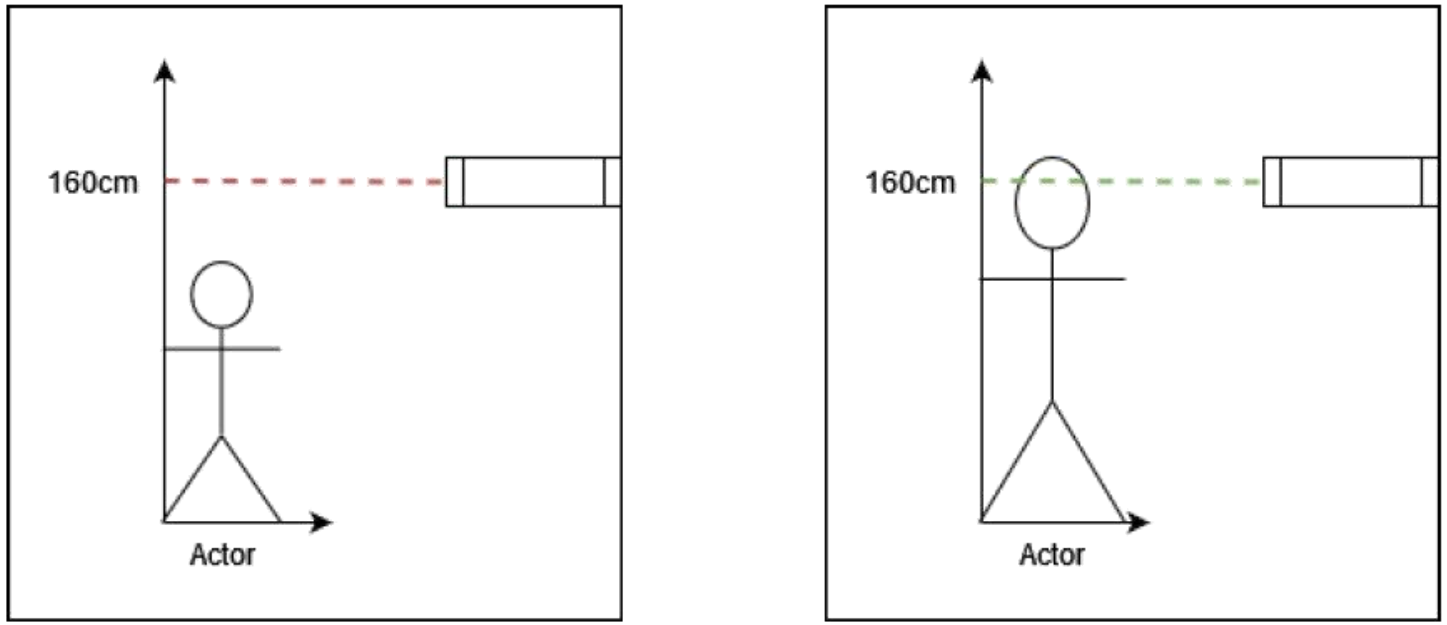
02

神经网络简介

神经网络简介

1. 神经网络的定义：
   - 神经网络是一种受人脑结构启发的数学模型。
   - 它用于处理复杂的数据关系。
2. 神经网络的组成：
   - 神经网络由多层相互连接的“神经元”组成。
   - 每一层的神经元将信息传递到下一层。
3. 神经网络的核心功能：
   - 它通过大量数据的“训练”来自动学习复杂的模式。
   - 每个神经元有自己的权重和偏置，这些参数在训练过程中会调整。
4. 网络结构：
   - 简单的神经网络包含输入层、隐藏层和输出层。
   - 复杂的神经网络可能包含多个隐藏层，甚至不同的结构，每一层提取不同特征。
5. 神经网络的应用：
   - 神经网络可以解决图像识别、语音处理和自然语言处理等复杂问题。
6. 神经网络与数据拟合的关系：
   - 神经网络是一种高度灵活且强大的数据拟合工具。
   - 其目标是学习一个函数，该函数能够最好地描述输入数据和输出结果之间的关系。

