机器学习简介

武雷，南京师范大学（基础理论）

任杰，北京理工大学（实战演习）

Outline

- 初识神经网络：回归和分类问题、神经网络原理
- 神经网络进阶：全链接、激活函数、输出层设计、误差计算
- 常见神经网络：BDT、CNN、RNN、GAN、Auto-Encoder
- 新物理研究中的机器学习：模型分析、新信号寻找
- 实战演习：scikit-learn、pytorch、tensorflow

数学基础：微积分、线性代数、统计

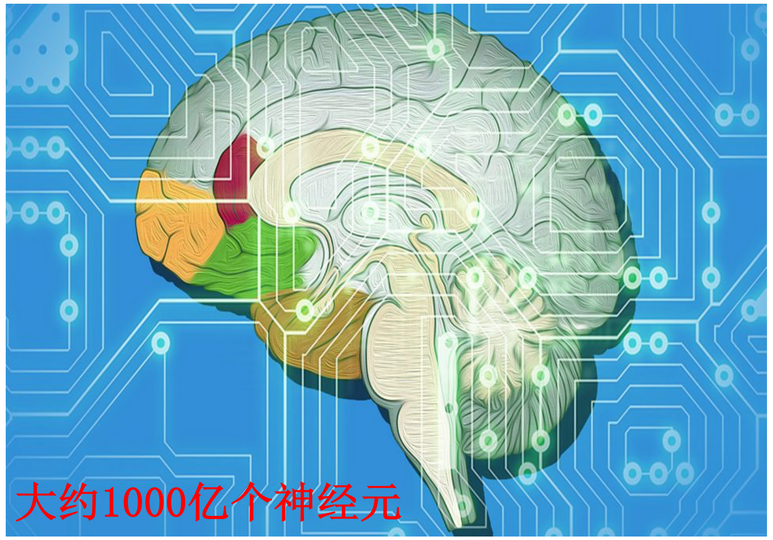
1 初识神经网络

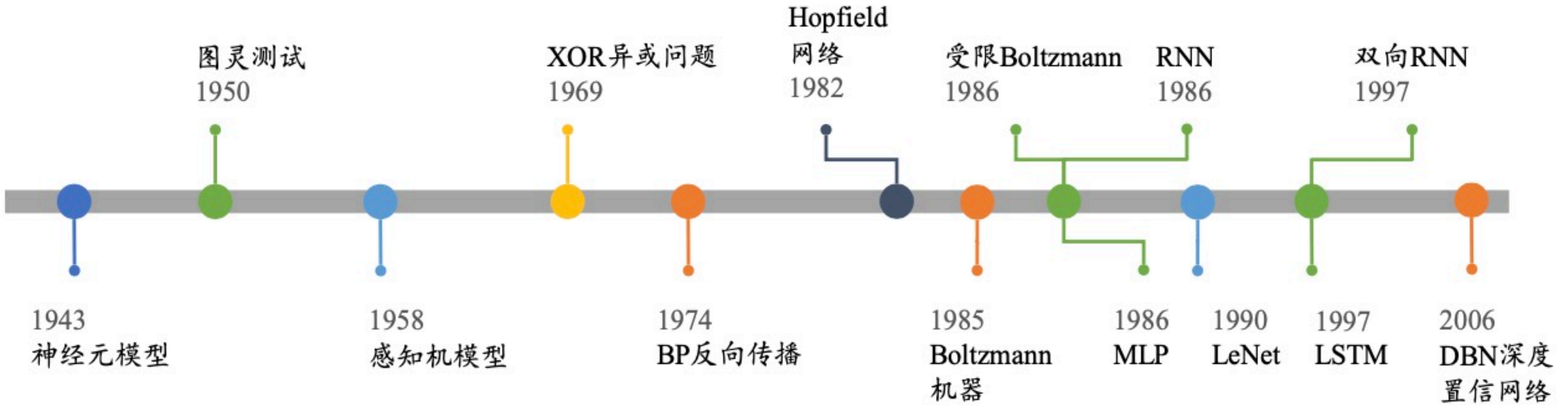
大量基本运算，简单

图像识别，困难

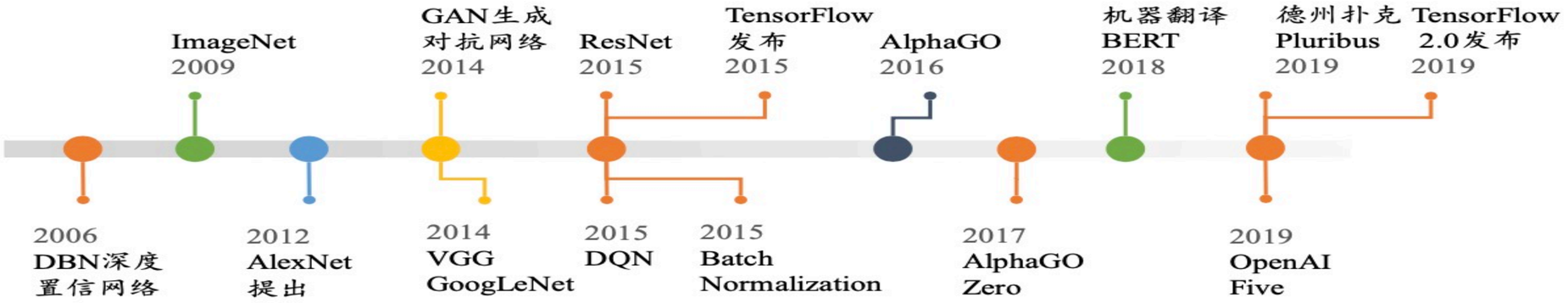
1 初识神经网络

1956召开的达特茅斯会议



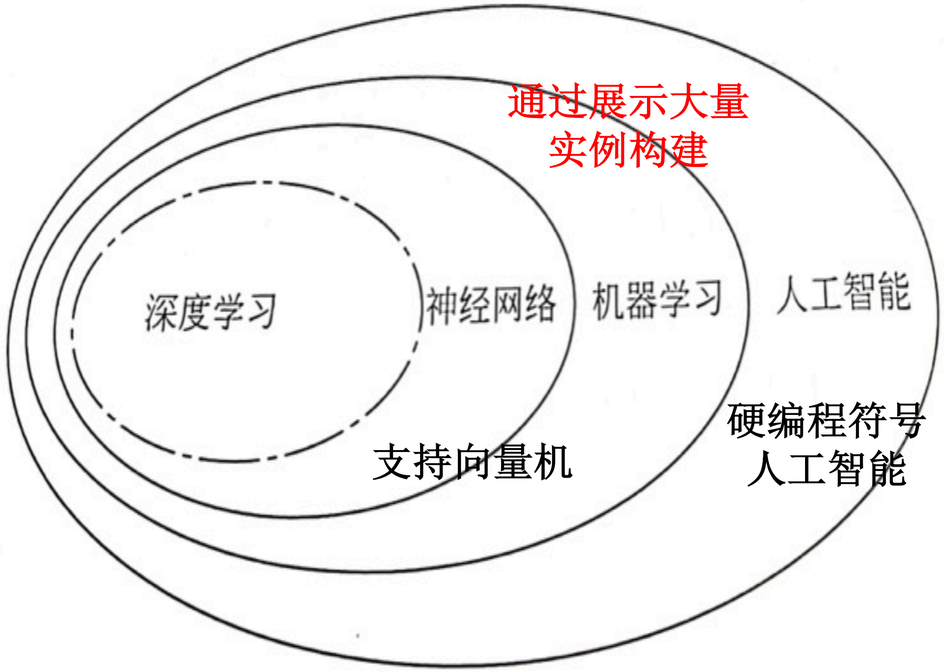


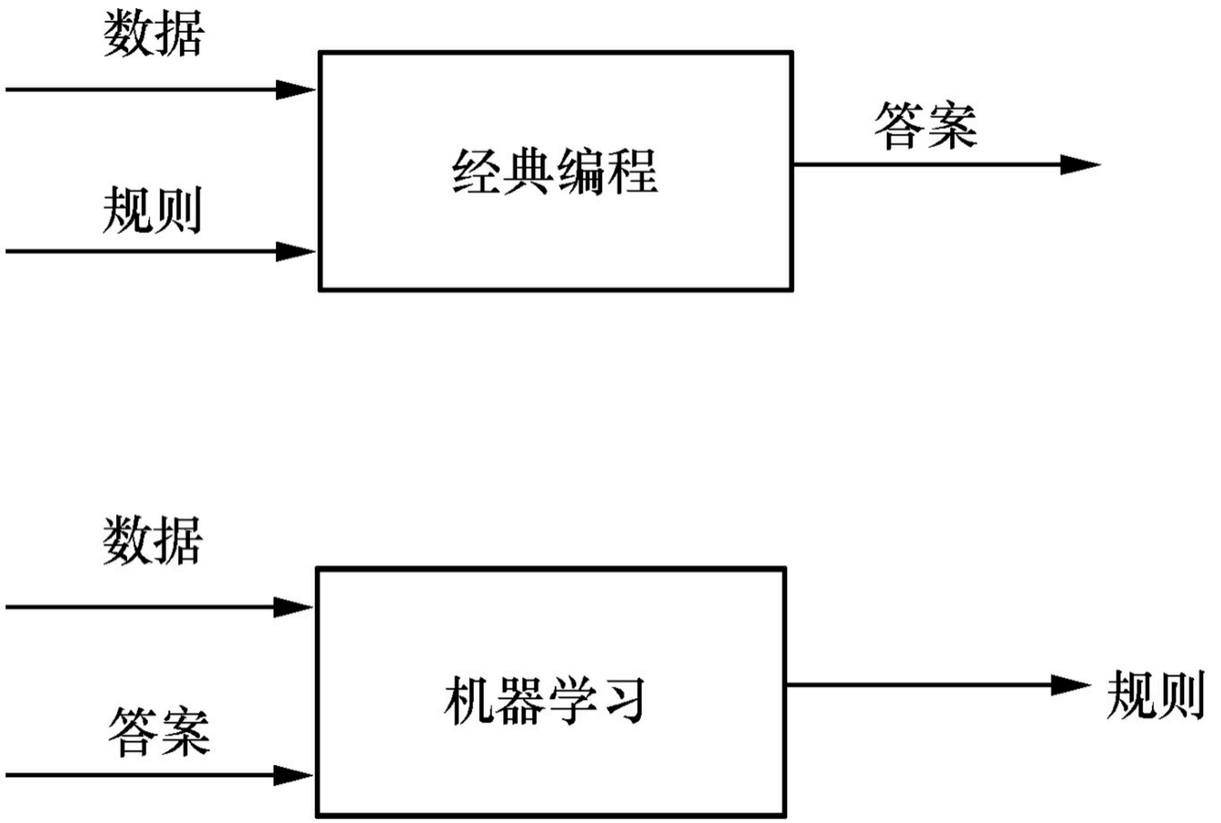
浅层神经网络发展时间线



深度学习发展时间线

1 初识神经网络


通过展示大量
实例构建
深度学习
神经网络
机器学习
人工智能
支持向量机
硬编程符号
人工智能


数据
规则
经典编程
答案
数据
答案
机器学习
规则

1 初识神经网络

《终结者》

暂时问题不大：非常受控的环境，给出有限的决策边界

1 初识神经网络

有监督学习：有标签（分类、回归等）

无监督学习：无标签（聚类、降维等）

半监督学习：带标签+无标签

强化学习：与环境交互

1 初识神经网络

回归问题（regression）：

建模和分析变量之间的关系，多预测问题

1 初识神经网络

预测器： 机器接受了一个输入，并做出应有的预测，输出结果

人类

计算机

1 初识神经网络

预测器：

不知道转换公式，仅知道两者成线性关系，以及一些正确的数值对示例

真实示例	千米	英里
1	0	0
2	100	62.137

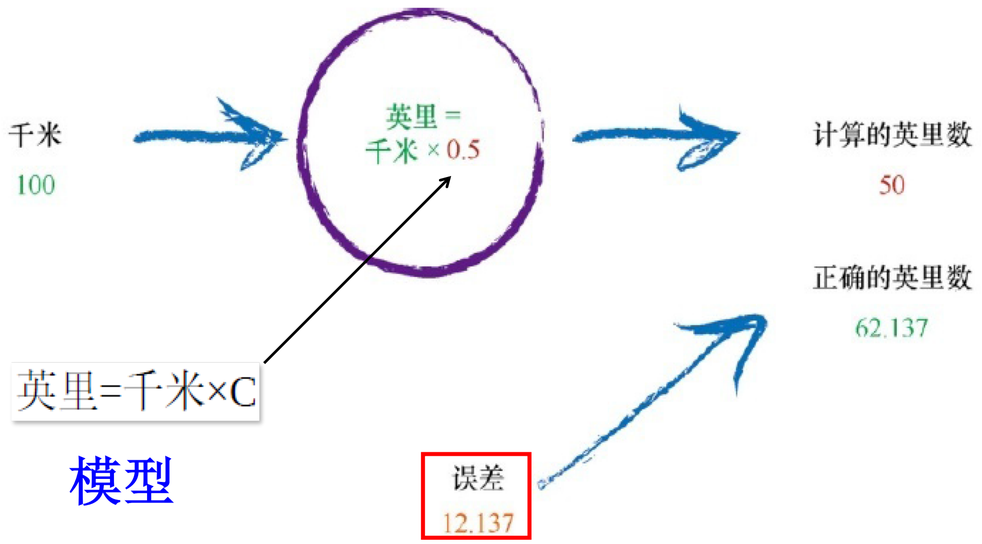
1 初识神经网络

预测器：

误差值=真实值-计算值

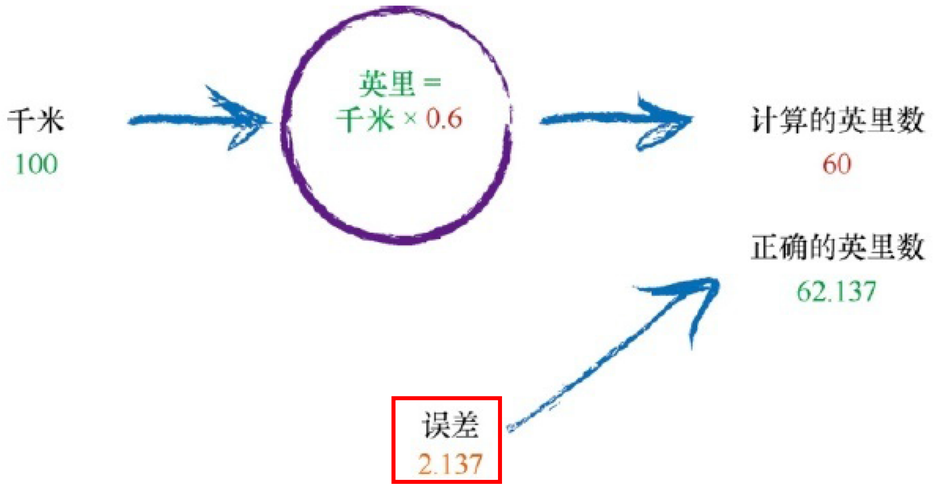
= 62.137-50

= 12.137



模型

持续细化误差值，并多次改进答案
（神经网络中学习的核心过程）



更多问题是没有一个简单的数学公式将输出和输入关联起来的。
这就是我们需要诸如神经网络这样相对成熟而复杂的方法的原因。

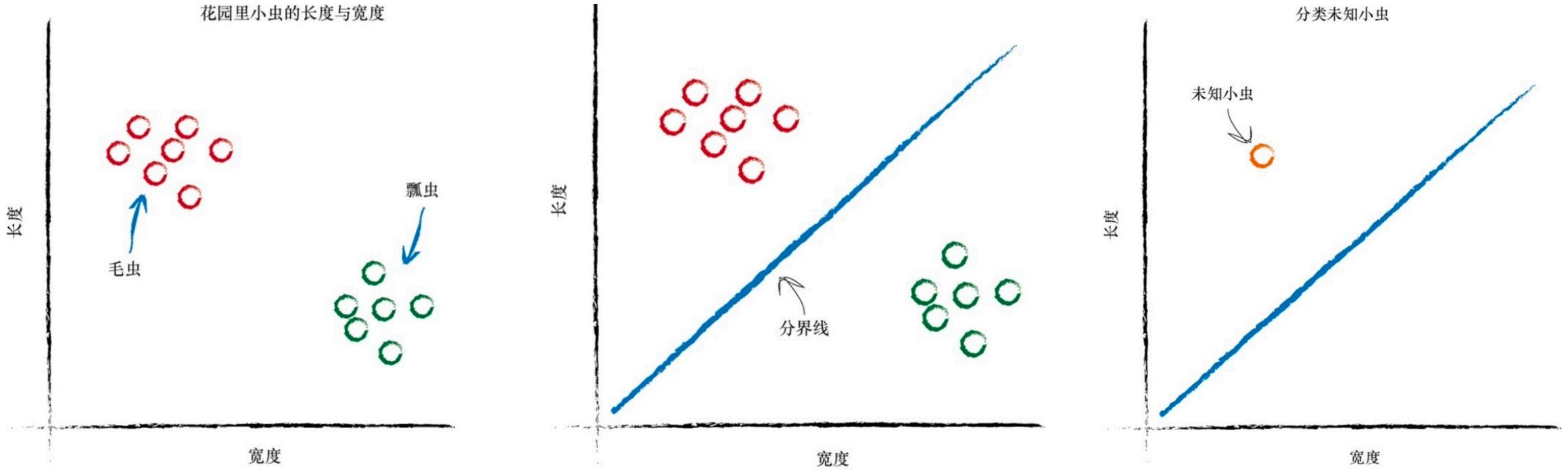
1 初识神经网络

分类问题（classification）：

即找一个函数判断输入数据所属的类别，可以是二类别问题（是/不是），也可以是多类别问题（在多个类别中判断输入数据具体属于哪一个类别）。与回归问题（regression）相比，分类问题的输出不再是连续值，而是离散值，用来指定其属于哪个类别。

1 初识神经网络

分类器：



如何得到正确的斜率呢?我们如何改进不能很好划分这两种小虫的分界线呢?

