神经网络
Neural Network

人工智能学院

人工智能与网络搜索教研中心

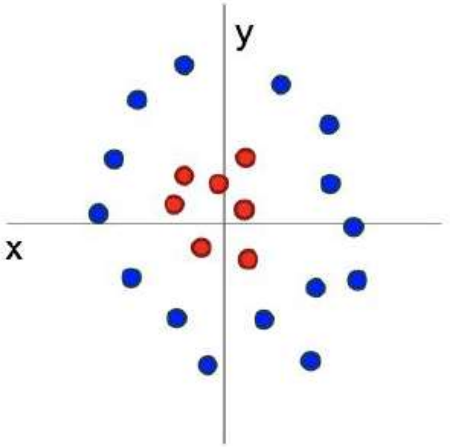
胡佳妮 jnhu@bupt.edu.cn

问题：线性分类器的能力是有限的

视觉角度

线性分类器对每个类学习一个模板

几何角度



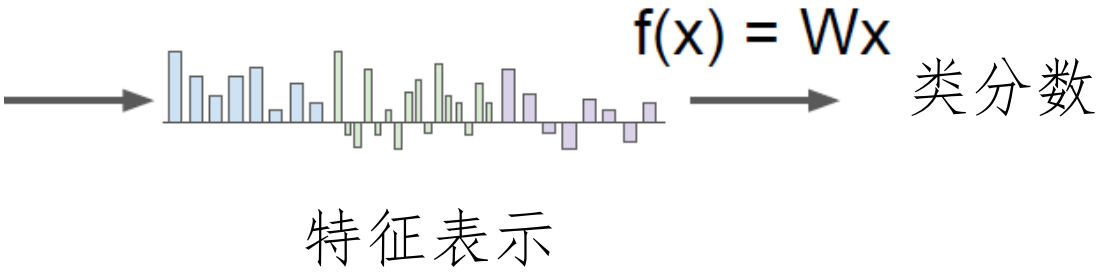
线性分类器只能学习线性决策边界

像素特征

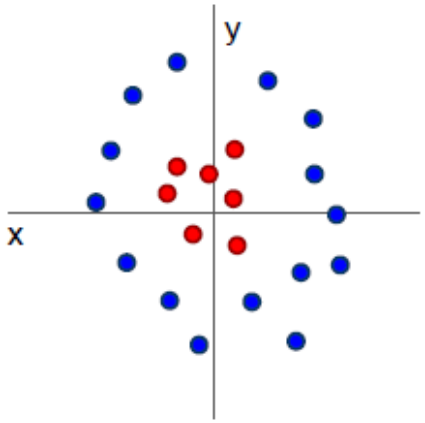
类分数

f(x) = Wx

图像特征

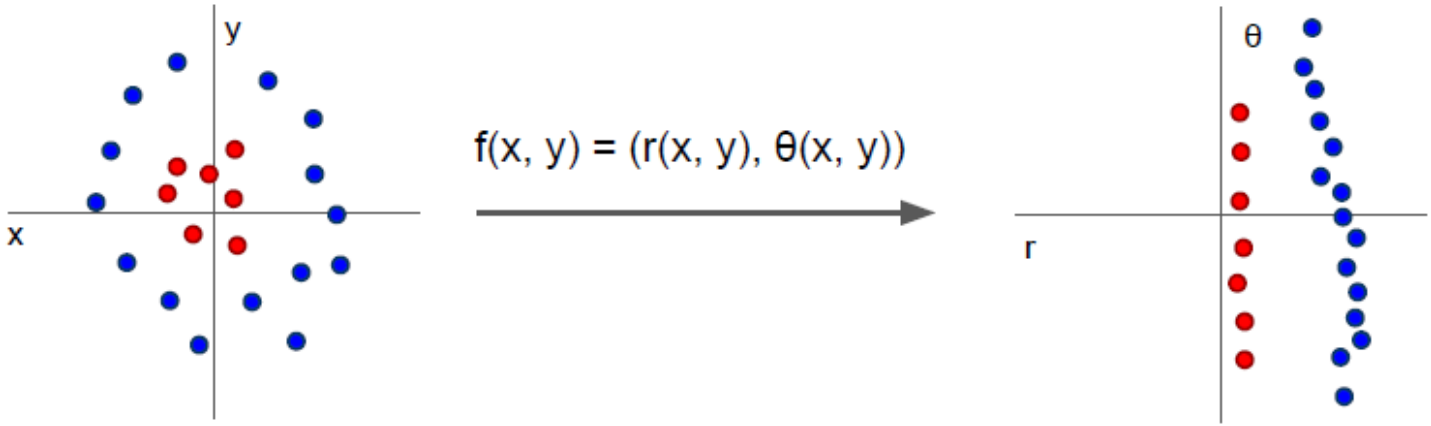


图像特征：动机



线性分类器
无法区分红
点和蓝点

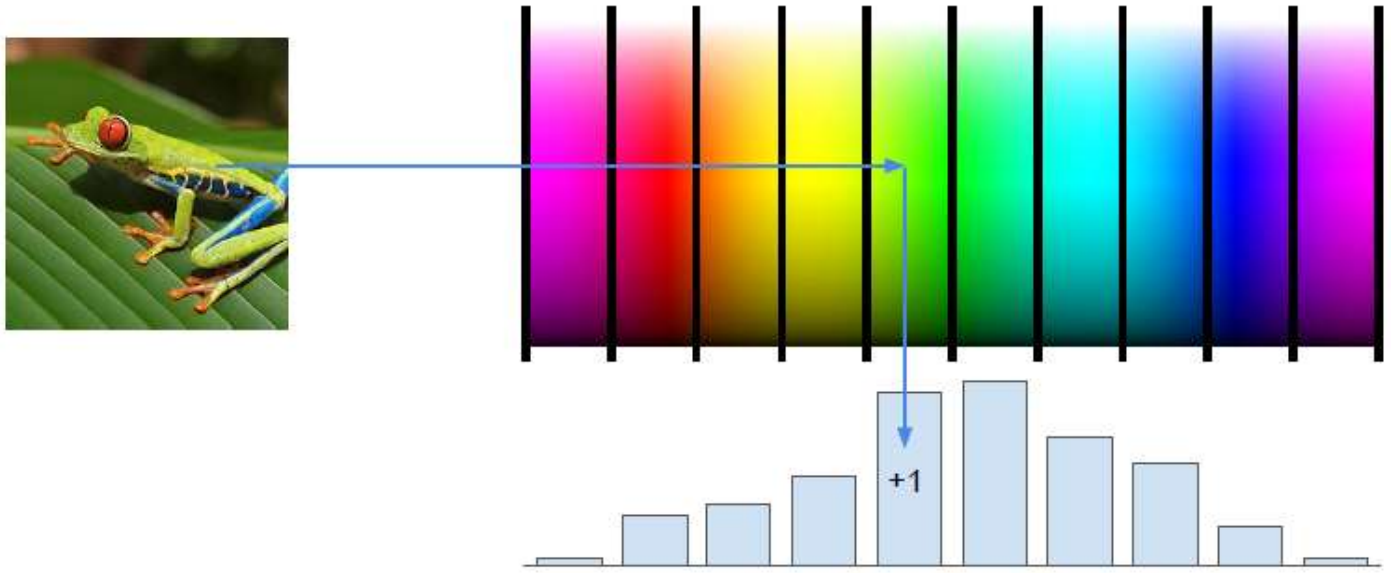
图像特征：动机



线性分类器无法区分红点和蓝点

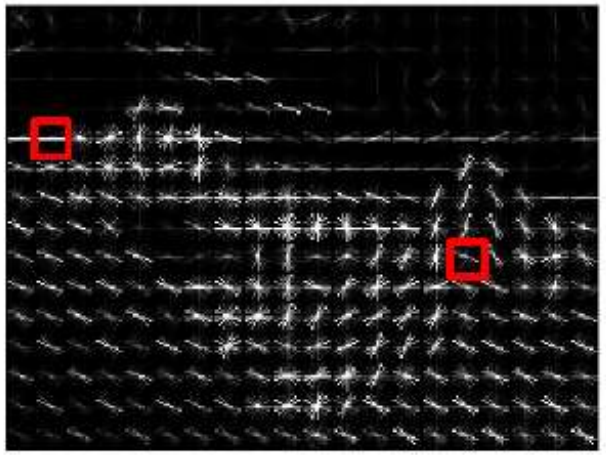
特征变换后可以用线性分类器分离

例子：颜色直方图（Color Histogram）



例子：方向梯度直方图（HoG）

将图像划分为8x8的像素区域，在每个区域内量化边缘方向到9组