- 感知信息的基本条件是对不同的输入能够区分

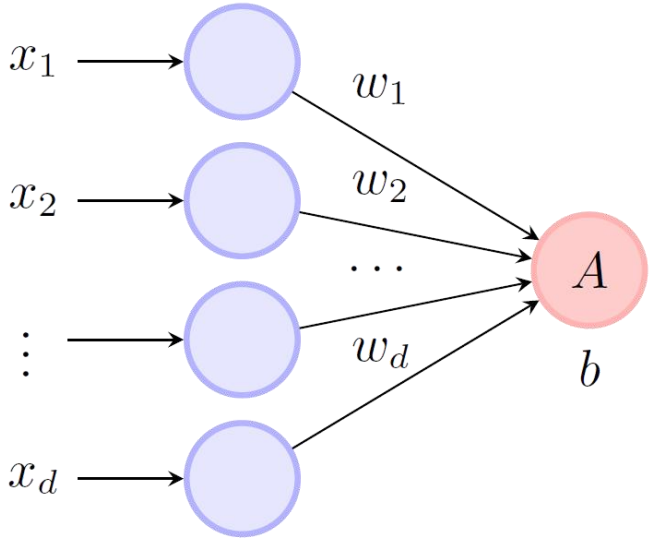


(a) 未达到 160cm　　(b) 达到 160cm

图 1.1: 身高提取器

$$\sigma(x) = \begin{cases} 1 & x \geq T \\ 0 & x < T \end{cases},$$

- 我们可以把这类收集信息的方式推广到一般的形式

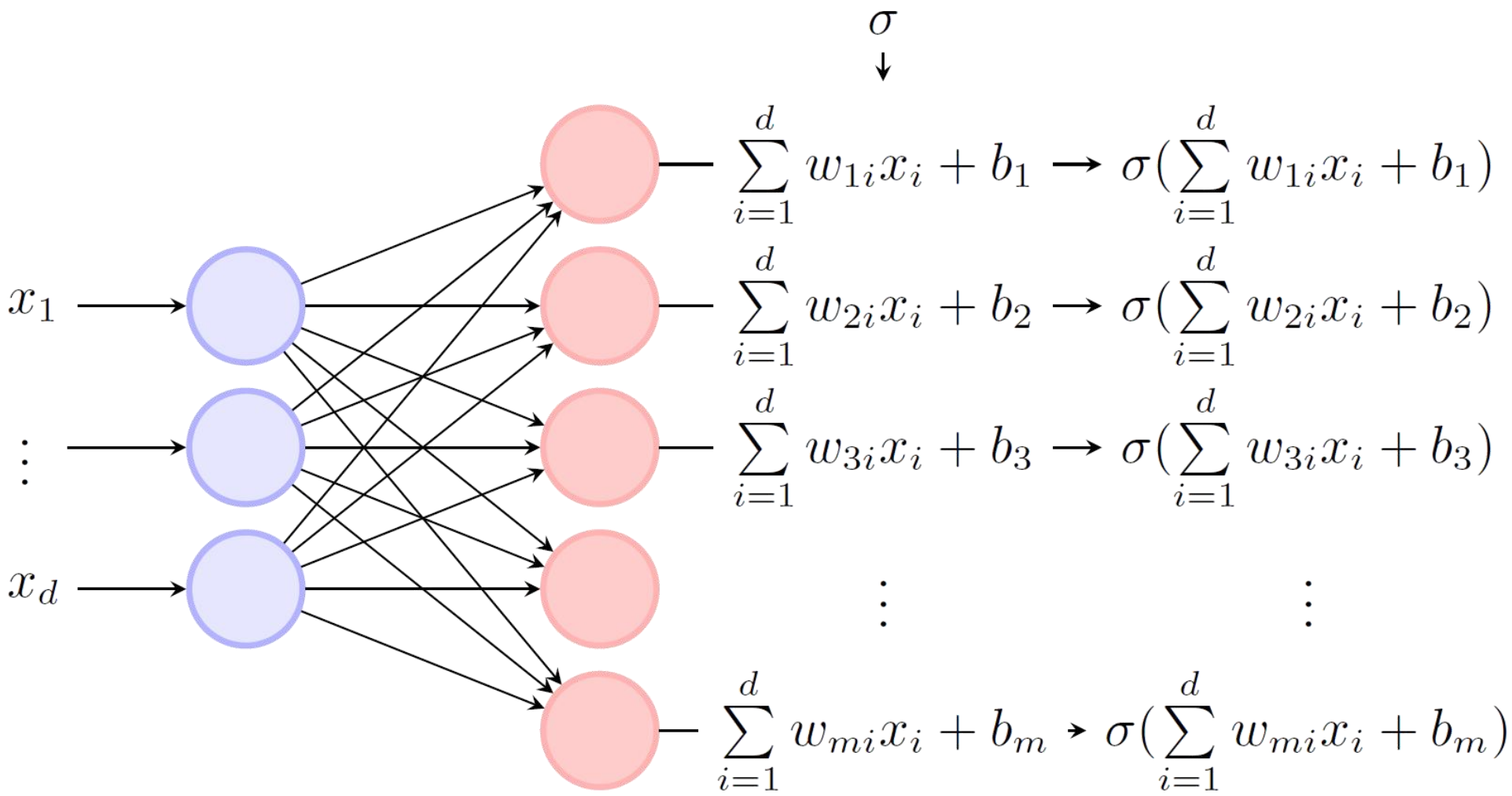


$$A = \sum_{i=1}^{d} w_i x_i + b$$