这个问题的答案处于神经网络学习的核心地带。

1 初识神经网络

分类器：

训练数据

实例	宽度	长度	小虫
1	3.0	1.0	瓢虫
2	1.0	3.0	毛虫

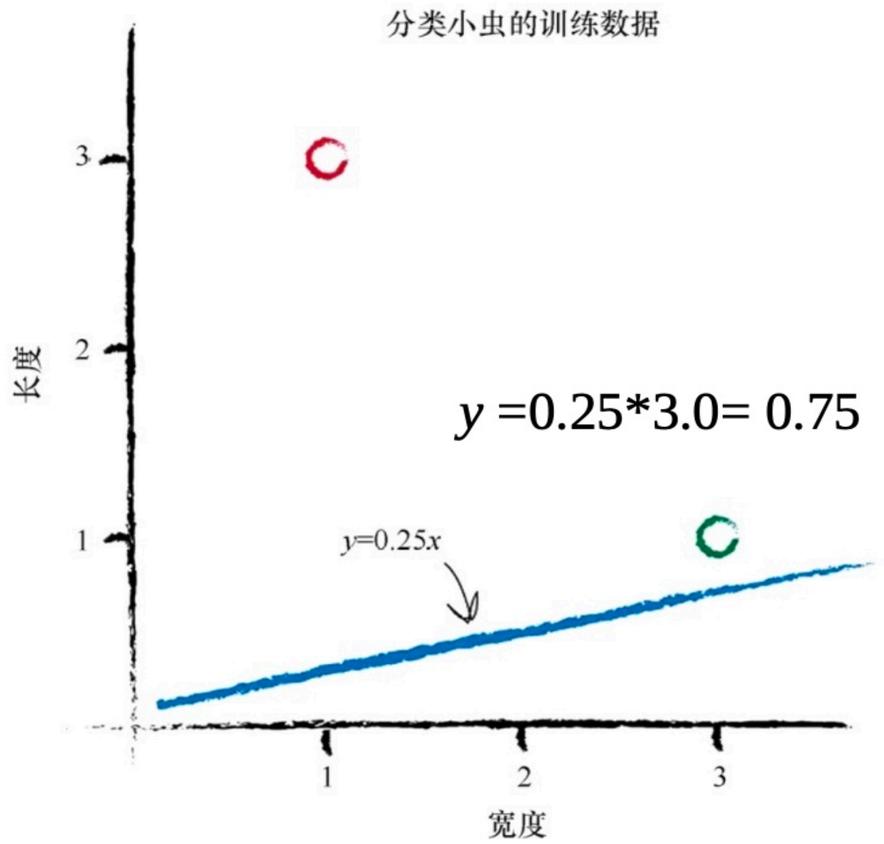
这些实例帮助我们调整分类函数的斜率。用来训练预测器或分类器的真实实例，我们称为训练数据。

1 初识神经网络

分类器：

模型

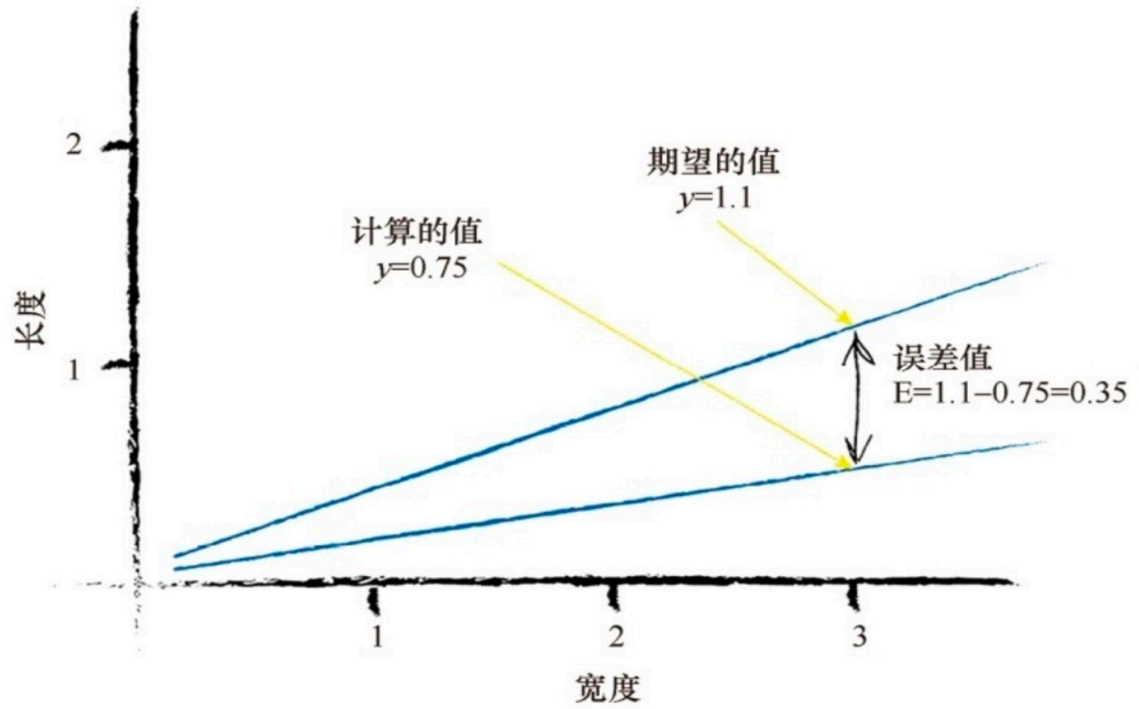
$y = Ax$



如何调整参数A?

1 初识神经网络

分类器：

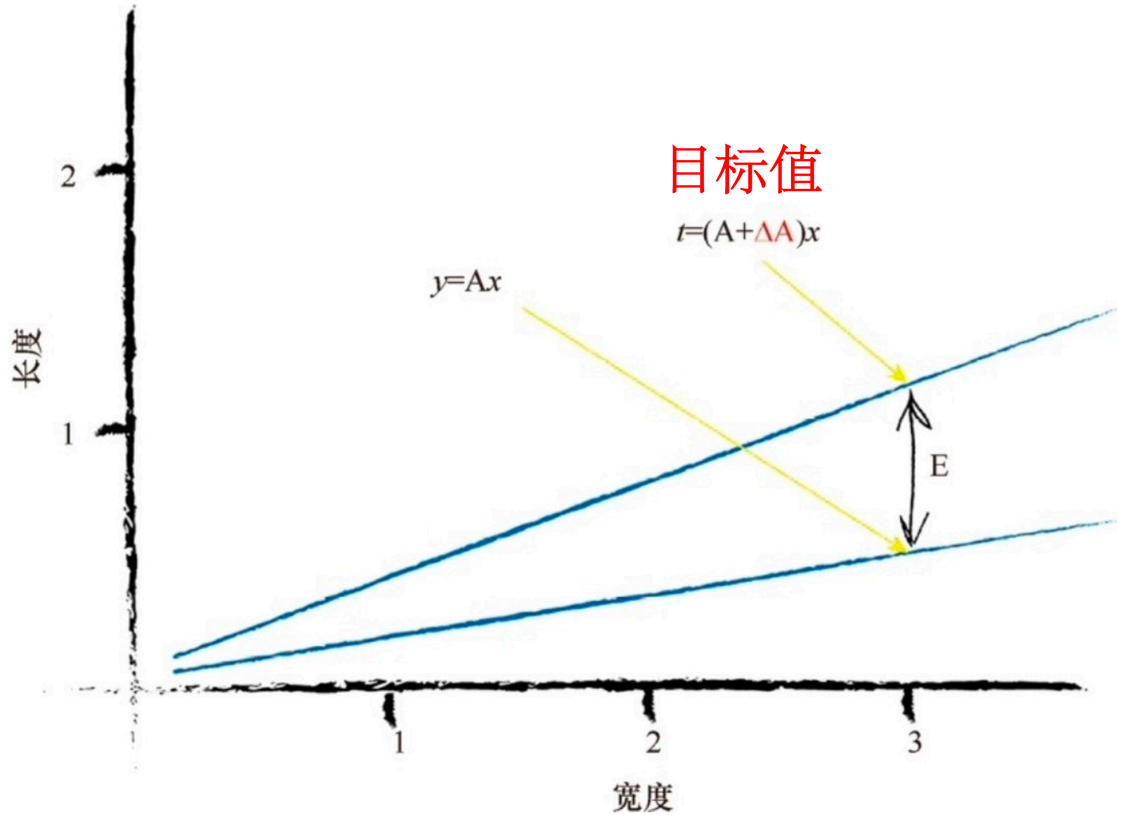


实例	宽度	长度	小虫
1	3.0	1.0	瓢虫
2	1.0	3.0	毛虫

E和A的关系？

1 初识神经网络

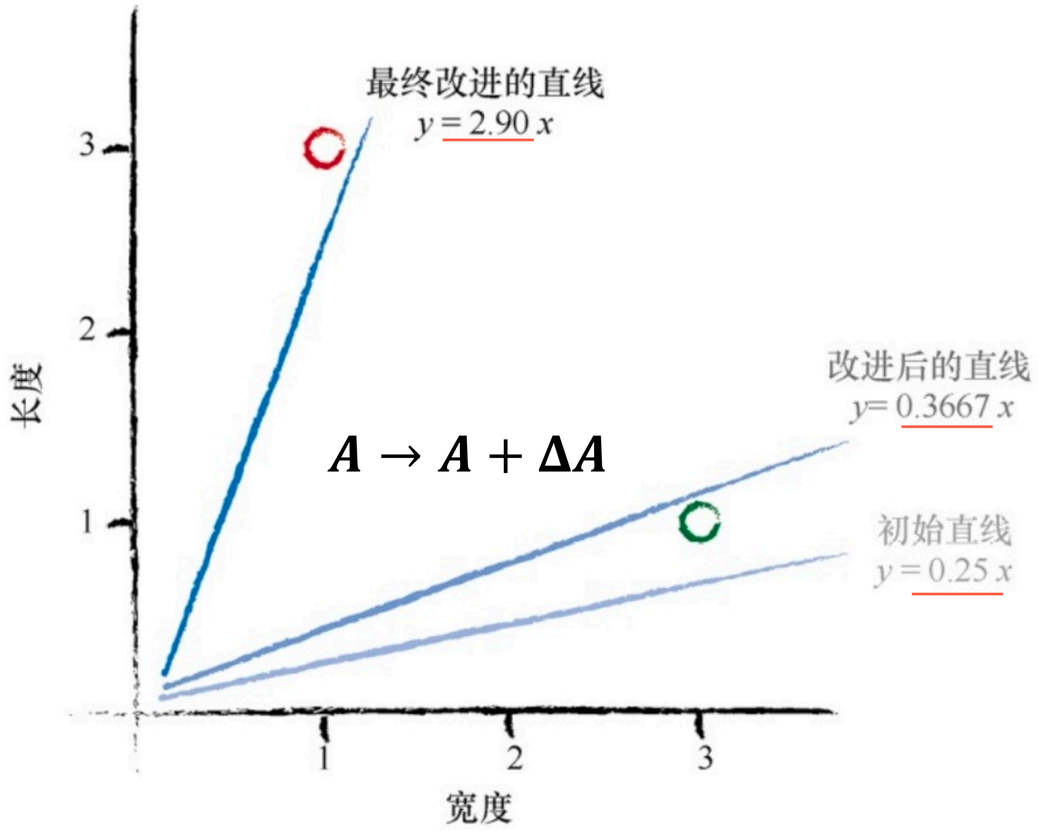
分类器：



$$\Delta A = E / x$$

1 初识神经网络

分类器：



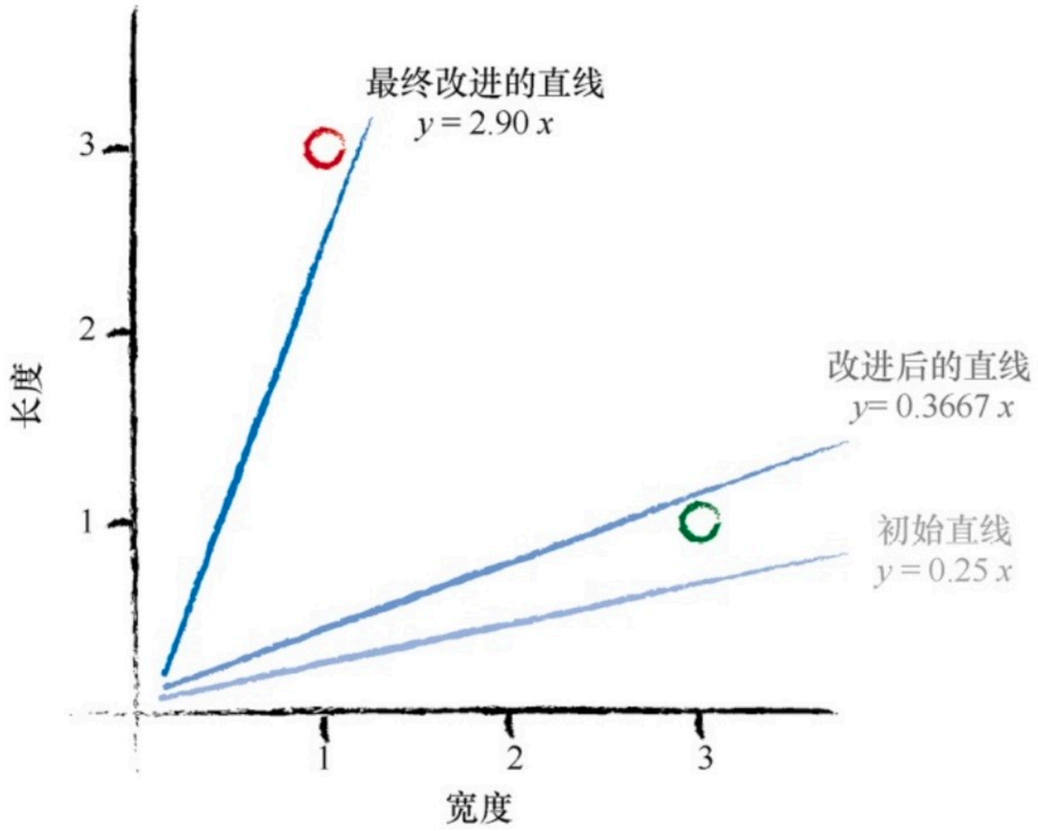
第一组目标值设置为1.1

第二组目标值设置为2.9

实例	宽度	长度	小虫
1	3.0	1.0	瓢虫
2	1.0	3.0	毛虫

1 初识神经网络

分类器：



问题来了：

改进的直线与最后一次训练样本非常匹配。

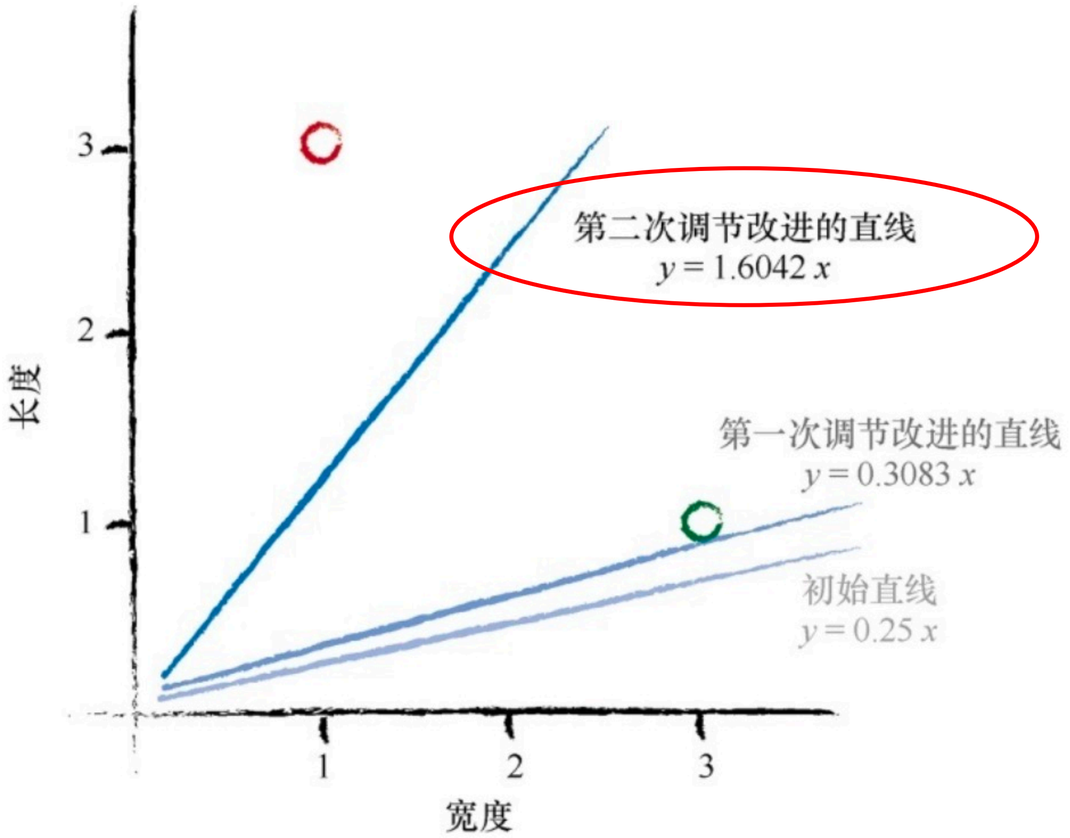
适度改进(moderate)