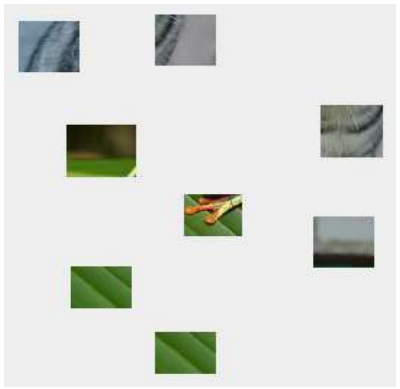
例：320x240的图像被划分为40x30份，每份由9个数字表示，因此特征向量长度为40*30*9=10800

Lowe, “Object recognition from local scale-invariant features”, ICCV 1999
Dalal and Triggs, "Histograms of oriented gradients for human detection," CVPR 2005

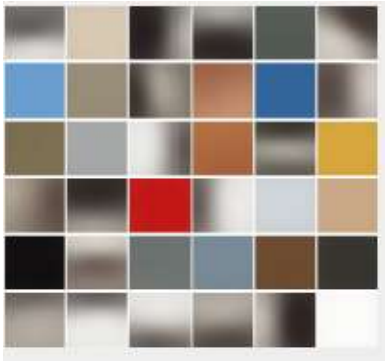
例子：词袋模型（Bag of Words）

步骤1：创建码本

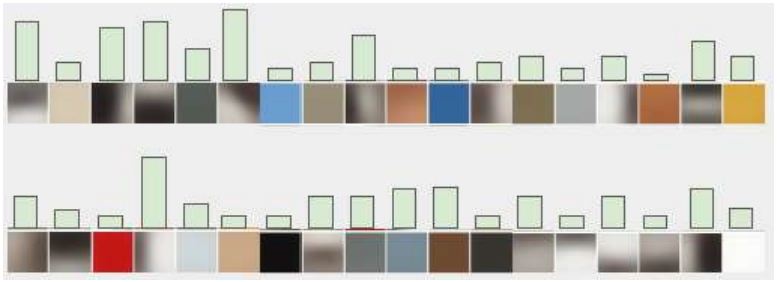
提取随机块



将随机块聚类形成“视觉词”的码本

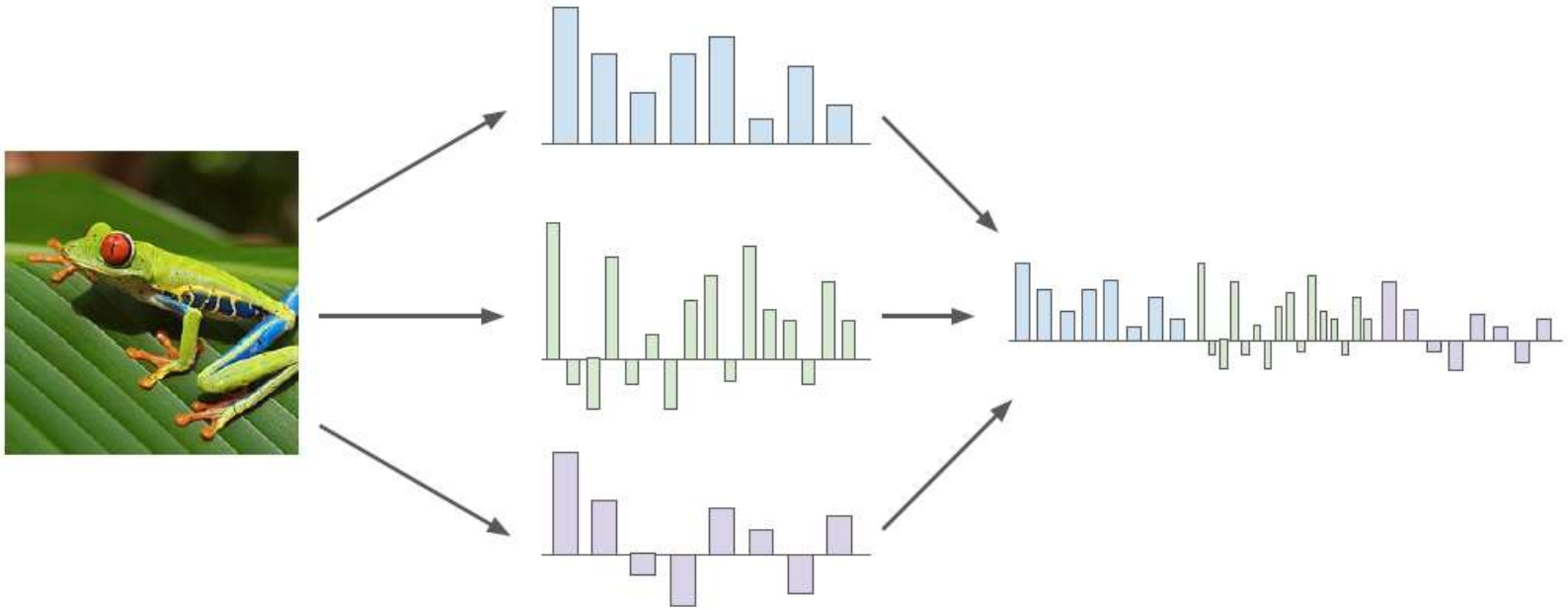


步骤2：图像编码



Fei-Fei and Perona, “A bayesian hierarchical model for learning natural scene categories”, CVPR 2005