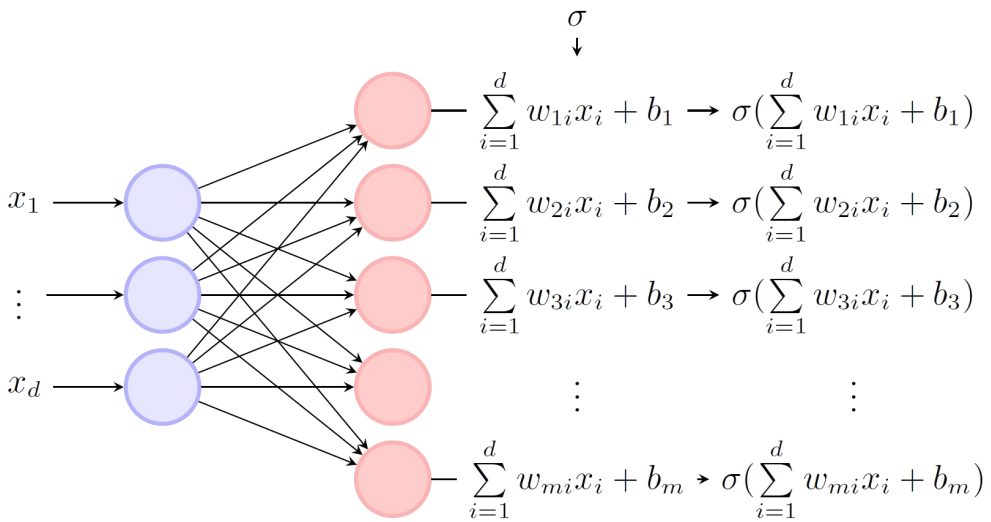
$$\begin{aligned} & w_1 x_1 + \cdots + w_d x_d + b \\ &= (w_1, \cdots, w_d) \cdot (x_1, \cdots, x_d) + b \\ &= \boldsymbol{w}^T \boldsymbol{x} + b. \end{aligned}$$

单个神经元的输出为 $\sigma(\boldsymbol{w}^T \boldsymbol{x} + b)$，$\sigma(\cdot)$ 通常叫做激活函数。