$$\Delta A = L(E / x)$$

learning rate

1 初识神经网络

分类器：

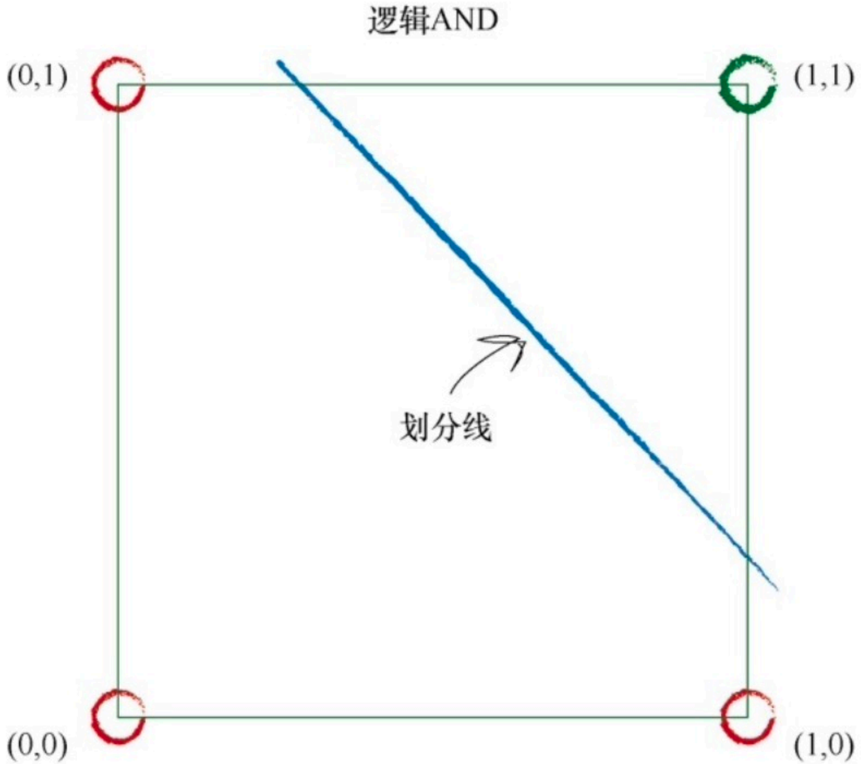


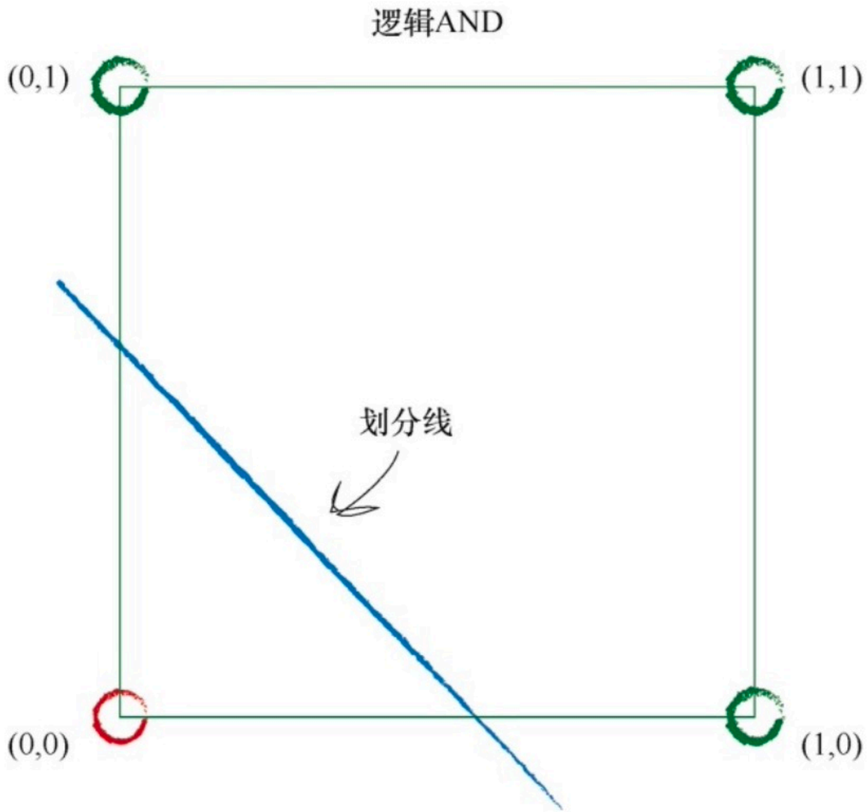
learning rate (L) = 0.5

1 初识神经网络

分类器：

线性分类器





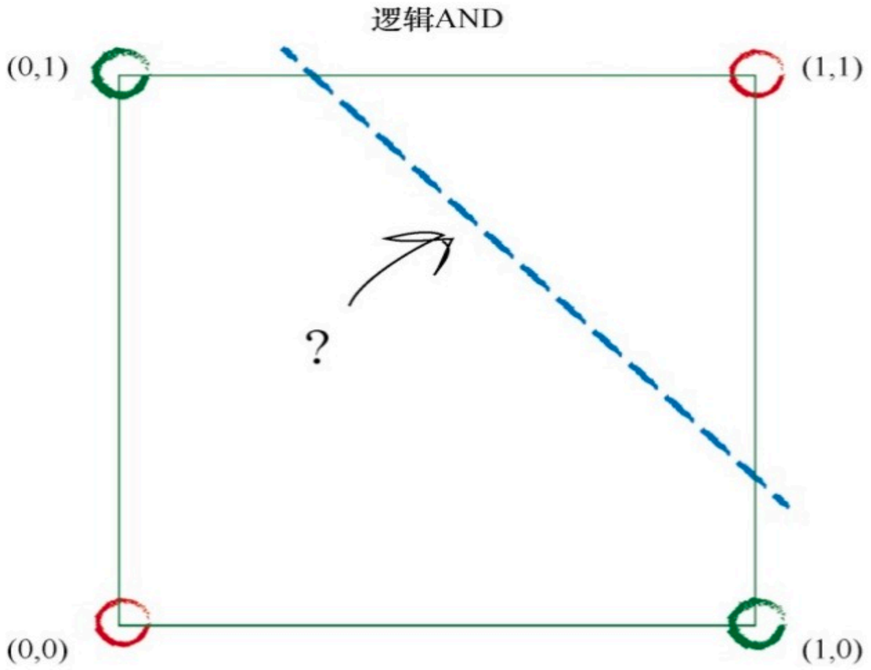
1 初识神经网络

分类器：

异或函数

输入A	输入B	逻辑XOR
0	0	0
0	1	1
1	0	1
1	1	0

只有在A或B仅有一个为真但两个输入不同时为真的情况下，才输出为真。

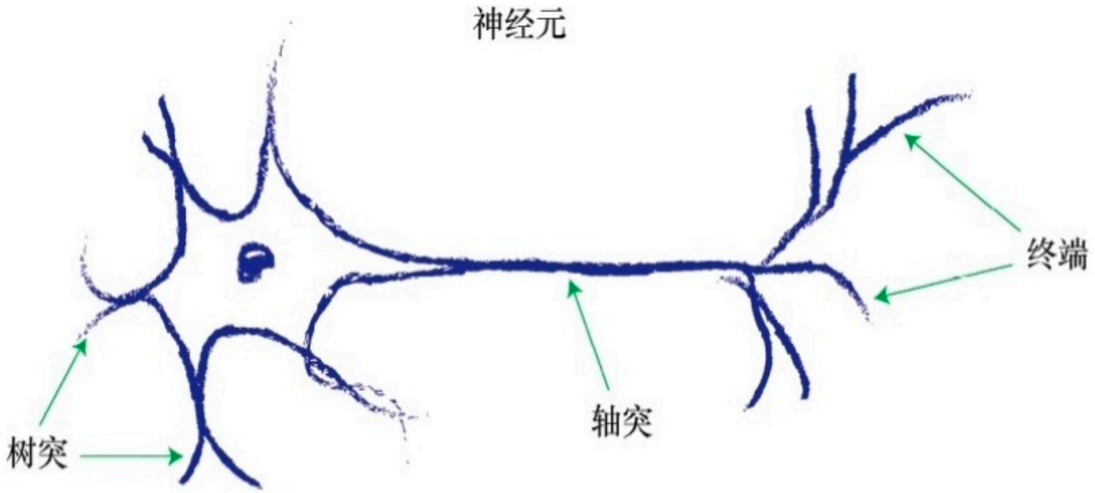


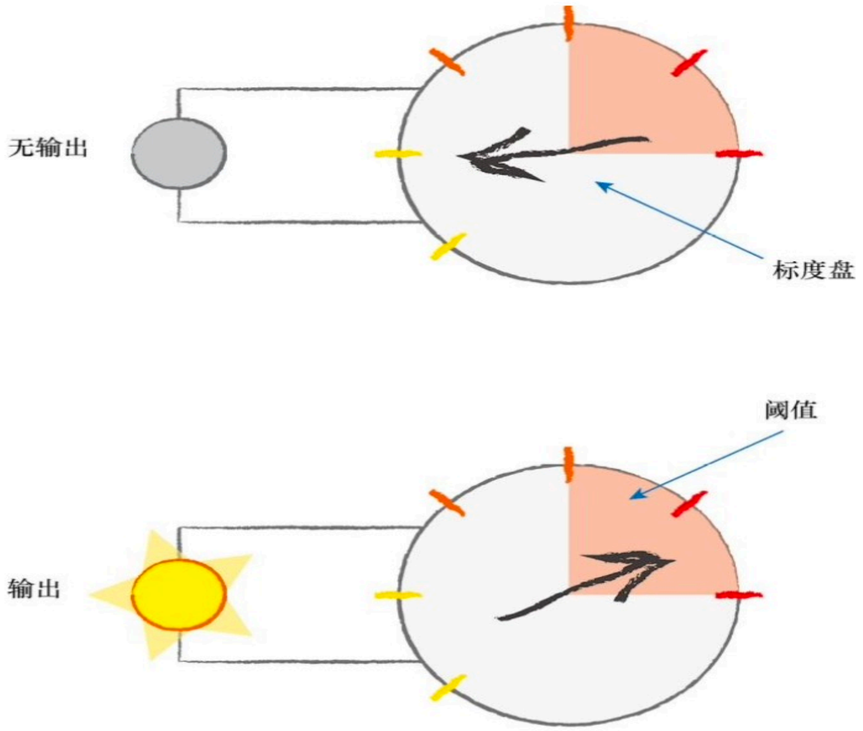
多个分类器一起工作。

这是神经网络的思想。

1 初识神经网络

神经元：





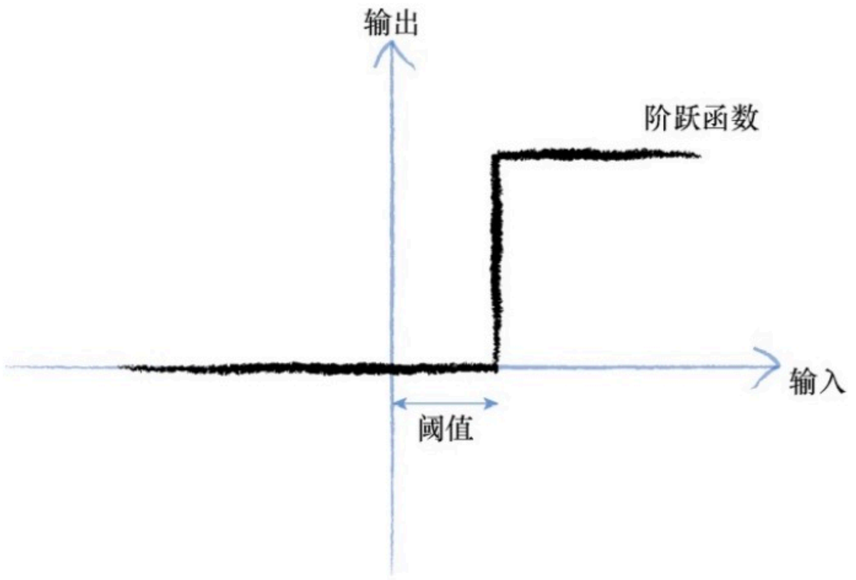
输入 —阈值→ 输出

抑制噪音

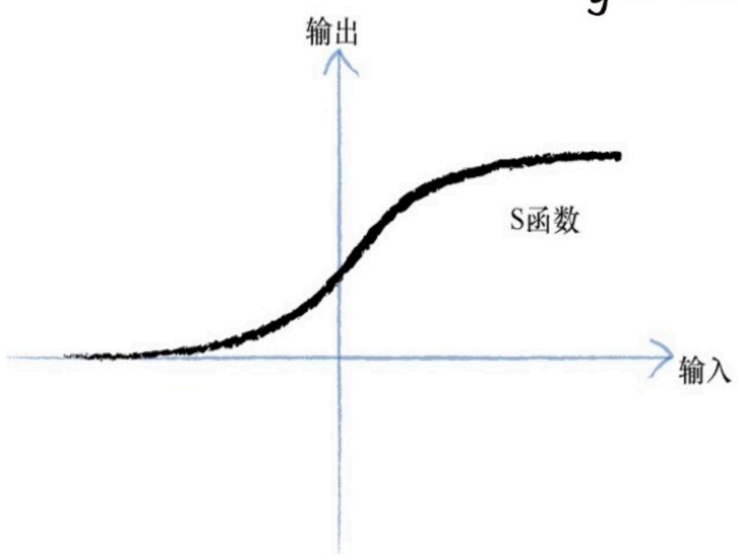
1 初识神经网络

神经元：

激活函数



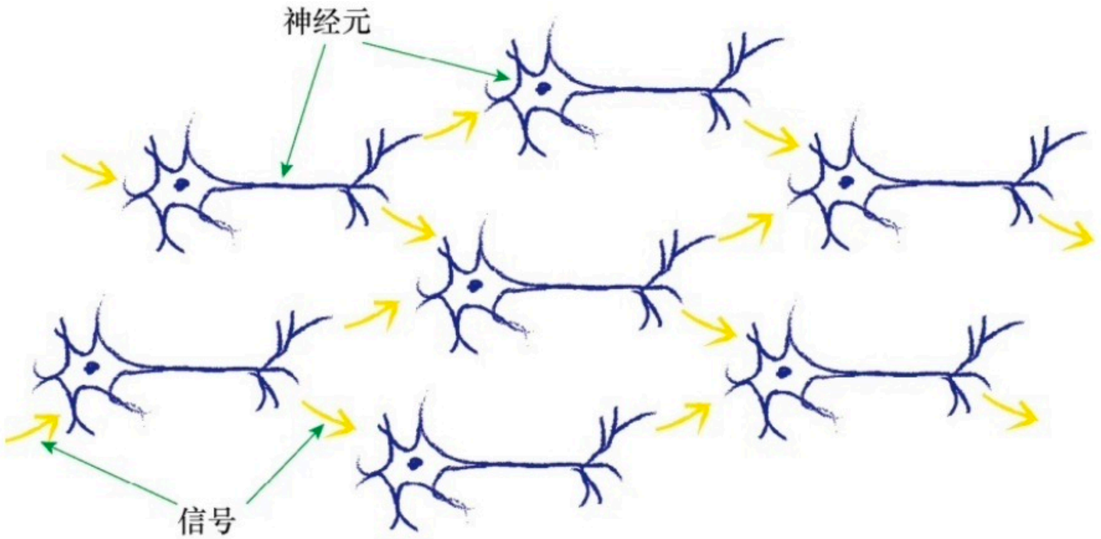
$$y = \frac{1}{1 + e^{-x}}$$



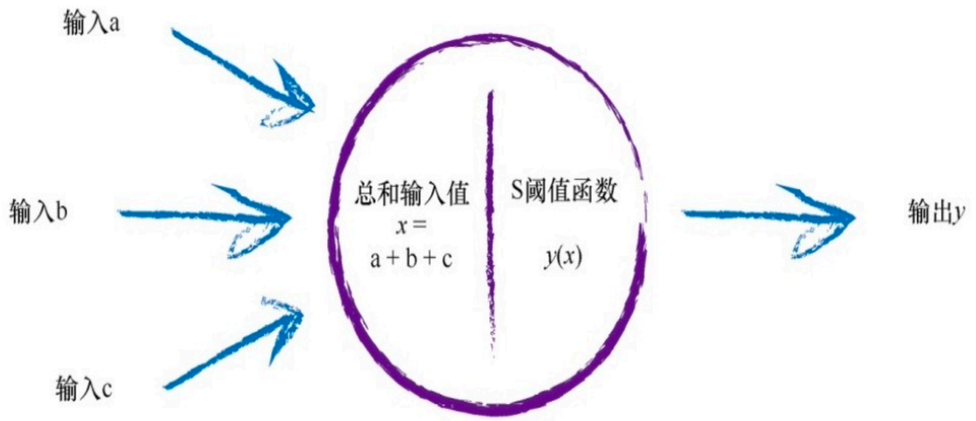