图像特征



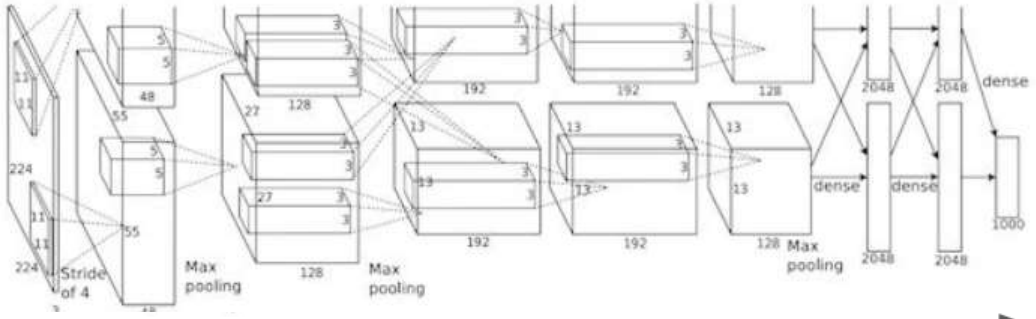
图像特征 vs 卷积网络

特征提取

f

分类得分
（长度为10的向量）

训练

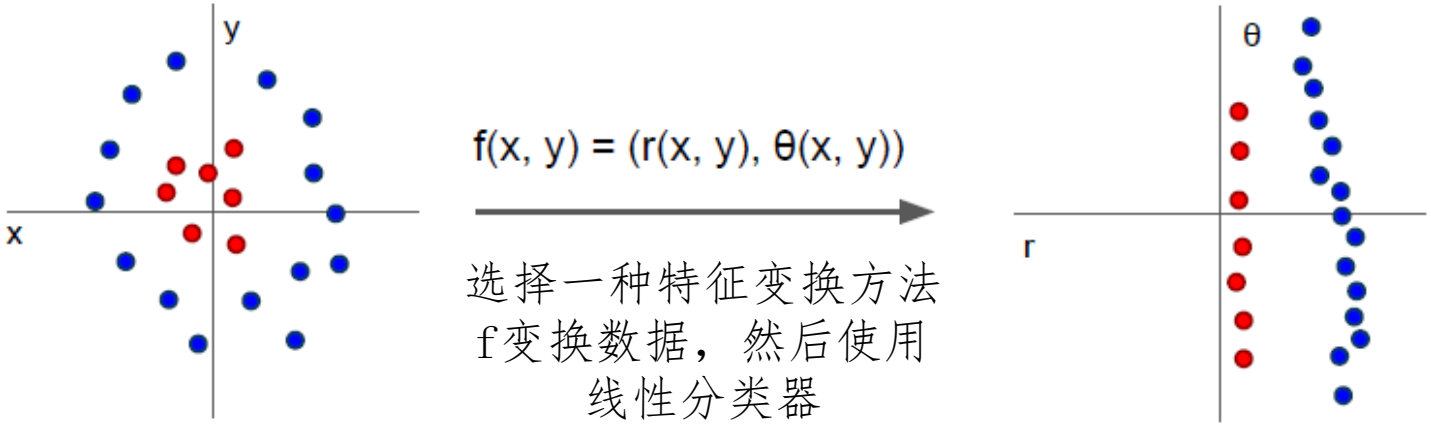


分类得分
（长度为10的向量）

训练

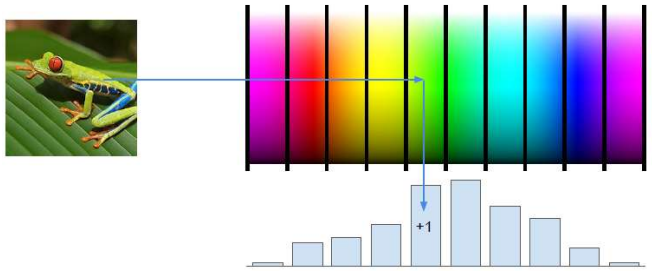
Krizhevsky, Sutskever, and Hinton, “Imagenet classification with deep convolutional neural networks”, NIPS 2012.
Figure copyright Krizhevsky, Sutskever, and Hinton, 2012. Reproduced with permission.

一种解决方案：特征变换

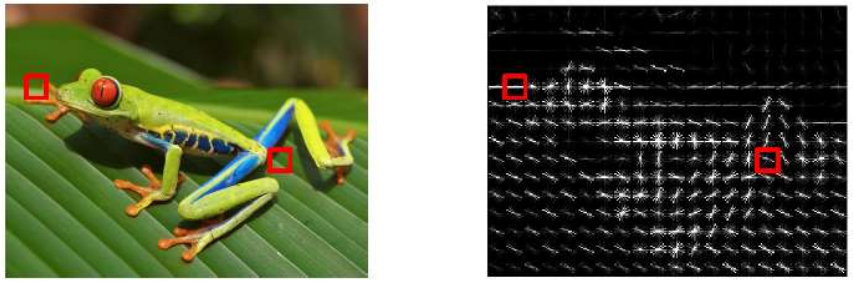


选择一种特征变换方法f变换数据，然后使用线性分类器

颜色直方图



方向梯度直方图



神经网络

神经网络

（**之前**）线性分类函数：$f = Wx$

神经网络

（**之前**）线性分类函数：$f = Wx$

（**现在**）2层神经网络：$f = W_2 \max(0, W_1 x)$

$$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$$

（在实践中，通常也会在每一层添加一个可学习的偏差）

神经网络

（**之前**）线性分类函数：$f = Wx$

（**现在**）2层神经网络：$f = W_2 \max(0, W_1 x)$

$$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$$

“神经网络”是一个非常宽泛的术语，更精确的可以叫做“**全连接网络**”，　有时也称作“**多层感知机**”(MLP)。

（在实践中，通常也会在每一层添加一个可学习的偏差）

神经网络

（**之前**）线性分类函数： $f = Wx$

（**现在**）2层神经网络： $f = W_2 \max(0, W_1 x)$

或 3层神经网络： $f = W_3 \max(0, W_2 \max(0, W_1 x))$

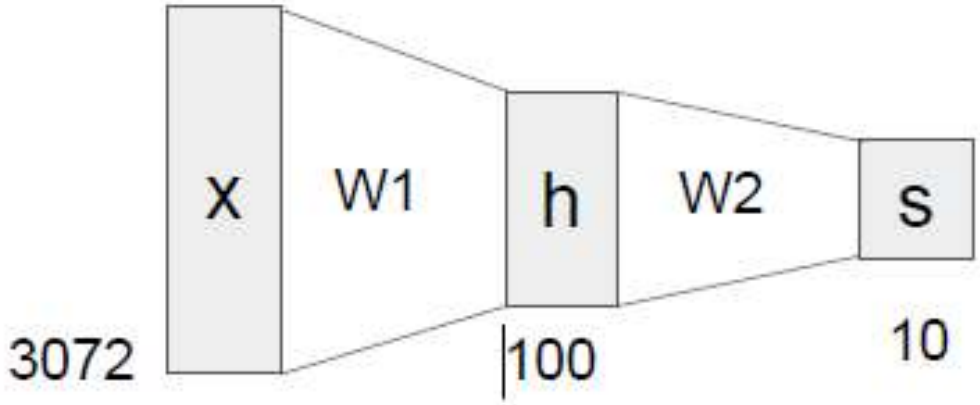
$$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$$