- 简单地把多个神经元堆在一起，我们就可以得到单层神经网络



- 单层神经网络



- 神经网络的高维输出：

$$y = \sigma \circ (W\boldsymbol{x} + \boldsymbol{b}),$$

- 参数的矩阵形式

$$W = \begin{pmatrix} w_{11}, & \cdots, w_{1d} \\ \vdots, & \cdots, \vdots \\ w_{m1}, & \cdots, w_{md} \end{pmatrix},$$

$$\boldsymbol{b} = (b_1, b_2, \cdots, b_m)^T.$$

- $\circ$ 表示对每个元素作用：

$$\sigma \circ ([0.1, 0.3, 0.5]) = [\sigma(0.1), \sigma(0.3), \sigma(0.5)]$$

- 考虑输入是两维的，$x_1, x_2 \in \{0,1\}^2$

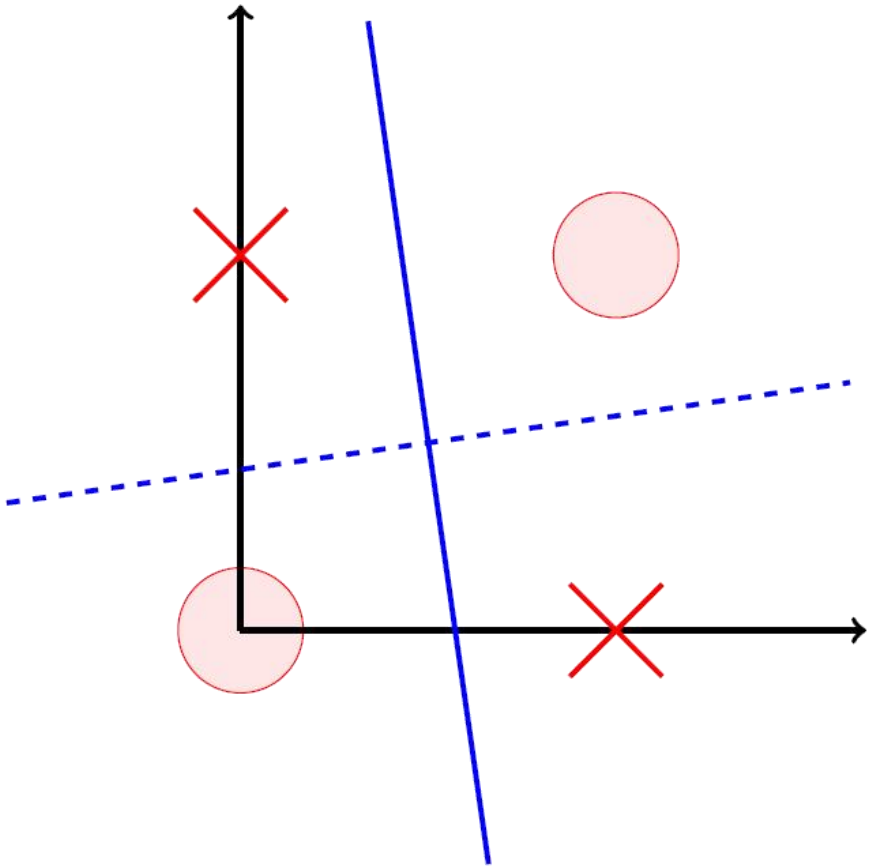
$$\sigma(x) = \begin{cases} 1 & x_1 + x_2 \geq T \\ 0 & x_1 + x_2 < T \end{cases}$$

AND: $T = 2$ Or: $T = 1$ Not: $T = 0$

$$\sigma(x) = -x$$

The Dartmouth Conference of 1956 was organized by Marvin Minsky, John McCarthy and two senior scientists: Claude Shannon and Nathan Rochester of IBM. The proposal for the conference included this assertion: **"every aspect of learning or any other feature of intelligence can be so precisely described that a machine can be made to simulate it"**.