Sigmoid 函数

1 初识神经网络

神经元：



多输入，模糊性



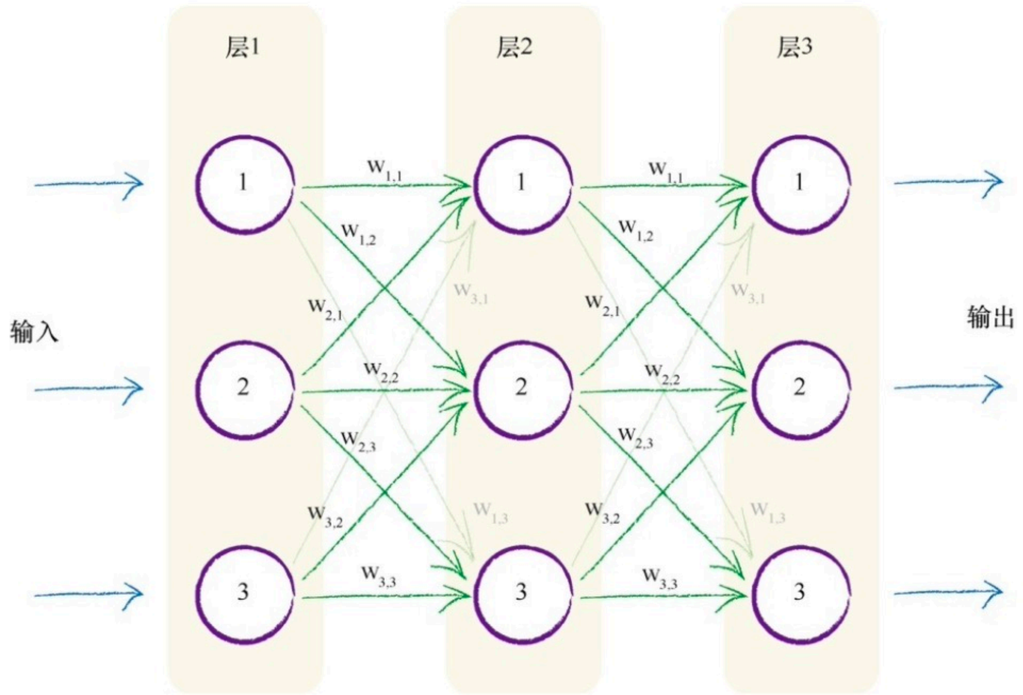
每个神经元接受来自其之前多个神经元的输入，并且如果神经元被激发了，它也同时提供信号给更多的神经元。

感知机

1 初识神经网络

初识神经网络：

复制到人造模型：构建多层神经元，并前后连接



如何进行学习？Or 网络参数如何调整？

输入的总和或阈值函数

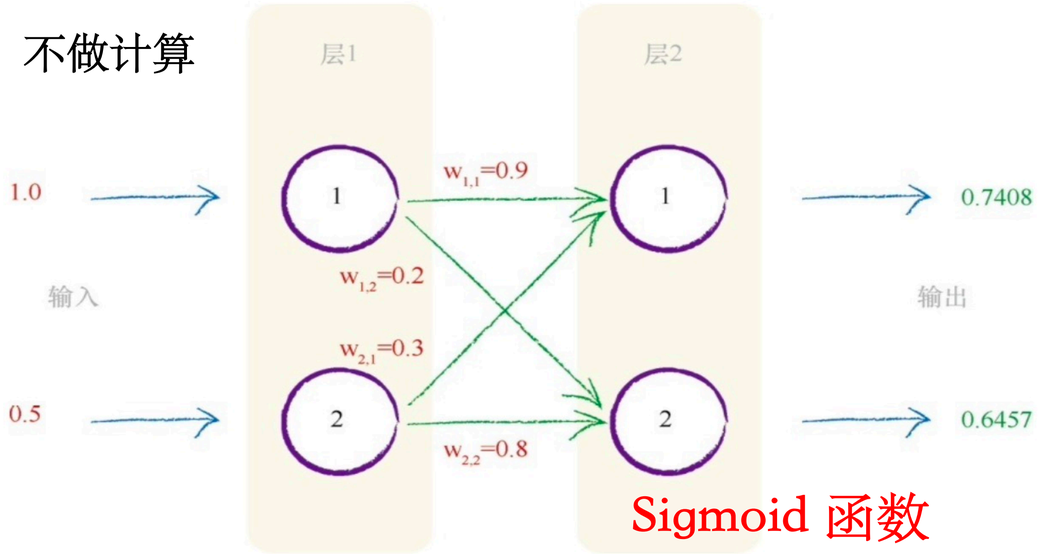
节点之间的连接强度

随着神经网络学习过程的进行，神经网络通过调整优化网络内部的链接权重改进输出，一些权重可能会变为零或接近于零。

1 初识神经网络

神经网络如何工作：

第一层节点是输入层，不做计算



Sigmoid 函数

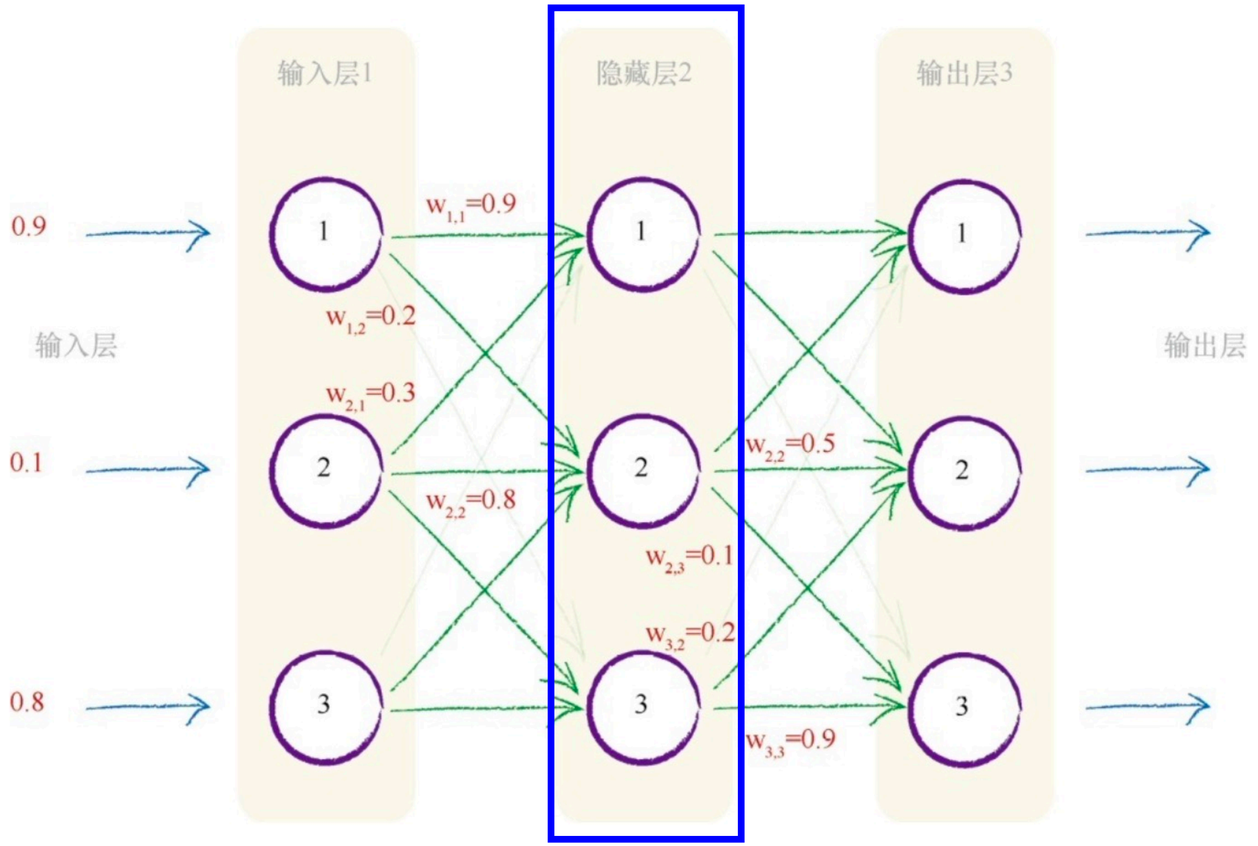
使用矩阵乘法表示所有计算

$$X = W \bullet I$$

$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input\_1 \\ input\_2 \end{pmatrix} = \begin{pmatrix} (input\_1 * w_{1,1}) + (input\_2 * w_{2,1}) \\ (input\_1 * w_{1,2}) + (input\_2 * w_{2,2}) \end{pmatrix}$$