（在实践中，通常也会在每一层添加一个可学习的偏差）

神经网络

（之前）线性分类函数：$f = Wx$

（现在）2层神经网络：$f = W_2 \max(0, W_1 x)$

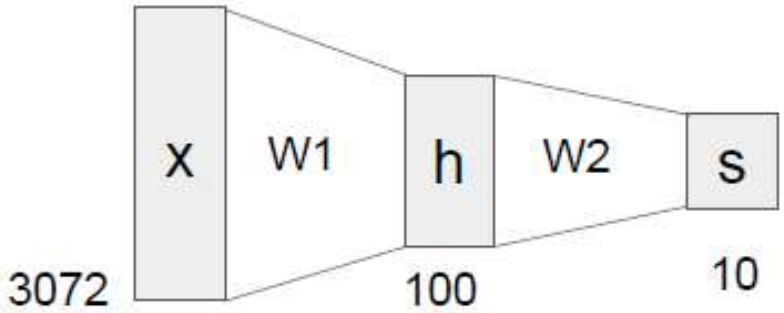


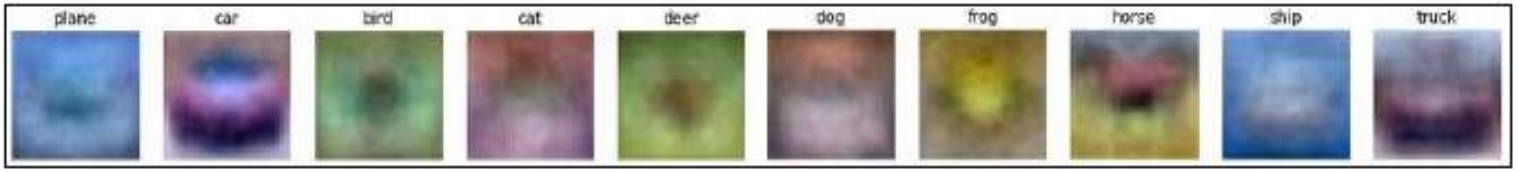
$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$

神经网络

（之前）线性分类函数：$f = Wx$

（现在）2层神经网络：$f = W_2 \max(0, W_1 x)$





学习100个模板而不是10个　　类之间共享模板

神经网络

（之前）线性分类函数：$f = Wx$

（现在）2层神经网络：$f = W_2 \max(0, W_1 x)$

函数 max (0, z) 叫做激活函数

问题：如果构造一个没有激活函数的神经网络会怎么样？

$$f = W_2 W_1 x$$

神经网络

（之前）线性分类函数：$f = Wx$

（现在）2层神经网络：$f = W_2 \max(0, W_1 x)$

函数 max (0, z) 叫做激活函数

问题：如果构造一个没有激活函数的神经网络会怎么样？

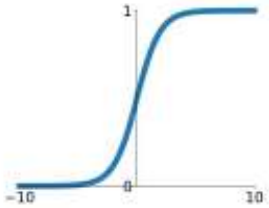
$$f = W_2 W_1 x \quad W_3 = W_2 W_1 \in \mathbb{R}^{C \times H}, f = W_3 x$$

答：最终会再次得到一个线性分类器！

激活函数

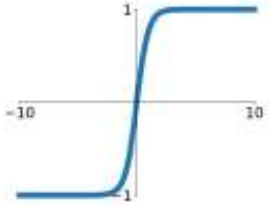
Sigmoid

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



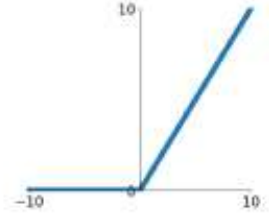
tanh

$$\tanh(x)$$



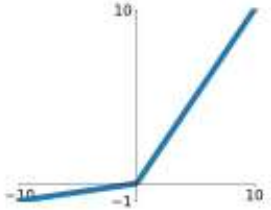
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

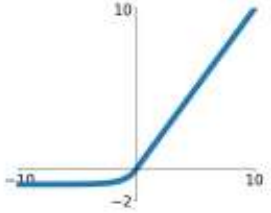


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

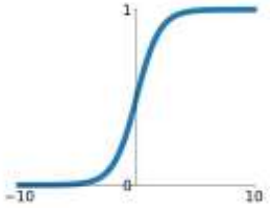
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



激活函数

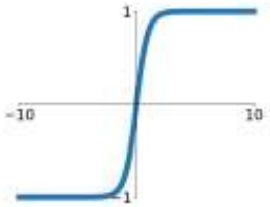
Sigmoid

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



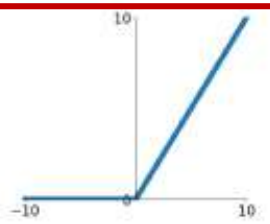
tanh

$\tanh(x)$



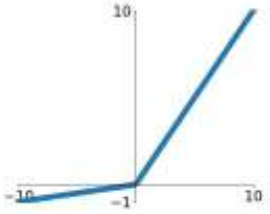
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

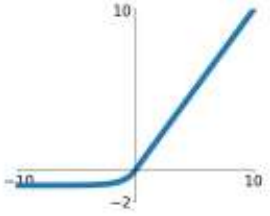


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

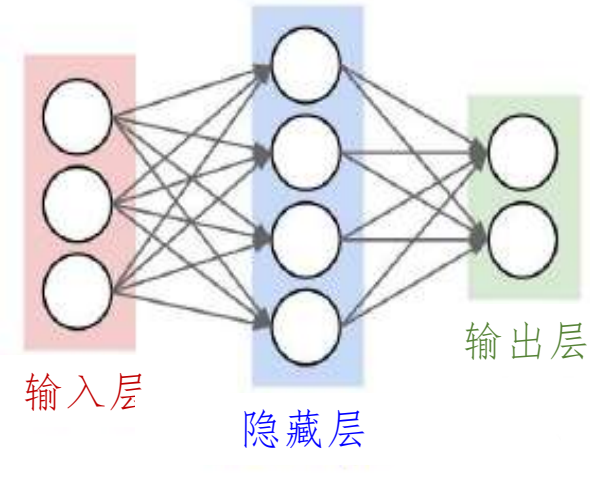
ELU

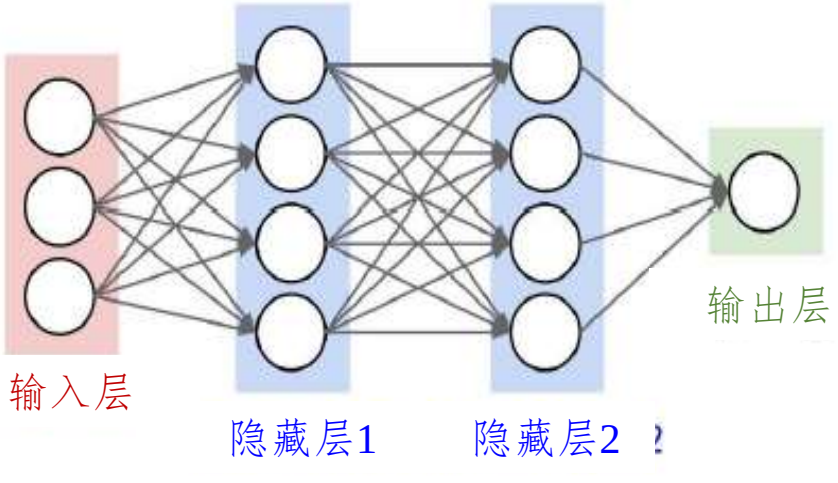
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