- 单层神经网络是无法做好XOR问题，



- 多层神经网络是在一层神经网络的基础上堆叠起来的。

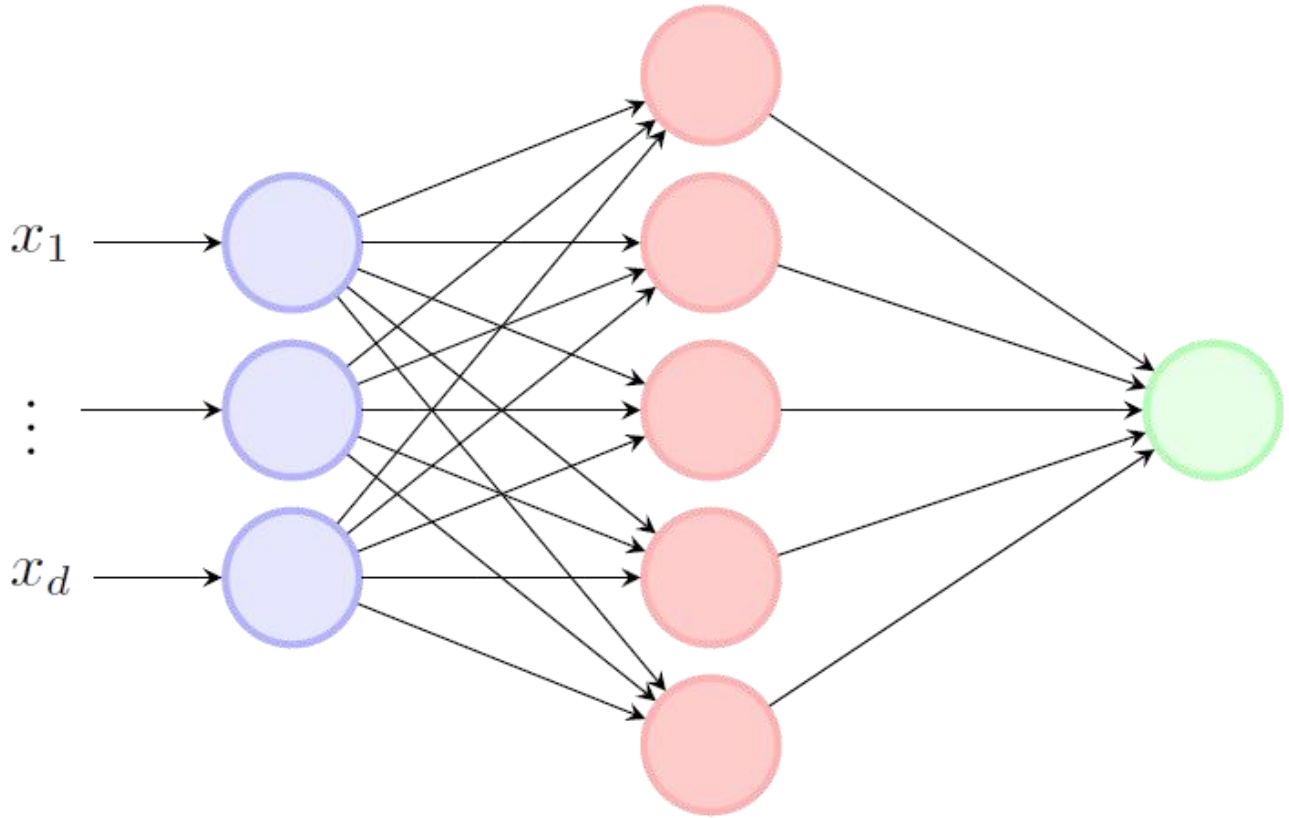


图 1.3: 两层神经网络

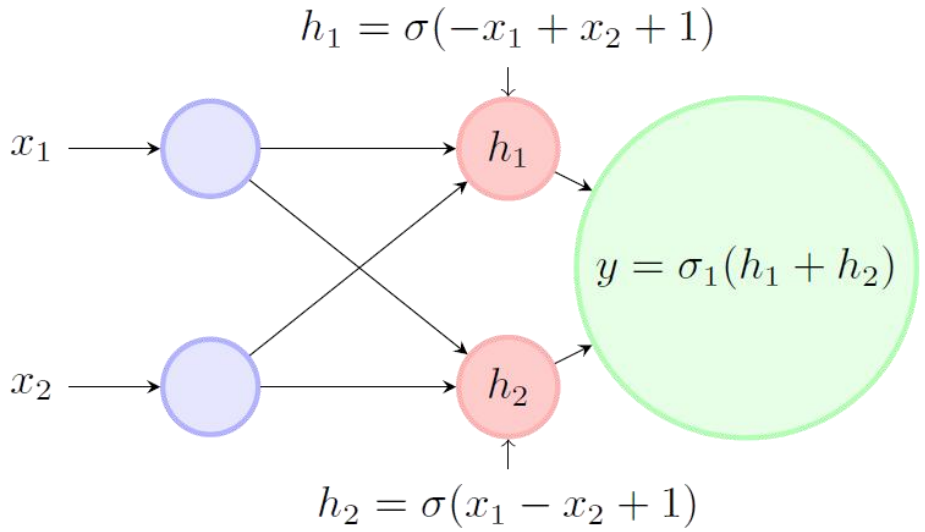


图 1.4: 两层网络解 XOR 问题

- 假设，$y = \sigma\left(\frac{h_1+h_2}{2}\right)$
- 代入(0,0), (1,1)，则$y = \sigma(1) = 1$。
- 代入(1,0), (0,1)，则$y = \sigma\left(\frac{1}{2}\right) = 0$。

如图 1.4所示，当 $x_1 = x_2 = 0$ 或者 $x_1 = x_2 = 1$ 时，$h_1 = h_2 = \sigma(1)$，而当 $x_1 = 1, x_2 = 0$ 或 $x_1 = 0, x_2 = 1$ 时，$h_{\{1,2\}} = \sigma(0), h_{\{2,1\}} = \sigma(2)$，这样就完成 XOR 问题的分类了。

- **万有逼近定理（Cybenko 1989, Hornik et al. 1989, Leshno et al. 1993）**