1 初识神经网络

三层神经网络：



$$X_{\text{hidden}} = W_{\text{input\_hidden}} \bullet I$$

3*1　　3*3　　3*1

$$O_{\text{hidden}} = \text{sigmoid}(X_{\text{hidden}})$$

$$X_{\text{output}} = W_{\text{hidden\_output}} \bullet O_{\text{hidden}}$$

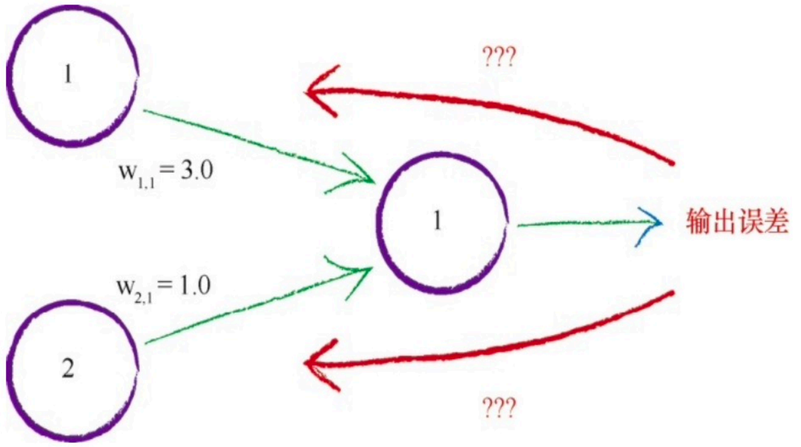
1 初识神经网络

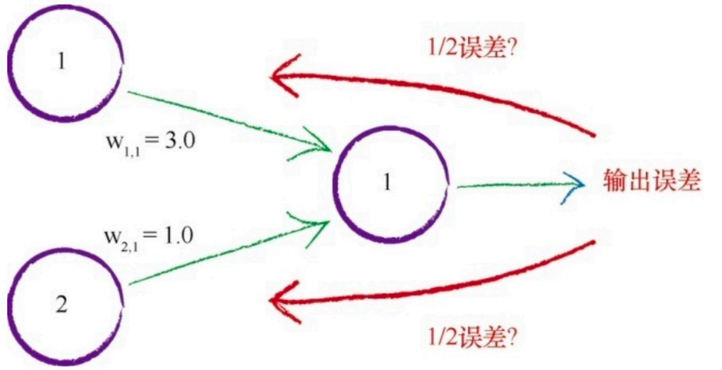
误差传递：

下一步，将神经网络的输出值与训练样本中的输出值进行比较，计算出误差。

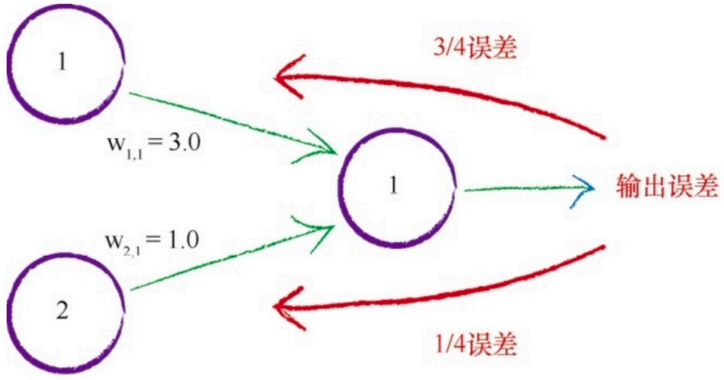
我们需要使用这个误差值来调整神经网络本身，进而改进神经网络的输出值。

二层一个输出





平均模式

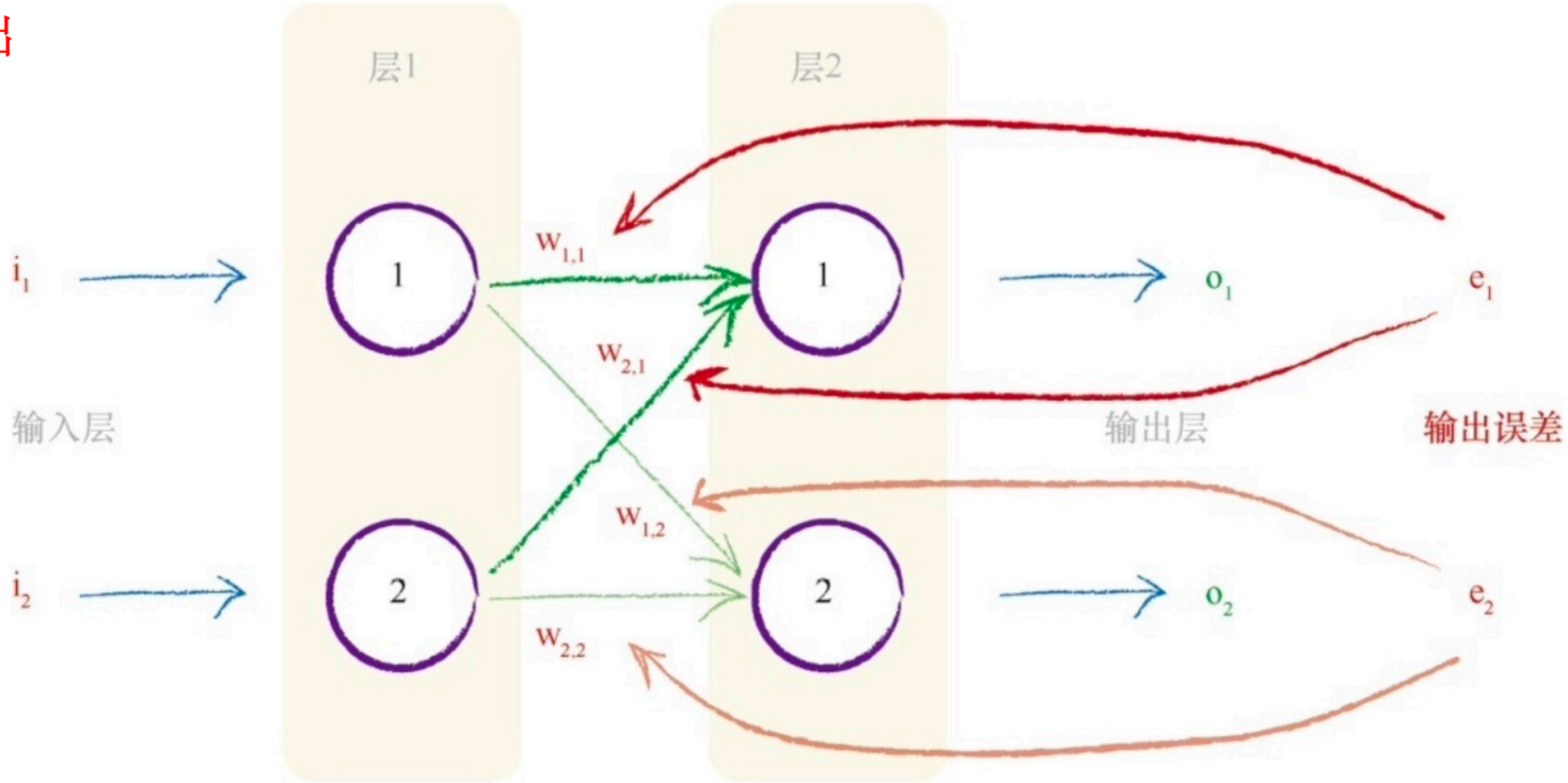


权重模式

1 初识神经网络

多个输出节点反向传播误差：

二层二个输出

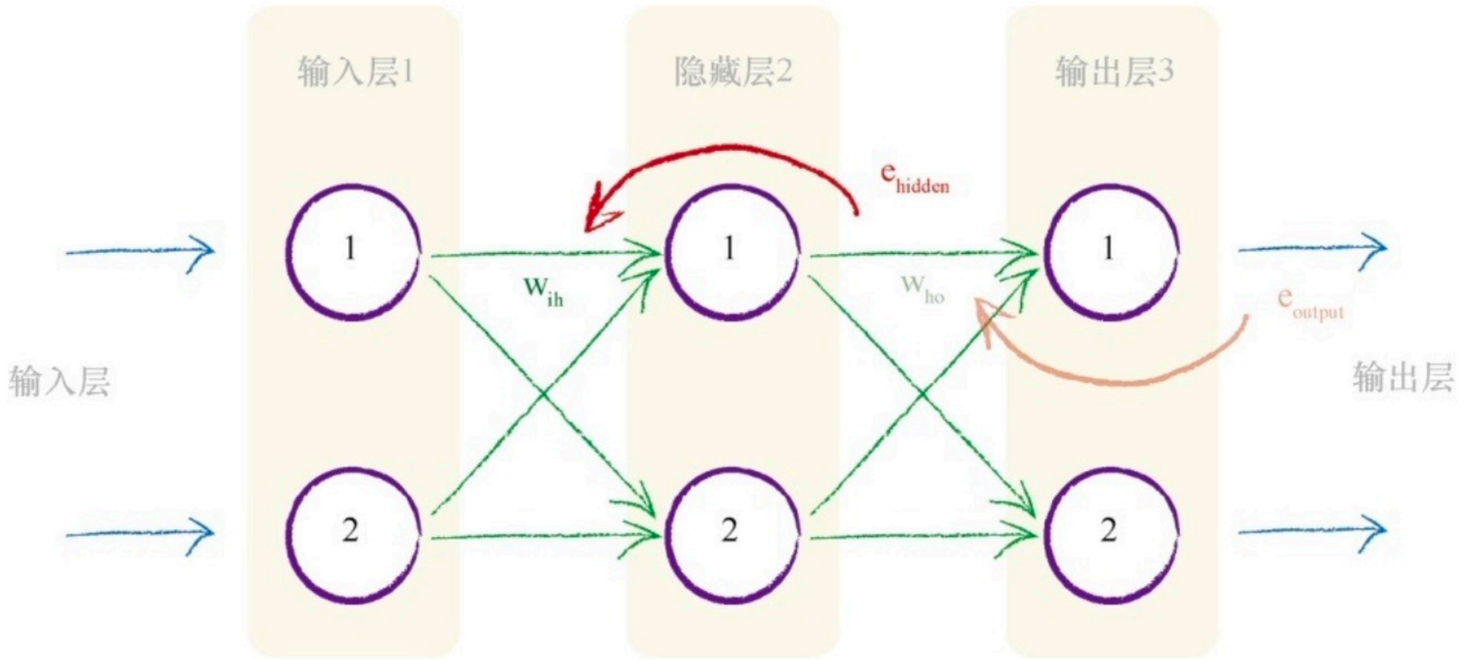


1 初识神经网络

多个输出节点反向传播误差：

三层二个输出

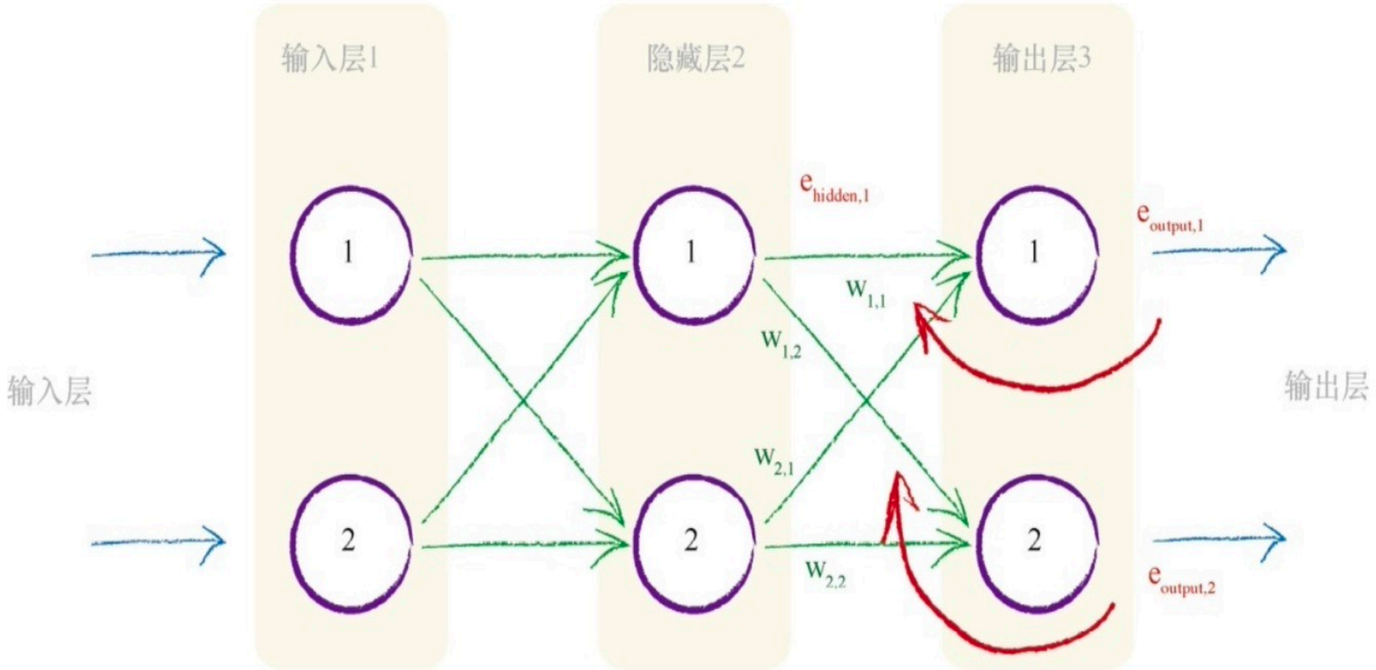
误差信息流



Hidden layer的误差是什么？

1 初识神经网络

多个输出节点反向传播误差：



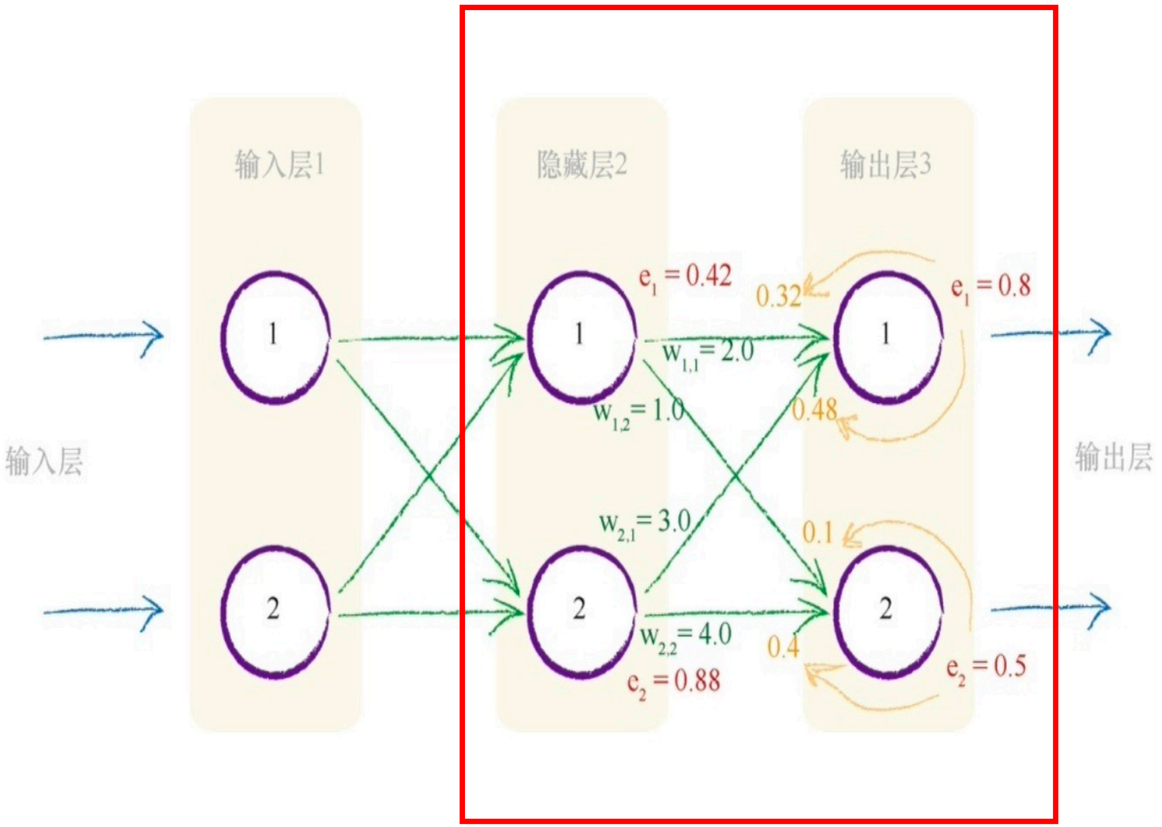
Hidden layer的误差是什么？

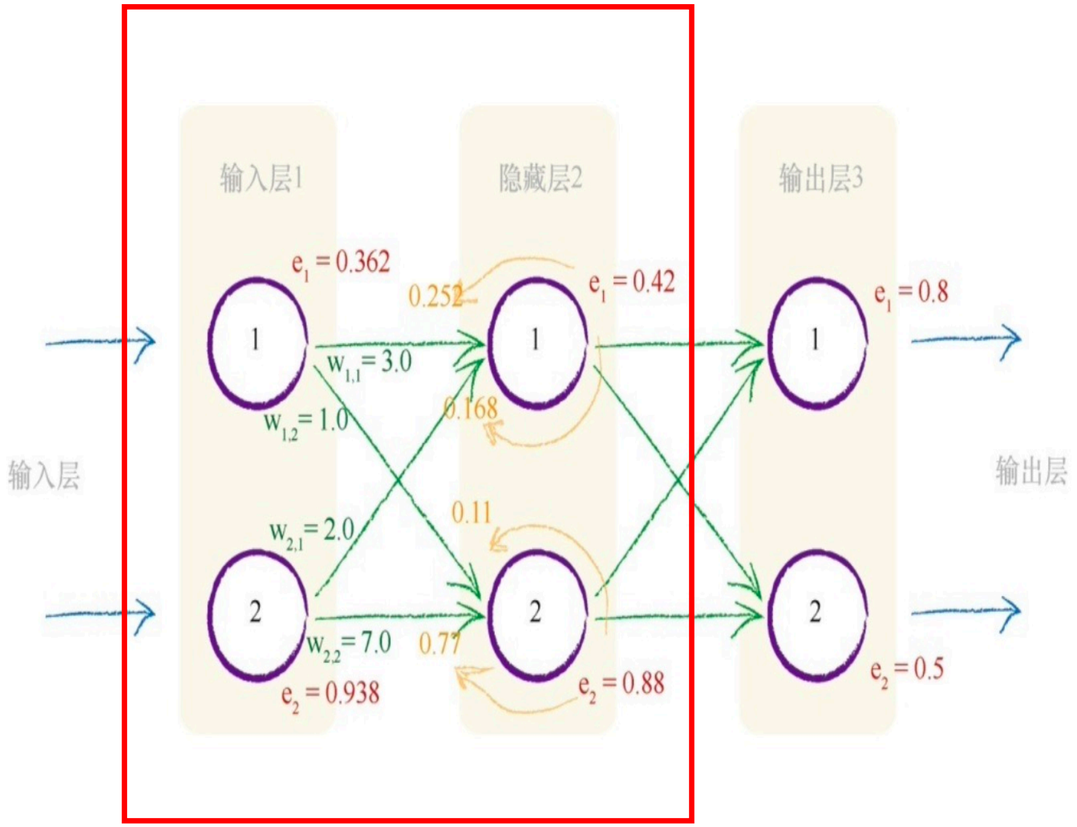
重组这两个链接的误差，
形成这个节点的误差。

例如：与隐藏层节点前向连接所有链接中分割误差的和。

1 初识神经网络

多个输出节点反向传播误差：





1 初识神经网络

多个输出节点反向传播误差：

输出层误差

$$error_{output} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

前向馈送信号和反向传播误差

都可以使用矩阵计算而变得高效

隐藏层误差

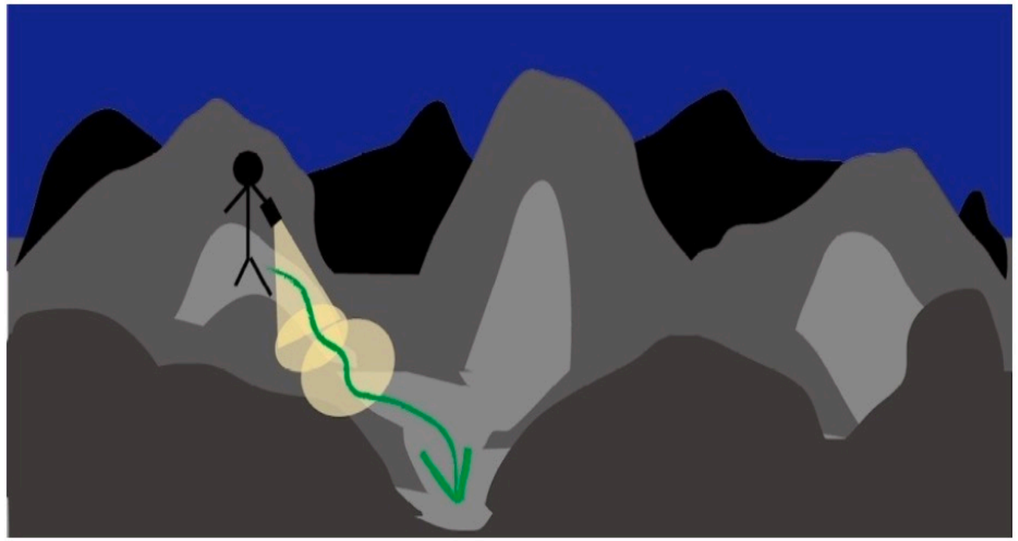
$$error_{hidden} = \begin{pmatrix} \frac{w_{1,1}}{w_{1,1} + w_{2,1}} & \frac{w_{1,2}}{w_{1,2} + w_{2,2}} \\ \frac{w_{2,1}}{w_{2,1} + w_{1,1}} & \frac{w_{2,2}}{w_{2,2} + w_{1,2}} \end{pmatrix} \cdot \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$$

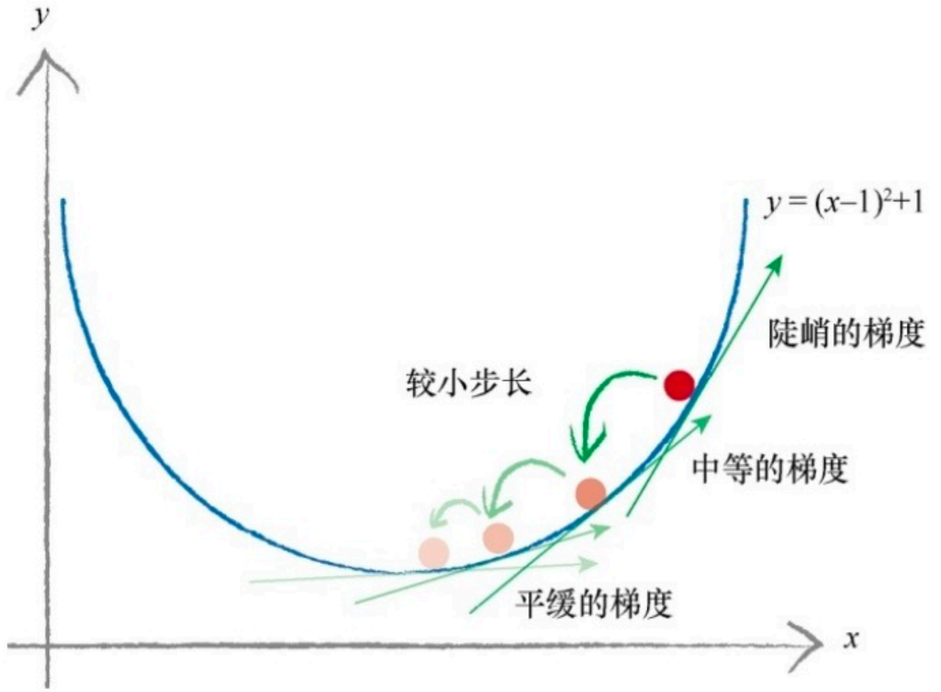
$$error_{hidden} = w^T_{hidden\_output} \cdot error_{output}$$