对大多数问题来说，ReLU是一个不错的选择

神经网络：结构

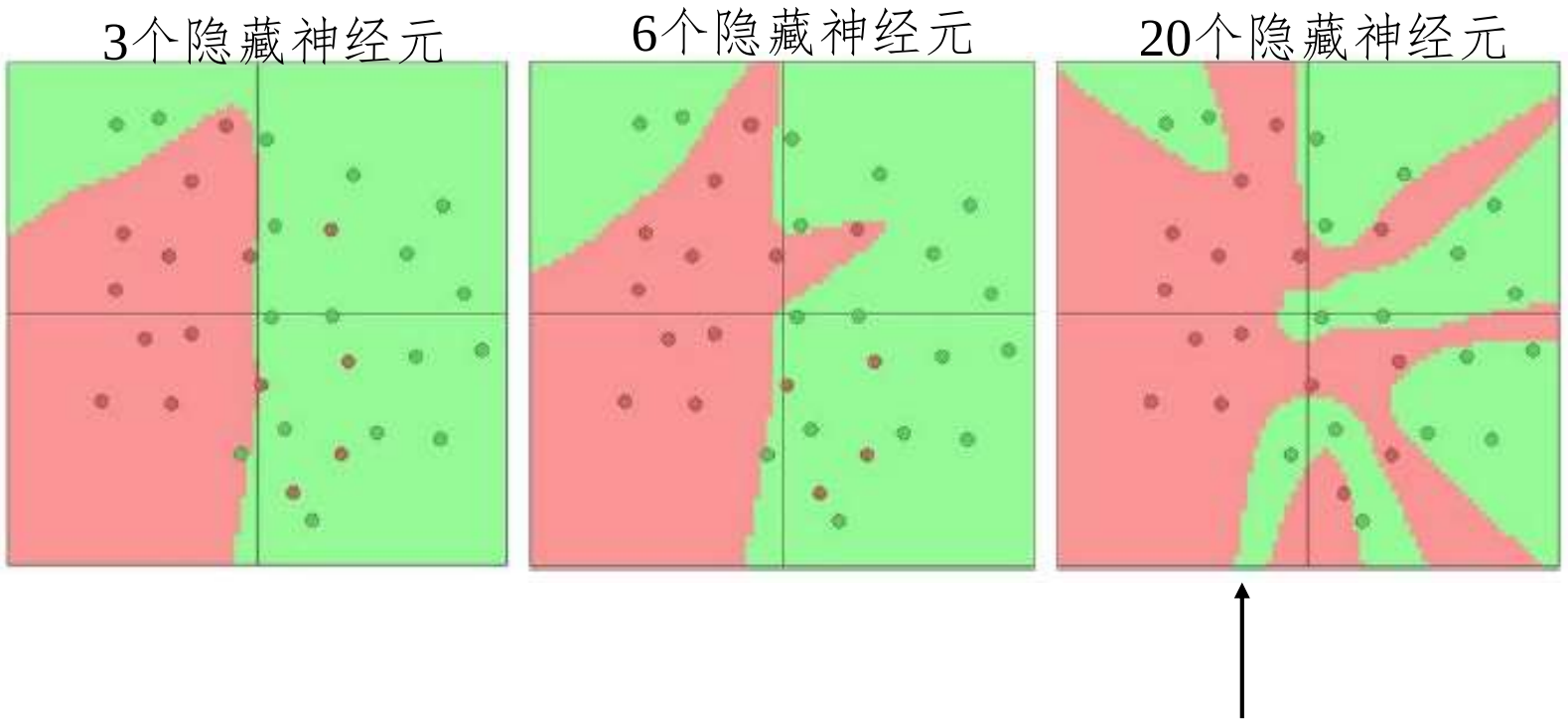




“两层神经网络”或者
“含有一个隐藏层的神经网络”

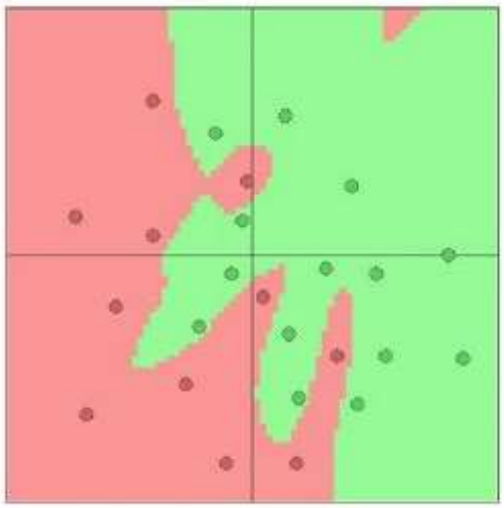
“三层神经网络”或者
“含有两个隐藏层的神经网络”