在 1980 年代末，一系列的工作证明当激活函数是非线性且非多项式的时候，对任意一个从有限维空间 X 到有限维空间 Y 的连续函数，当神经网络的隐藏层足够宽时，两层神 经网络可以以任意精度逼近该函数，这也就是万有逼近定理（universal approximation theorem。万有逼近定理展现了神经网络具有很强的逼近能力，给神经网络的使用提供了一个**重要的理论支撑**

- 万有逼近定理对实际问题有什么帮助?
- 万有逼近定理是否足够让我们使用神经网络?
- 为什么在实际中还要使用多层神经网络?
- 定理证明的解是否在实际训练中可以被找到？我们需要用什么样的训练方法，需要多少数据?
- 神经元的数目会不会多到我们实际训练无法承受?
- ……

当给定一组数据的时候，我们希望从这个**函数空间**中找到一个最佳函数来逼近这组数据背后的**目标函数**。为了实现这一点，我们通常会定义一个**损失函数** (loss function) 来衡量模型预测值和真实值之间的差距，然后通过优化算法调整 θ ，以最小化损失函数。这个过程就是我们通常说的**模型训练**。通过这种方法，我们希望最终得到的模型 f_θ 能够在整个输入空间，特别是在未见输入上，与目标函数f^*足够接近。