1 初识神经网络

实际上如何更新权重：

梯度下降(gradient descent)





下山最快就意味着最小化误差函数。

1 初识神经网络

实际上如何更新权重：

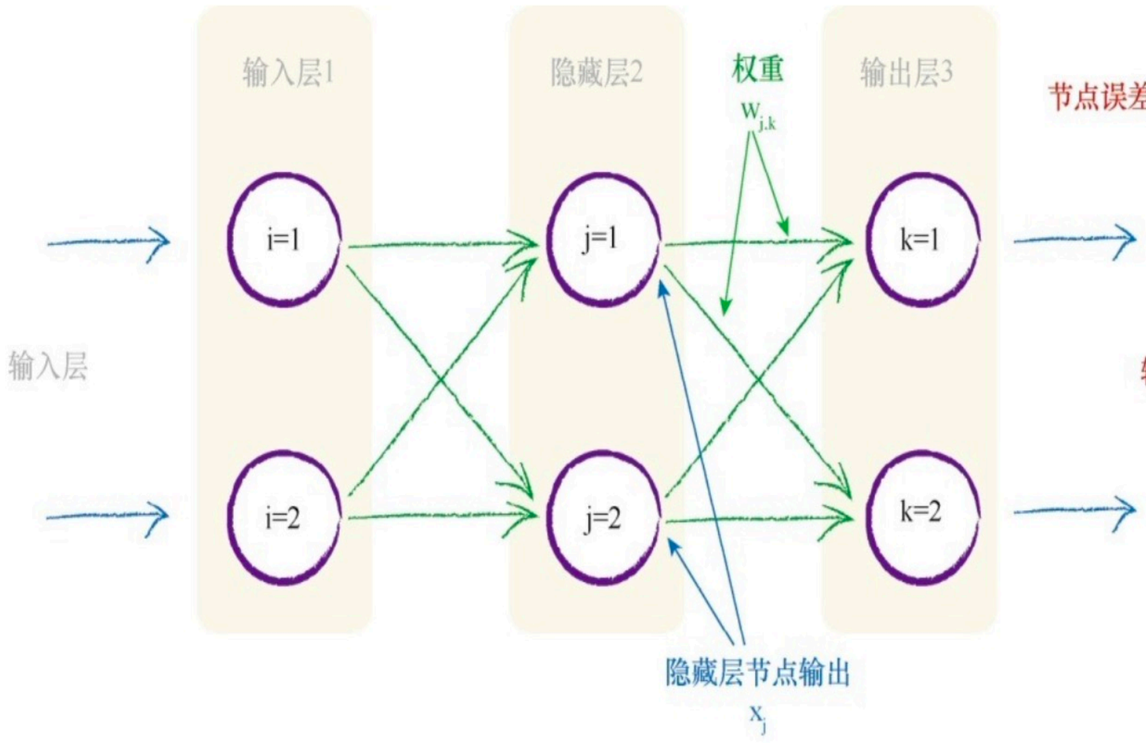
误差函数的选取

网络输出	目标输出	误差 （目标值-实际值）	误差 |目标值-实际值|	误差 （目标值-实际值）^{2}
0.4	0.5	0.1	0.1	0.01
0.8	0.7	-0.1	0.1	0.01
1.0	1.0	0	0	0
求和		0	0.2	0.02

没能优化　　不连续　　还不错

1 初识神经网络

实际上如何更新权重：



$$\frac{\partial E}{\partial w_{j,k}} = \frac{\partial}{\partial w_{j,k}} \Sigma_n (t_n - o_n)^2$$

$$\frac{\partial E}{\partial w_{j,k}} = \frac{\partial}{\partial w_{j,k}} (t_k - o_k)^2$$

1 初识神经网络

实际上如何更新权重：

训练神经网络的关键！

优化隐藏层和输出层之间的权重

$$\frac{\partial E}{\partial w_{j,k}} = -2(t_k - o_k) \cdot \frac{\partial}{\partial w_{j,k}} \text{sigmoid}\left(\Sigma_j w_{j,k} \cdot o_j\right)$$

$$= -2(t_k - o_k) \cdot \text{sigmoid}\left(\Sigma_j w_{j,k} \cdot o_j\right)\left(1 - \text{sigmoid}\left(\Sigma_j w_{j,k} \cdot o_j\right)\right) \cdot o_j$$

1 初识神经网络

实际上如何更新权重：

误差函数的选取

$$\frac{\partial E}{\partial w_{i,j}} = -(e_j) \cdot sigmoid(\Sigma_i w_{i,j} \cdot o_i)(1 - sigmoid(\Sigma_i w_{i,j} \cdot o_i)) \cdot o_i$$

$$new\ w_{j,k} = old\ w_{j,k} - \alpha \cdot \frac{\partial E}{\partial w_{j,k}}$$

1 初识神经网络

实际上如何更新权重：

$$\begin{pmatrix} \Delta w_{1,1} & \Delta w_{2,1} & \Delta w_{3,1} & \cdots \\ \Delta w_{1,2} & \Delta w_{2,2} & \Delta w_{3,2} & \cdots \\ \Delta w_{1,3} & \Delta w_{2,3} & \Delta w_{j,k} & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{pmatrix} = \begin{pmatrix} E_1 * S_1(1-S_1) \\ E_2 * S_2(1-S_2) \\ E_k * S_k(1-S_k) \\ \cdots \end{pmatrix} \cdot \begin{pmatrix} O_1 & O_2 & O_j & \cdots \end{pmatrix}$$

下一层的值

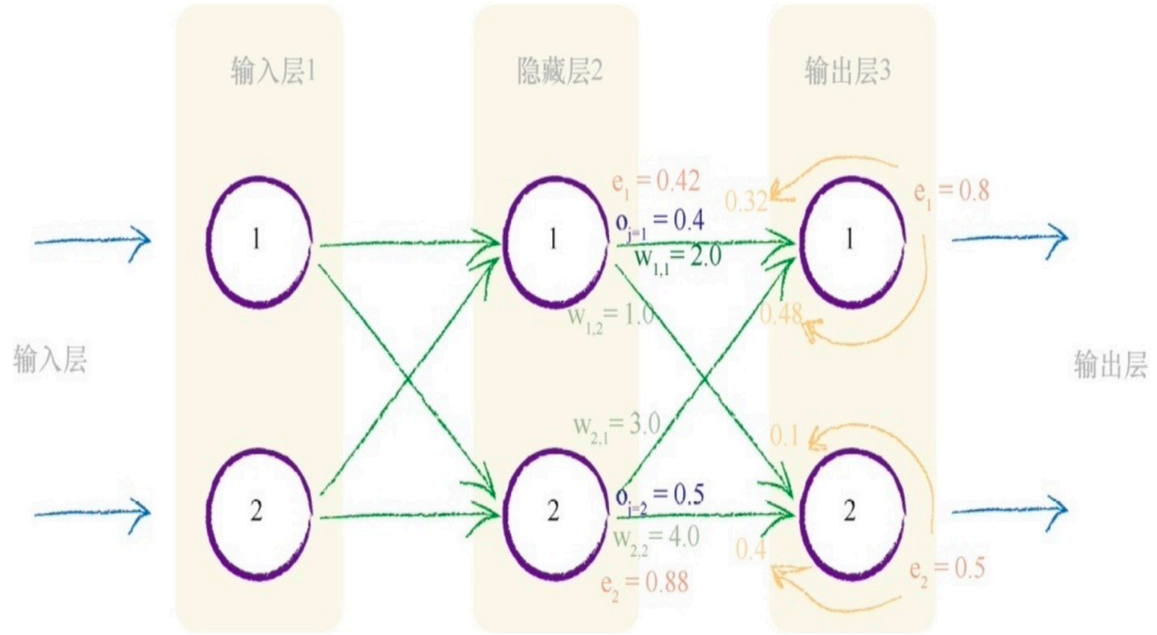
前一层的值

权重更新矩阵

$$\Delta W_{j,k} = \alpha \cdot E_k \cdot O_k(1 - O_k) \cdot O_j^T$$

1 初识神经网络

实际上如何更新权重：



$$\frac{\partial E}{\partial w_{j,k}} = -(t_k - o_k) \cdot sigmoid(\Sigma_j w_{j,k} \cdot o_j)(1 - sigmoid(\Sigma_j w_{j,k} \cdot o_j)) \cdot o_j$$

- 第一项（$t_k - o_k$）得到误差$e_1 = 0.8$。
- S函数内的求和$\Sigma_j w_{j,k} o_j$ 为（2.0×0.4）+（3.0 * 0.5）= 2.3。
- sigmoid $1/(1 + e^{-2.3})$ 为0.909。中间的表达式为0.909 *（1-0.909）= 0.083。
- 由于我们感兴趣的是权重$w_{1,1}$，其中j=1，因此最后一项o_j 也很简单，也就是$o_{j=1}$。此处，o_j 值就是0.4。

-0.0265

1 初识神经网络

实际上如何更新权重：

解决一个机器学习问题主要有两部分：

数据和算法

算法又有三个部分组成：假设函数、损失函数、算法优化。

2 神经网络进阶

大神：

Yann LeCun

BP, CNN

Geoffrey Hinton

BP, DNN

Yoshua Bengio

GAN

2 神经网络进阶

机器学习：

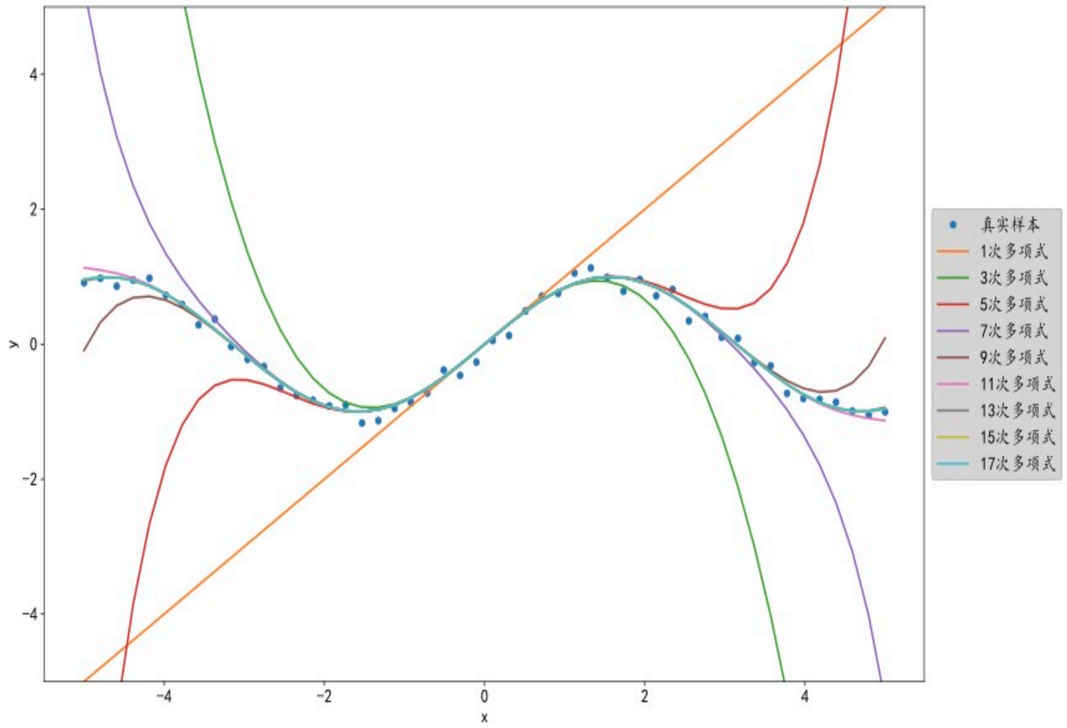
- “对于任务T和性能度量P，一个计算机程序被认为可以从经验E中学习是指：通过经验E改进后，它在任务T上由性能度量P衡量的性能有所提升。”---Mitchell 1997
- **任务T：** 回归、分类、机器翻译、异常检测等；
- **度量P：** 准确率、错误率等；
- **经验E：** 监督、无监督度等。
- **训练集：** 网络模型
- **测试集：** 检验网络，泛化

独立同分布： 随机变量服从同一分布，并且互相独立。

2 神经网络进阶

容量、过拟合、欠拟合：

模型的容量：模型拟合复杂函数的能力，即模型可以表示的函数集（假设空间）的大小。



多项式模型容量示意图

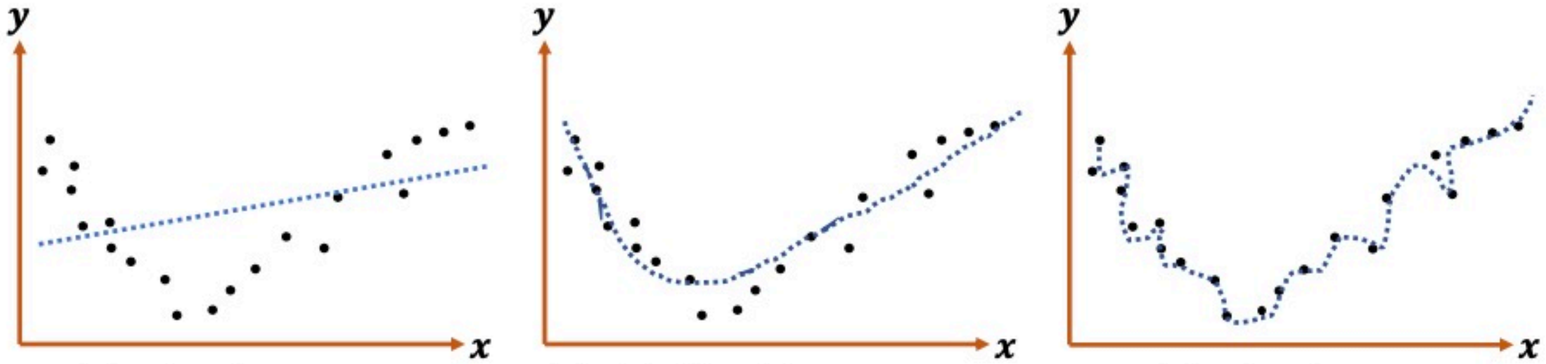
函数的假设空间越大，就越有可能找到一个函数更好地逼近真实分布的函数模型。 但有可能损害模型的泛化能力。

2 神经网络进阶

容量、过拟合、欠拟合：

过拟合(Overfitting)：当模型的容量过大时，导致学习的模型在训练集上面表现较好，但是在测试集的样本上表现不佳。

欠拟合(Underfitting)：当模型的容量过小时，模型不能够很好地学习到训练集数据的模态，导致训练集上表现不佳，同时在测试集的样本上表现也不佳。



（a）欠拟合　（b）模型容量合适　（c）过拟合

2 神经网络进阶

容量、过拟合、欠拟合：

超参数： 在开始学习过程之前设置值的参数，而不是通过训练得到的参数数据。

例如：学习率、权值衰减系数、训练次数 。

验证集： 验证性能

MNIST

训练集-验证集-测试集

选择模型的超参数(模型选择)

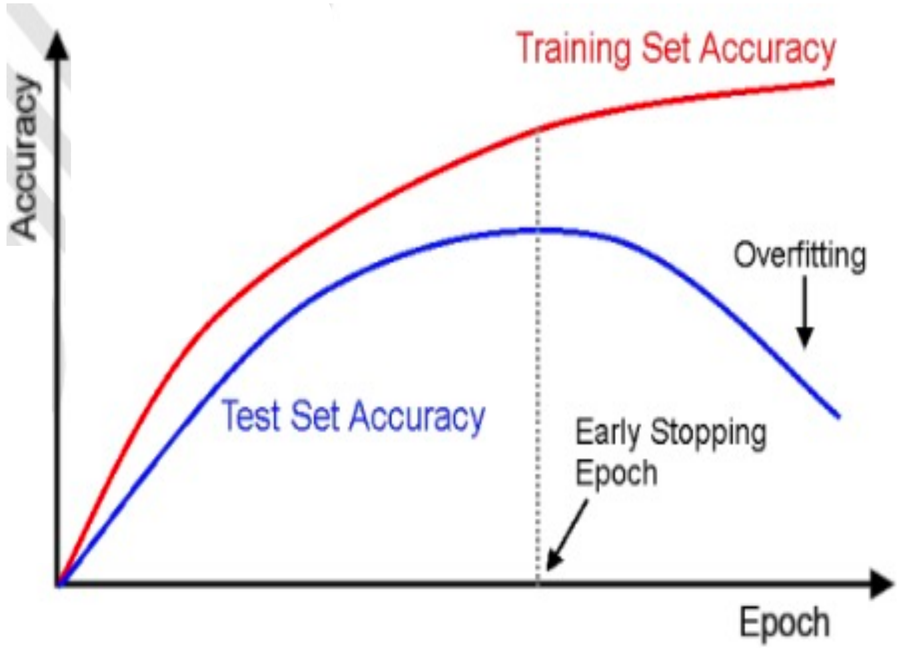
调整网络拓扑结构

判断是否过拟合和欠拟合

2 神经网络进阶

控制过拟合和欠拟合：

Early stopping



监控验证准确率的变化，当发现验证准确率连续n个Epoch没有下降时，可以预测可能已经达到了最适合的Epoch附近，从而提前终止训练。

2 神经网络进阶

容量、过拟合、欠拟合：

正则化：通过限制网络参数的稀疏性， 可以来约束网络的实际容量。

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n + \varepsilon$$

网络参数

损失函数+参数稀疏性约束：

$$min\ \mathcal{L}(f_\theta(\boldsymbol{x}), y) + \lambda \cdot \Omega(\theta), (\boldsymbol{x}, y) \in \mathbb{D}^{\text{train}}$$