网络层数量和大小的设置

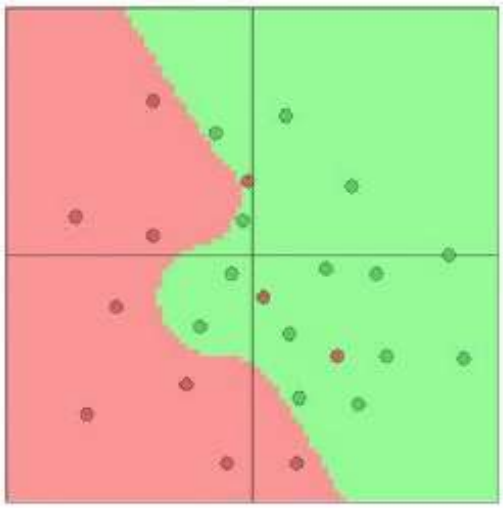


不使用网络的大小作为正则化器，使用更强的正则化方法代替

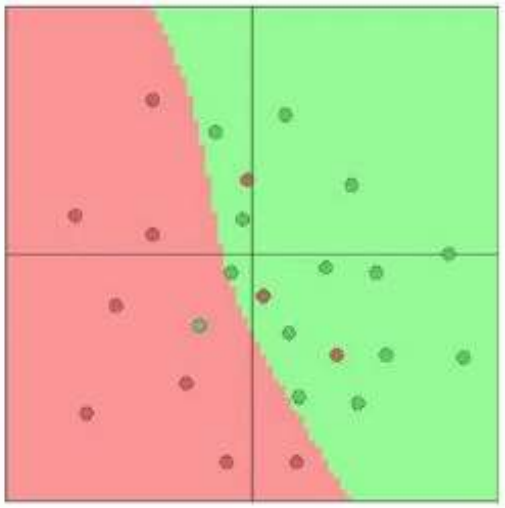
λ = 0.001



λ = 0.01



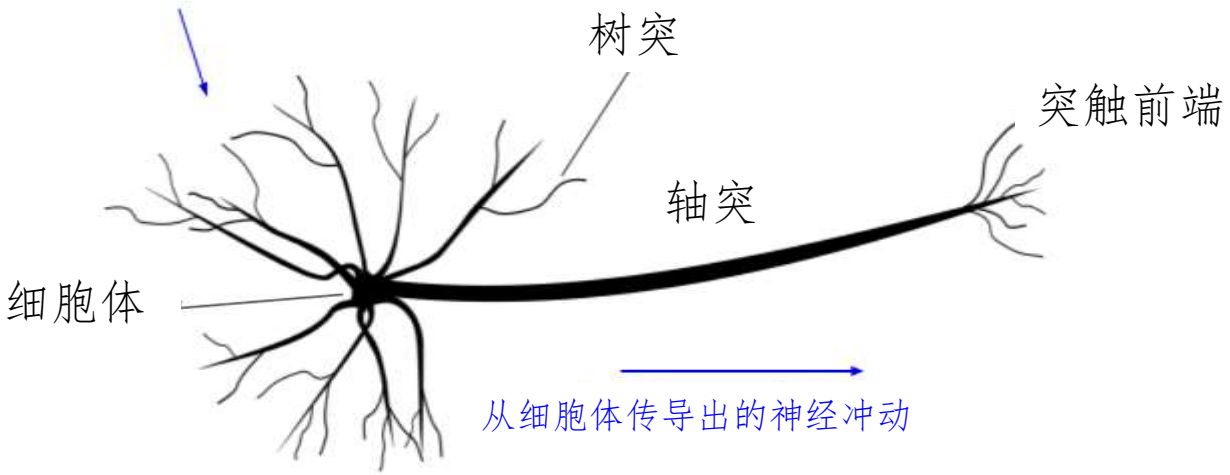
λ = 0.1

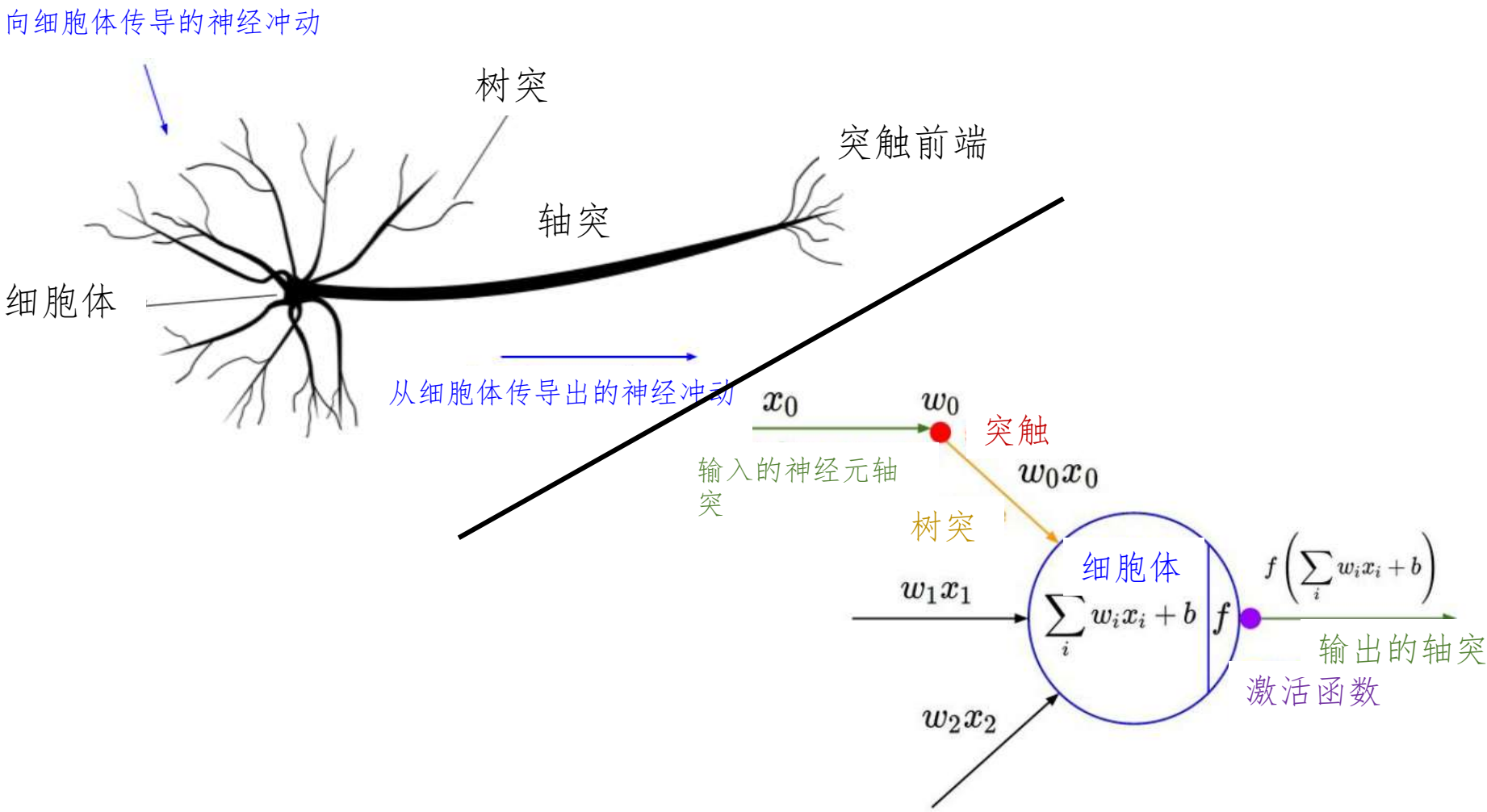


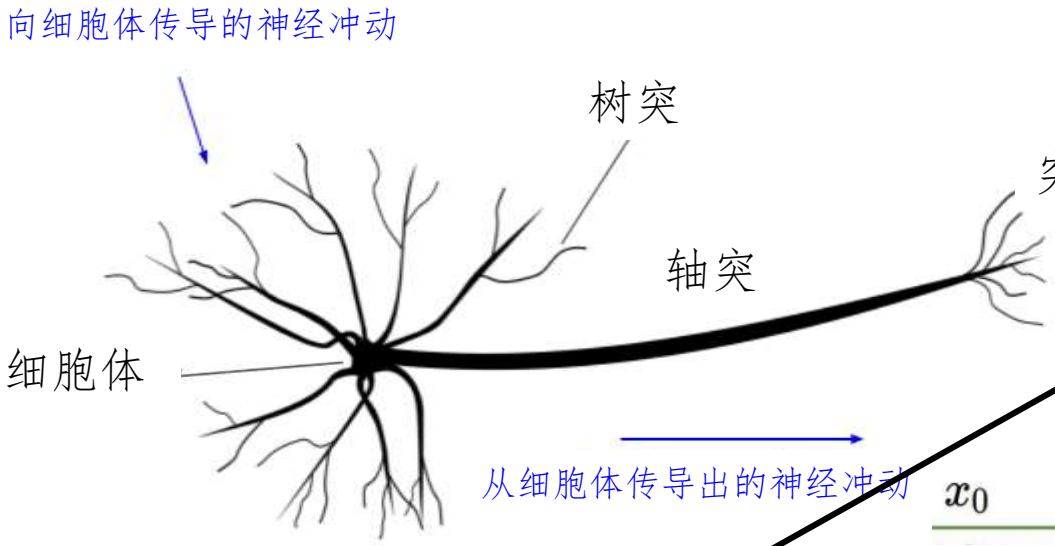
$$L(W) = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