损失函数 L，常用的均方差：

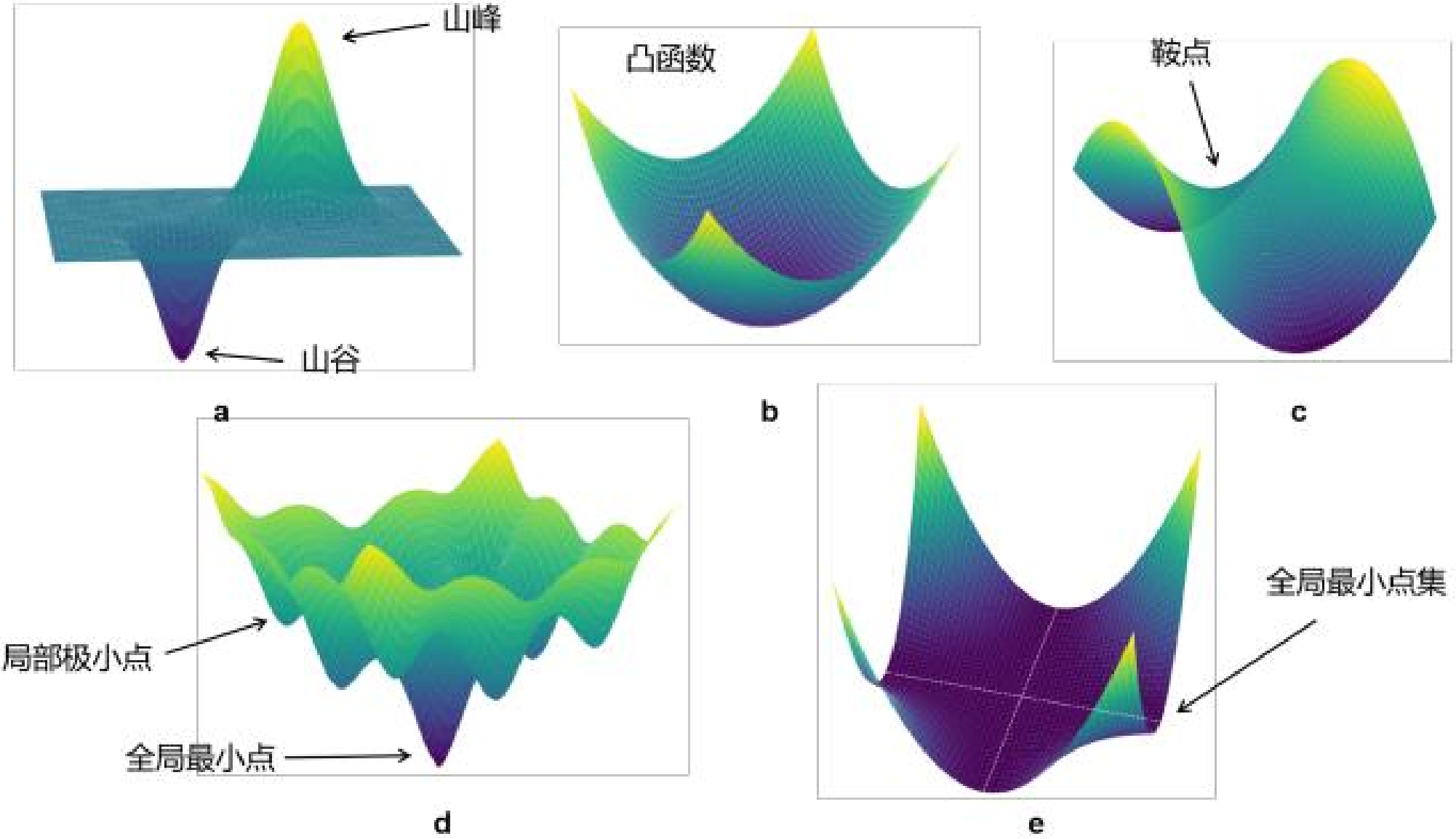
$$\ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}), f^*(\boldsymbol{x})) = (f_{\boldsymbol{\theta}}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2.$$

训练误差L_S：

$$L_S = \frac{1}{n}\sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), y_i).$$

最小化训练误差，得到在训练数据上误差最小的模型：

$$\min_{\boldsymbol{\theta}} L_S.$$


山峰
山谷
a
凸函数
b
鞍点
c
局部极小点
全局最小点
d
全局最小点集
e