正则化系数

向量范数

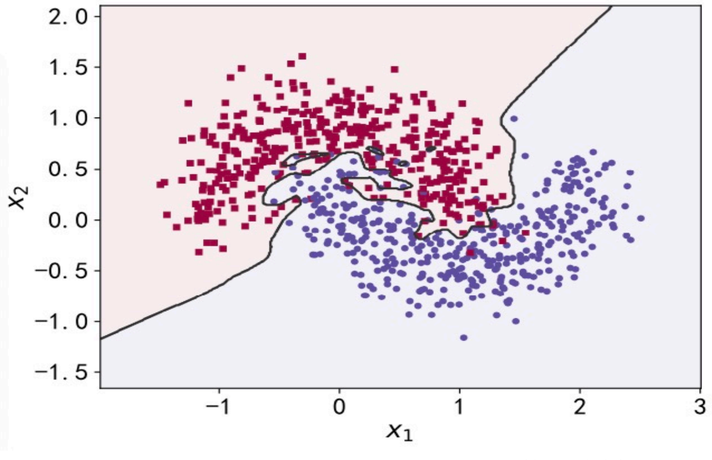
$$\Omega(\theta) = \sum_{\theta_i} \|\theta_i\|_l$$

2 神经网络进阶

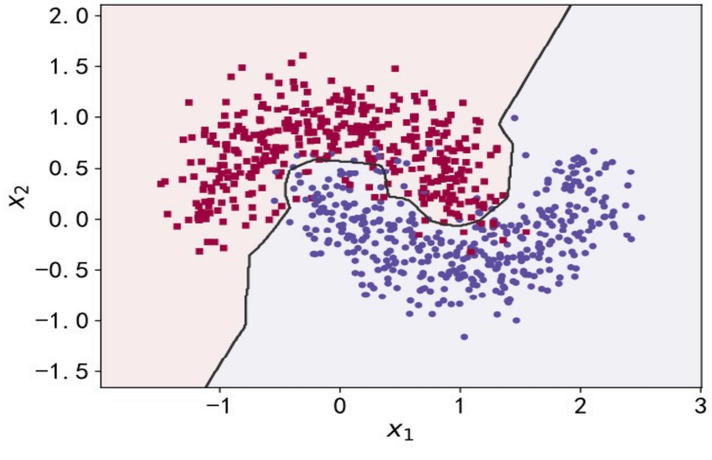
容量、过拟合、欠拟合：

L2 正则化项

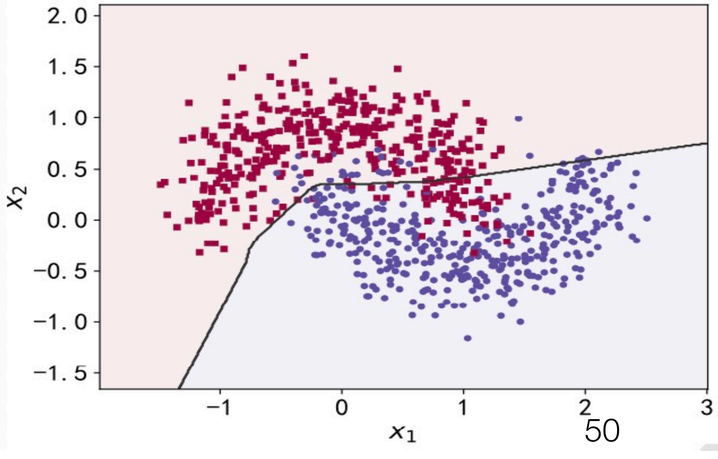
正则化系数:0.00001



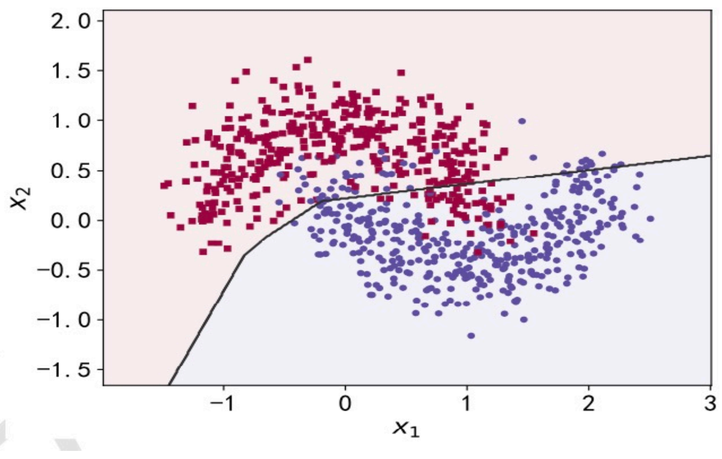
正则化系数:0.001



正则化系数:0.1



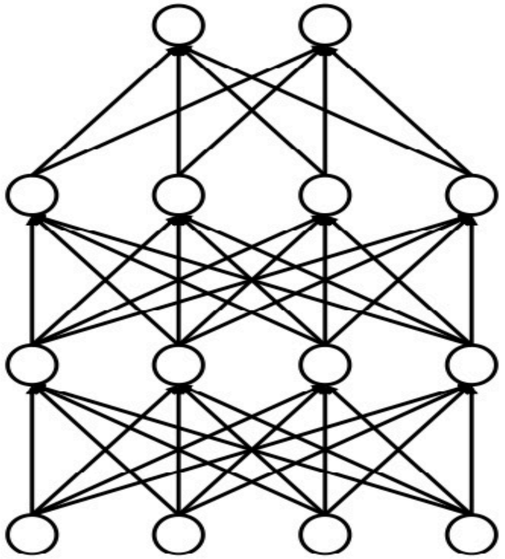
正则化系数:0.13



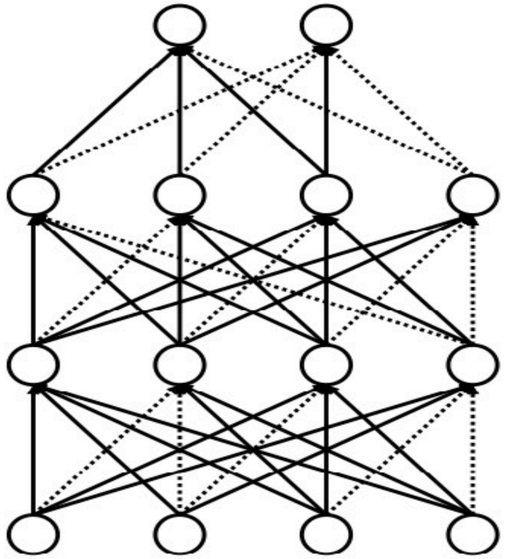
2 神经网络进阶

容量、过拟合、欠拟合：

Dropout：通过随机断开神经网络的连接，减少每次训练时实际参与计算的模型的参数量。



(a)标准全连接网络



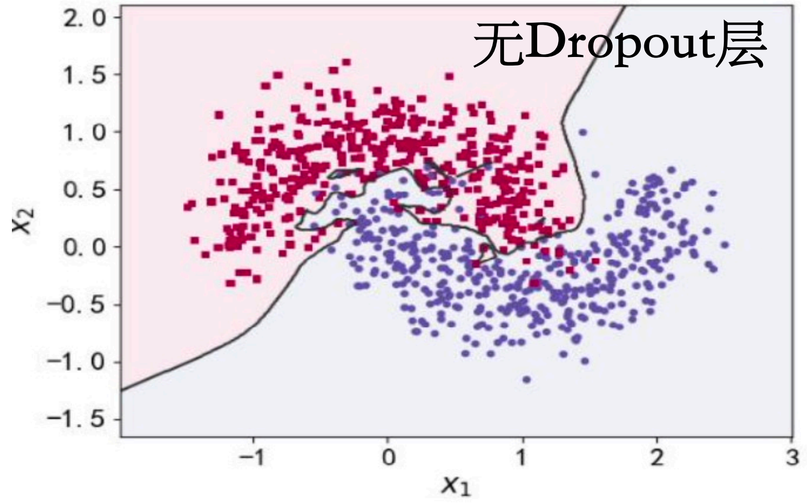
(b)带Dropout的全连接网络

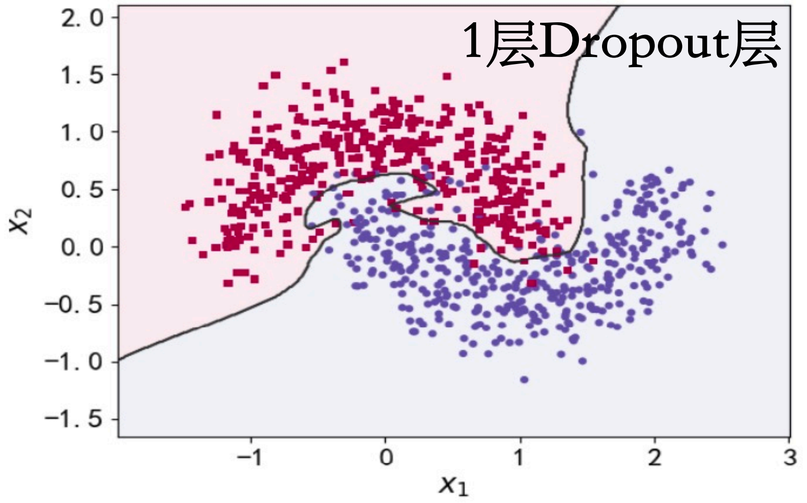
每条连接是否断开符合某种预设的概率分布

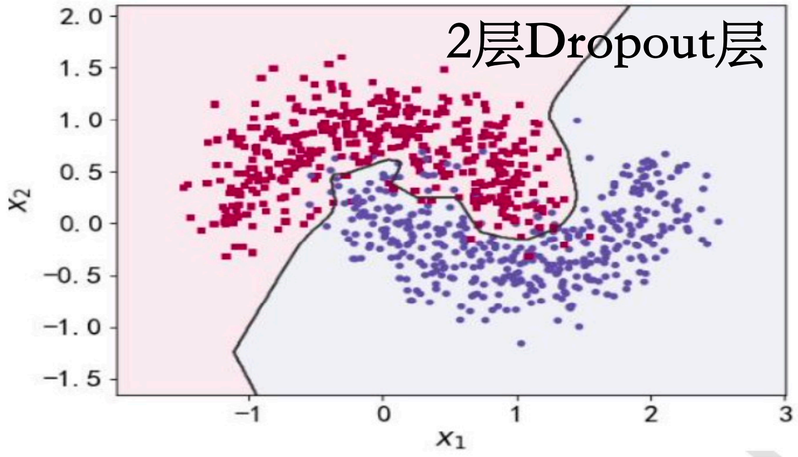
在测试时，Dropout 会恢复所有的连接，保证模型测试时获得最好的性能。

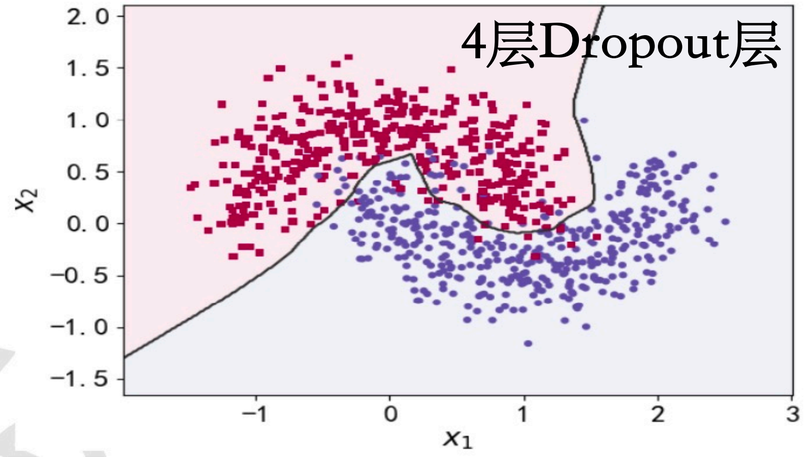
2 神经网络进阶

容量、过拟合、欠拟合：





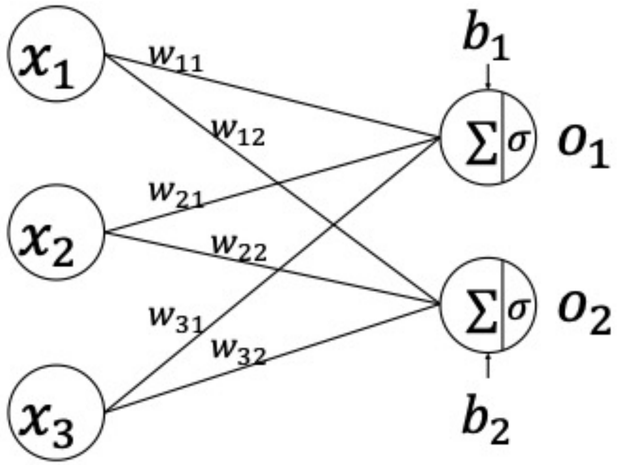




2 神经网络进阶

全连接：

每个输出节点与全部的输入节点相连接，这种网络层称为全连接层(Fully-connected Layer)，或者稠密连接层 (Dense Layer)



输出矩阵　　输入矩阵　　权值矩阵　　偏置矩阵

$$\begin{bmatrix} o_1^{(1)} & o_2^{(1)} \\ o_1^{(2)} & o_2^{(2)} \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} \end{bmatrix} @ \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix}$$

2 神经网络进阶

激活函数：

Sigmoid函数也叫Logistic 函数：

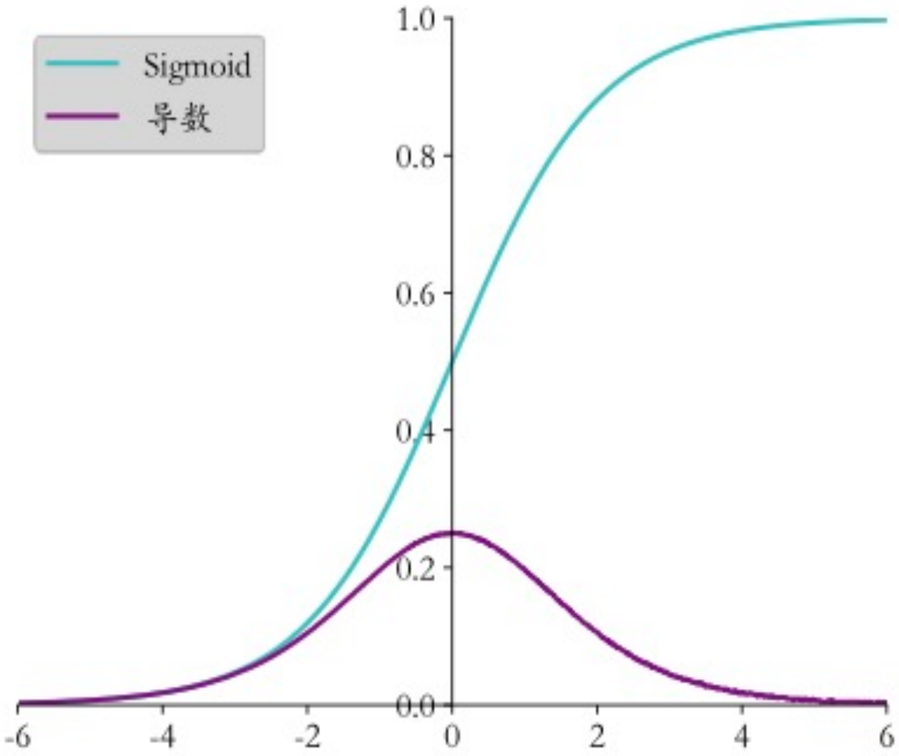
$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