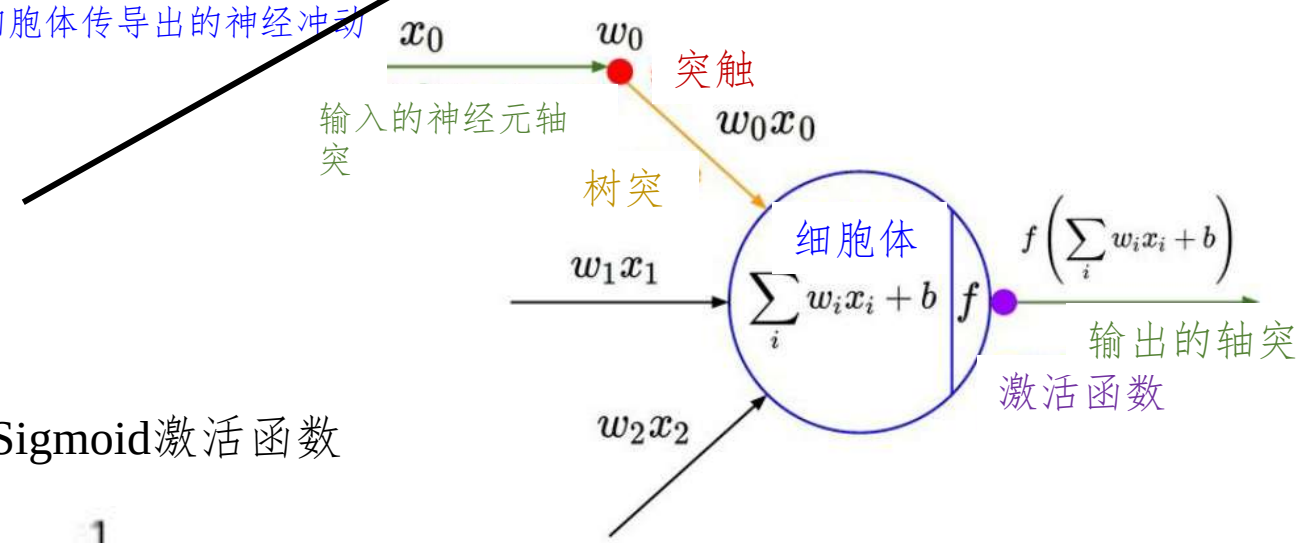
ConvNetJS 网络演示地址:
http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

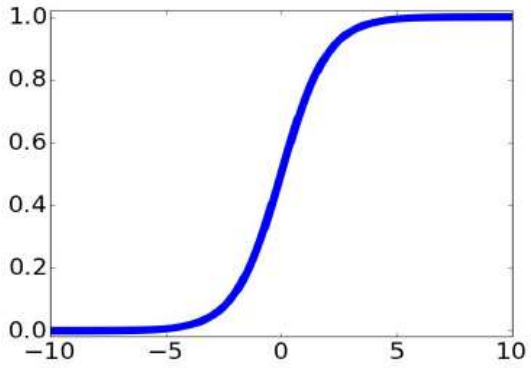
向细胞体传导的神经冲动




向细胞体传导的神经冲动
树突
突触前端
轴突
细胞体
从细胞体传导出的神经冲动
x_0
w_0
突触
输入的神经元轴突
w_0x_0
树突
细胞体
$\sum_i w_i x_i + b$
f
$f\left(\sum_i w_i x_i + b\right)$
输出的轴突
激活函数
w_1x_1
w_2x_2



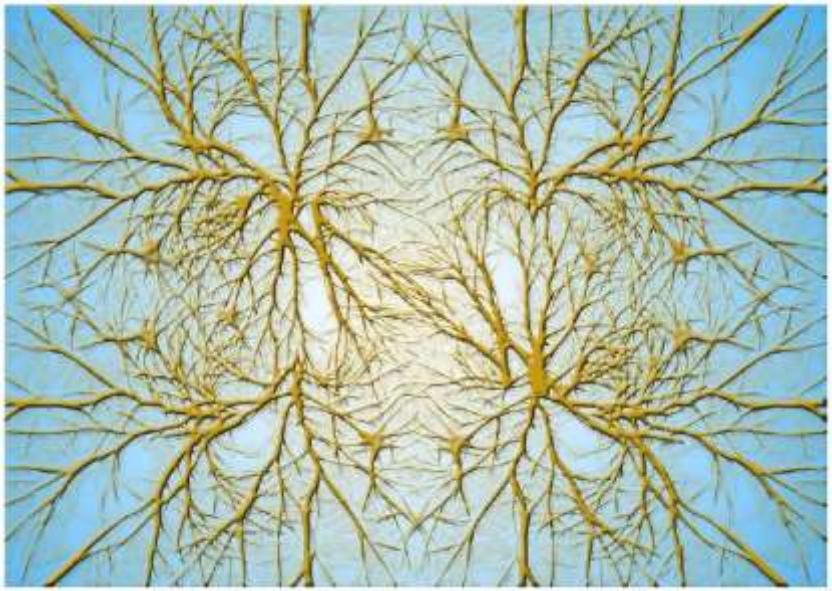




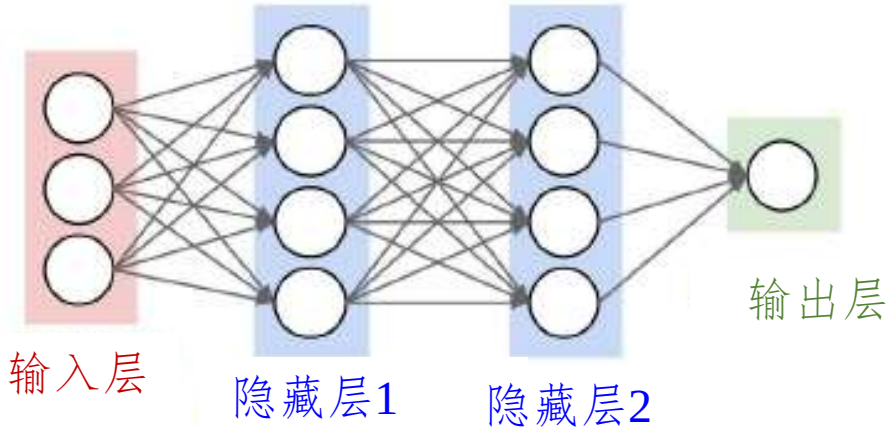
Sigmoid激活函数

$$\frac{1}{1+e^{-x}}$$

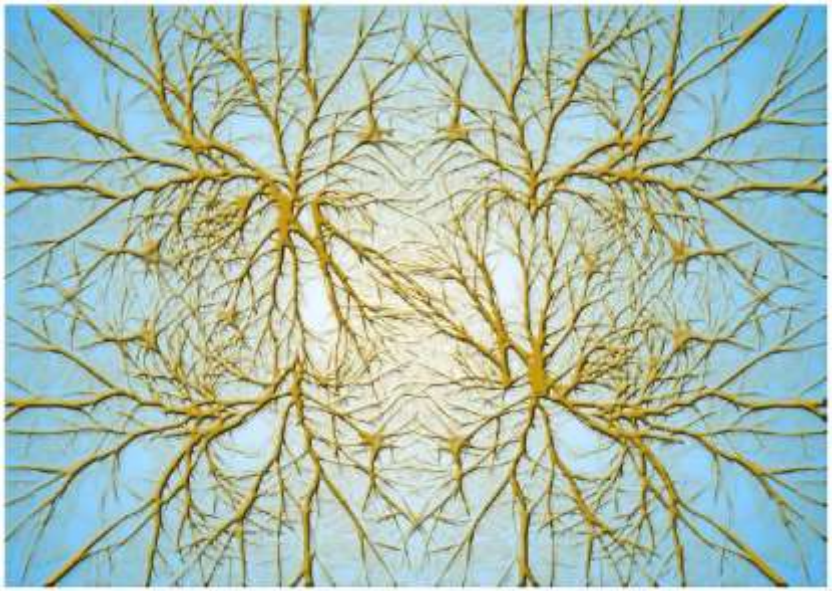
生物神经元：
复杂的连接模式



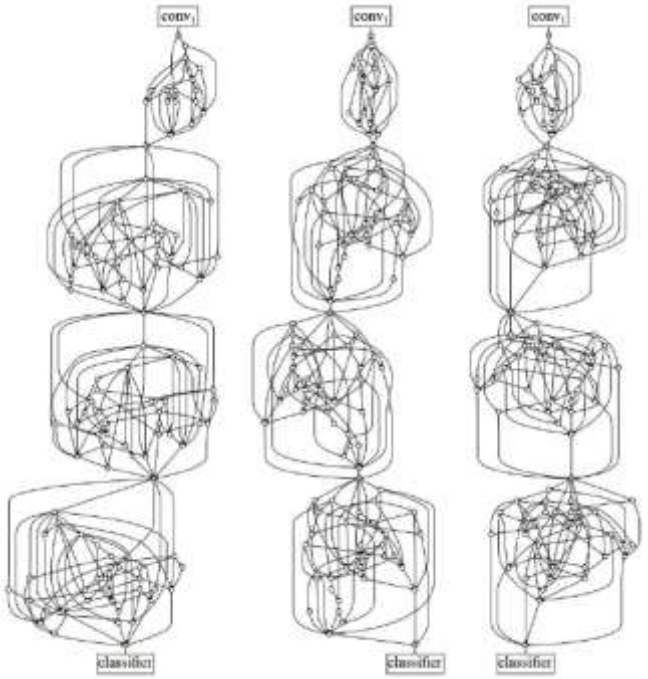
神经网络中的神经元：
以**规则**的形式构成以
提高计算效率



生物神经元：
复杂的连接模式



但是**随机连接**的神经网络同样可以工作！



Xie et al, “Exploring Randomly Wired Neural Networks for Image Recognition”, arXiv 2019

要特别注意对大脑的类比！

生物神经元：

- 许多不同的类型
- 树突可以进行复杂的非线性计算
- 突触不是一个单一的量，而是一个复杂的非线性动力系统

问题：怎样计算梯度？

$$s = f(x; W_1, W_2) = W_2 \max(0, W_1 x)$$ 非线性函数

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$ SVM 预测损失

$$R(W) = \sum_k W_k^2$$ 正则化

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i + \lambda R(W_1) + \lambda R(W_2)$$ 整体损失：数据损失 + 正则化

如果可以计算 $\frac{\partial L}{\partial W_1}, \frac{\partial L}{\partial W_2}$ 就能学习W_1和W_2

不好的做法：在纸上推导 $\nabla_W L$

$$s = f(x; W) = Wx$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

$$= \sum_{j \neq y_i} \max(0, W_{j,:} \cdot x + W_{y_i,:} \cdot x + 1)$$

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i + \lambda \sum_k W_k^2$$

$$= \frac{1}{N} \sum_{i=1}^{N} \sum_{j \neq y_i} \max(0, W_{j,:} \cdot x + W_{y_i,:} \cdot x + 1) + \lambda \sum_k W_k^2$$

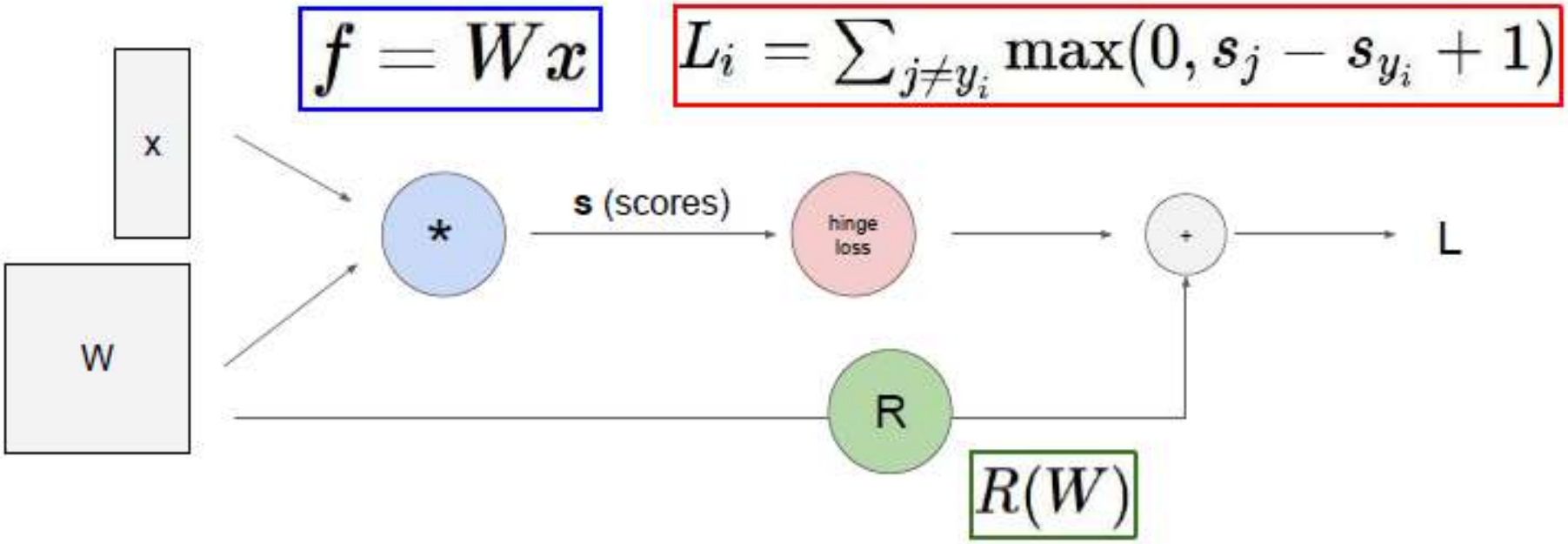
$$\nabla_W L = \nabla_W \left(\frac{1}{N} \sum_{i=1}^{N} \sum_{j \neq y_i} \max(0, W_{j,:} \cdot x + W_{y_i,:} \cdot x + 1) + \lambda \sum_k W_k^2 \right)$$

问题：非常麻烦，许多矩阵计算

问题：如果想改变损失函数怎么办？例如使用softmax而不是SVM，需要从头开始推导

问题：对非常复杂的模型来说是不可行的！

更好的做法：计算图 + 反向传播



卷积神经网络（AlexNet）

输入图像

权重

损失