把$x \in R$的输入“压缩”到$x \in (0,1)$区间

转译为：概率输出，信号强度

在输入值较大或较小时容易出现梯度值接近于 0 的现象，

称为梯度弥散现象



2 神经网络进阶

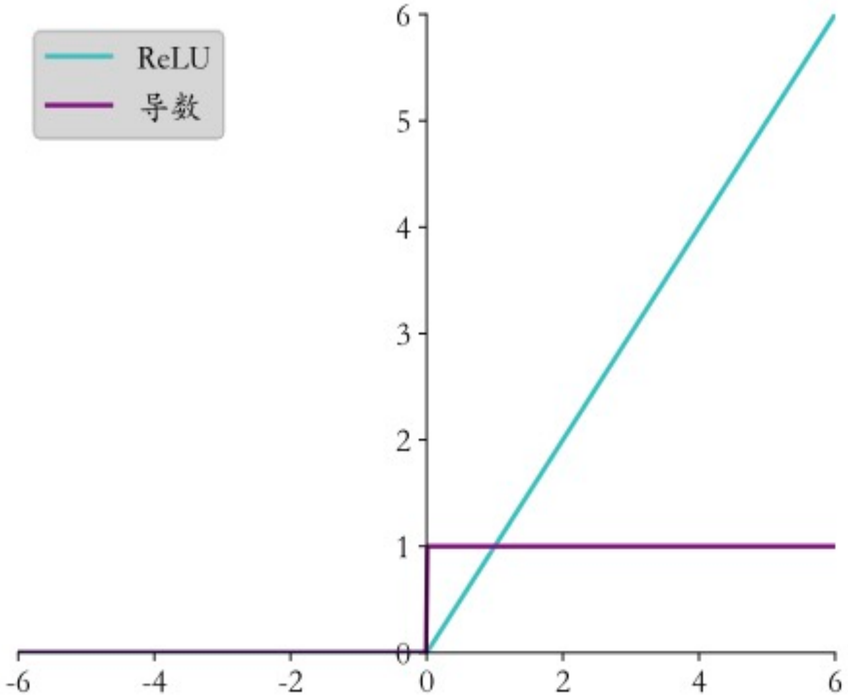
激活函数：

ReLU (REctified Linear Unit，修正线性单元)函数

$$\mathrm{ReLU}(x) = max(0, x)$$

可缓解梯度弥散和梯度爆炸。

ReLU 函数在x<0时导数值恒为0，也可能会造成梯度弥。

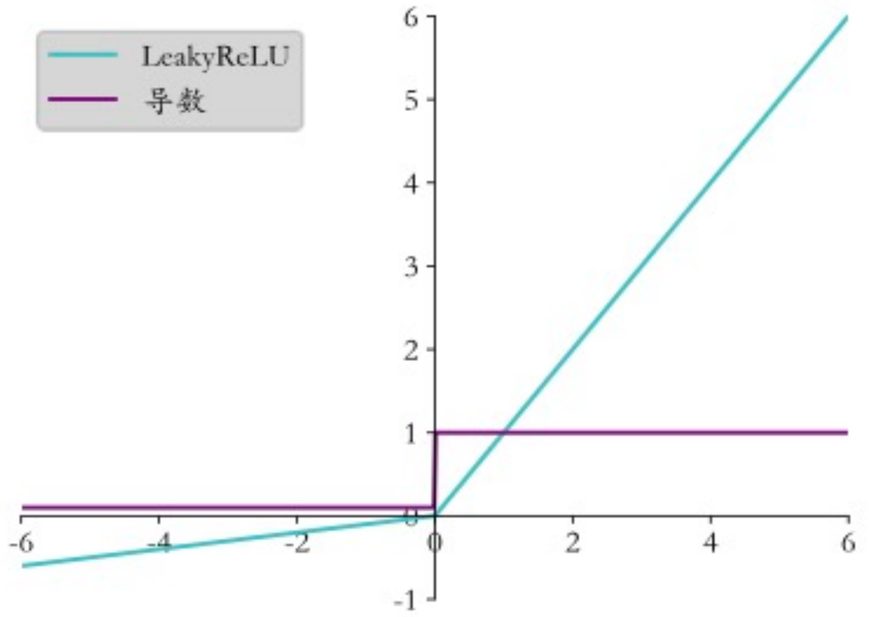


2 神经网络进阶

激活函数：

LeakyReLU函数

$$\text{LeakyReLU} \triangleq \begin{cases} x & x \geq 0 \\ px & x < 0 \end{cases}$$

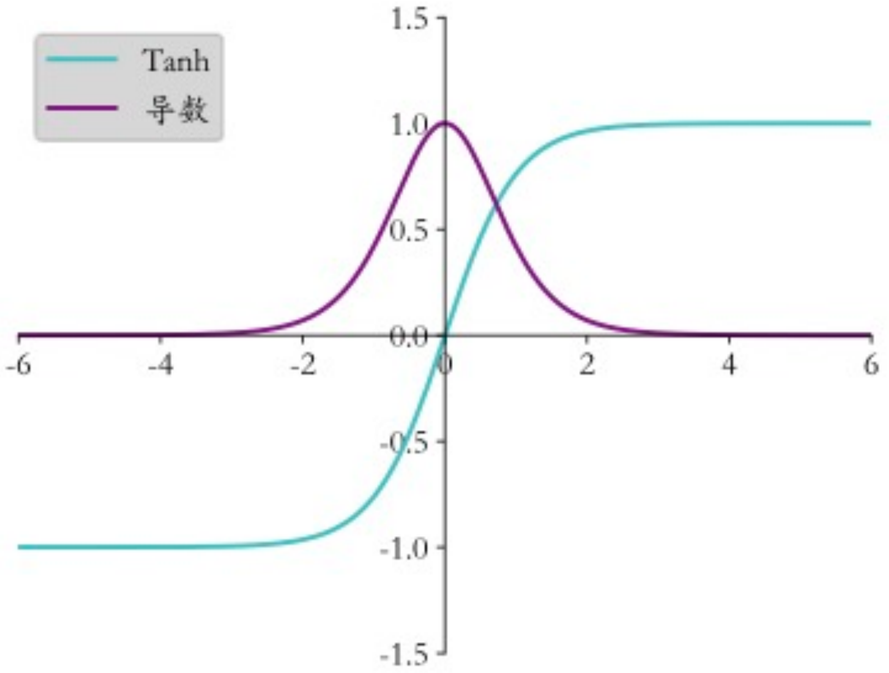


2 神经网络进阶

激活函数：

双曲正切函数

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



2 神经网络进阶

输出层设计：

网络的最后一层：完成维度变换、特征提取、是否使用激活函数，以及使用什么激活函数等

- 输出属于整个实数空间，或者某段普通的实数空间，比如函数值趋势的预 测，年龄的预测问题等。

*输出层可以不加激活函数。误差的计算直接基于最后一层的输出**o**和真实值**y**进行计算。*

- 输出值在[-1, 1]之间

可以简单地使用 tanh 激活函数

- 输出值特别地落在[0, 1]的区间，如图片生成，图片像素值一般用[0, 1]区间的值表示;或者二分类问题的概率，如硬币正反面的概率预测问题。

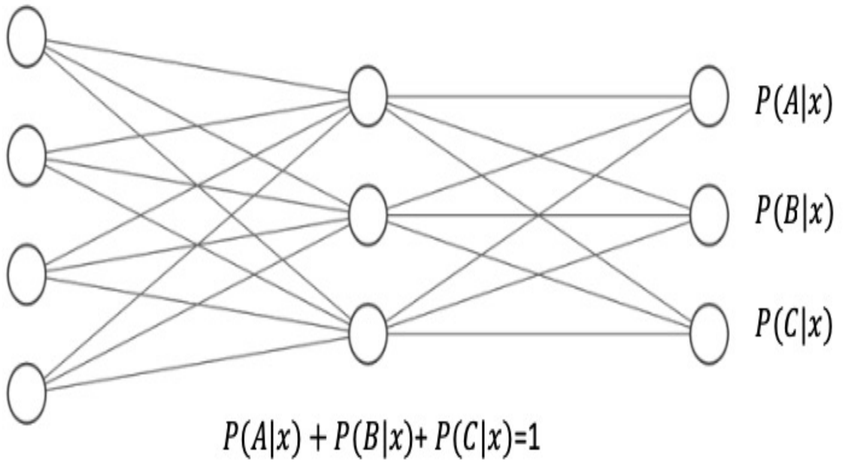
Sigmoid 函数刚好具有此功能。

2 神经网络进阶

输出层设计：

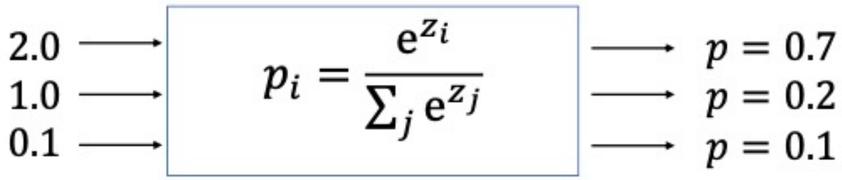
网络的最后一层：完成维度变换、特征提取、是否使用激活函数，以及使用什么类型的激活函数等

- 输出值落在[0, 1]的区间，并且所有输出值之和为 1，常见的如：多分类问题，如 MNIST 手写数字图片识别，图片属于10个类别的概率之和应为 1。



Softmax函数

$$Softmax(Z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$



2 神经网络进阶

误差计算：

在搭建完模型结构后，下一步就是选择合适的误差函数来计算误差。 其中均方差函数和交叉熵函数在深度学习中比较常见，均方差函数主要用于回归问题，交叉熵函数主要用于分类问题。

均方差误差函数：

$$\mathbf{MSE}(\boldsymbol{y},\boldsymbol{o}) = \frac{1}{d_{out}} \sum_{i=1}^{d_{out}} (y_i - o_i)^2$$

当 MSE 函数达到最小值 0 时，输出等于真实标签， 此时神经网络的参数达到最优状态。

2 神经网络进阶

误差计算：

交叉熵函数

某个信息分布的P(i)的熵（信息的不确定性）定义为：

$$H(P) \triangleq -\sum_{i} P(i) \log_2 P(i)$$

e.g. [0,0,0,1], [0.1,0.1,0.1,0.7] , [0.25,0.25,0.25,0.25]

由于0<P(i)<1，因此熵$H(P)$总是大于等于 0。当熵取得最小值 0 时，不确定性为 0。

2 神经网络进阶

误差计算：

交叉熵函数

$$H(p||q) \triangleq -\sum_i p(i) \log_2 q(i)$$

交叉熵可以分解为p的熵$H(p)$和p与q的 KL 散度

$$H(p||q) = H(p) + D_{KL}(p||q)$$

2 神经网络进阶

误差计算：

交叉熵函数

$$D_{KL}(p||q) = \sum_i p(i)\log\left(\frac{p(i)}{q(i)}\right)$$

用于衡量 2 个分布之间距离的指标。

$p = q$时，$D_{KL}(p||q)$取得最小值0，p与q之间的差距越大，$D_{KL}(p||q)$也越 大。

特别地，当分类问题中 y 的编码分布p采用 One-hot 编码y时：$H(p) = 0$，此时

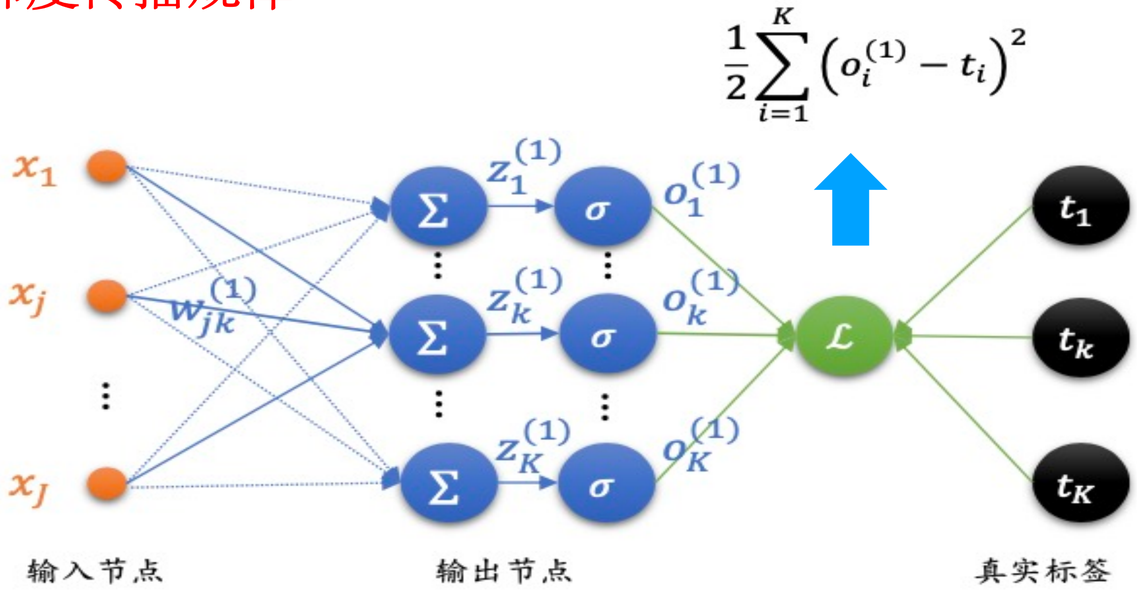
$$H(p||q) = H(p) + D_{KL}(p||q) = D_{KL}(p||q)$$

退化到真实标签分布y与输出概率分布o之间的 KL 散度上。

2 神经网络进阶

反向传播算法：

梯度传播规律



全连接层模型

$$\frac{\partial \mathcal{L}}{\partial w_{jk}} = (o_k - t_k)\frac{\partial o_k}{\partial w_{jk}}$$

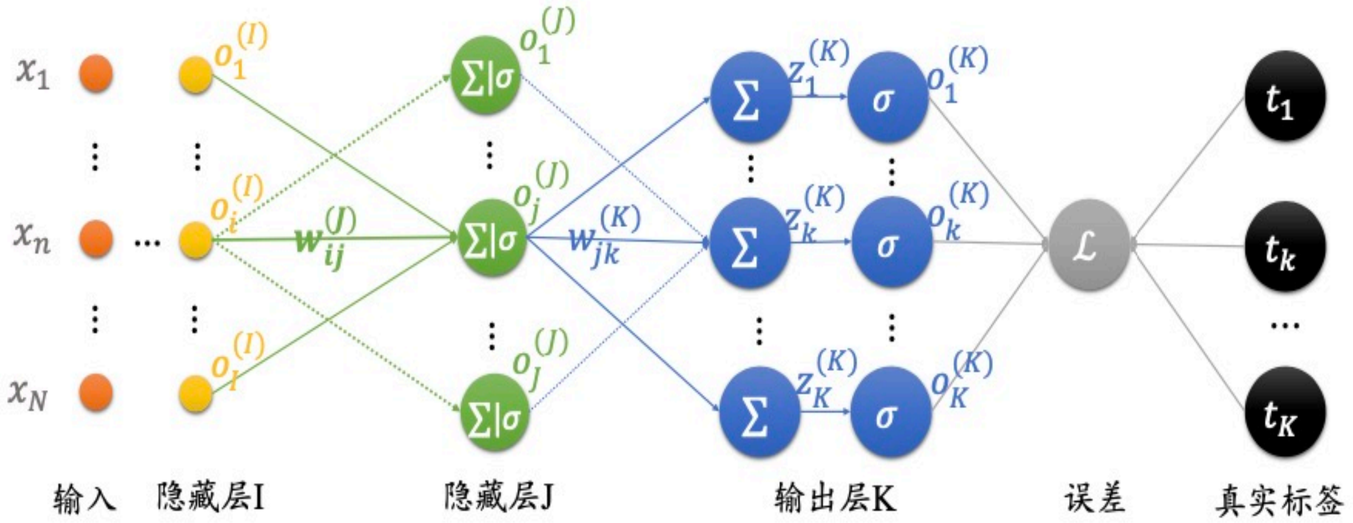
$$\frac{\partial \mathcal{L}}{\partial w_{jk}} = \underbrace{(o_k - t_k)o_k(1-o_k)}_{\delta_k} x_j$$

表征连接线的

终止节点的误差梯度传播的某种特性

2 神经网络进阶

反向传播算法：



链式法则

输出层：

$$\frac{\partial \mathcal{L}}{\partial w_{jk}} = \delta_k^{(K)} o_j$$

$$\delta_k^{(K)} = o_k(1-o_k)(o_k - t_k)$$

倒数第二层：

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \delta_j^{(J)} o_i$$

$$\delta_j^{(J)} = o_j(1-o_j)\sum_k \delta_k^{(K)} w_{jk}$$

倒数第三层：

$$\frac{\partial \mathcal{L}}{\partial w_{ni}} = \delta_i^{(I)} o_n$$

$$\delta_i^{(I)} = o_i(1-o_i)\sum_j \delta_j^{(J)} w_{ij}$$

3 常见神经网络

决策树（Decision Tree）：

决策树：对一组输入特征迭代分类

Boosting： combining many weak learners (trees) into a strong classifier.

一棵树的输出

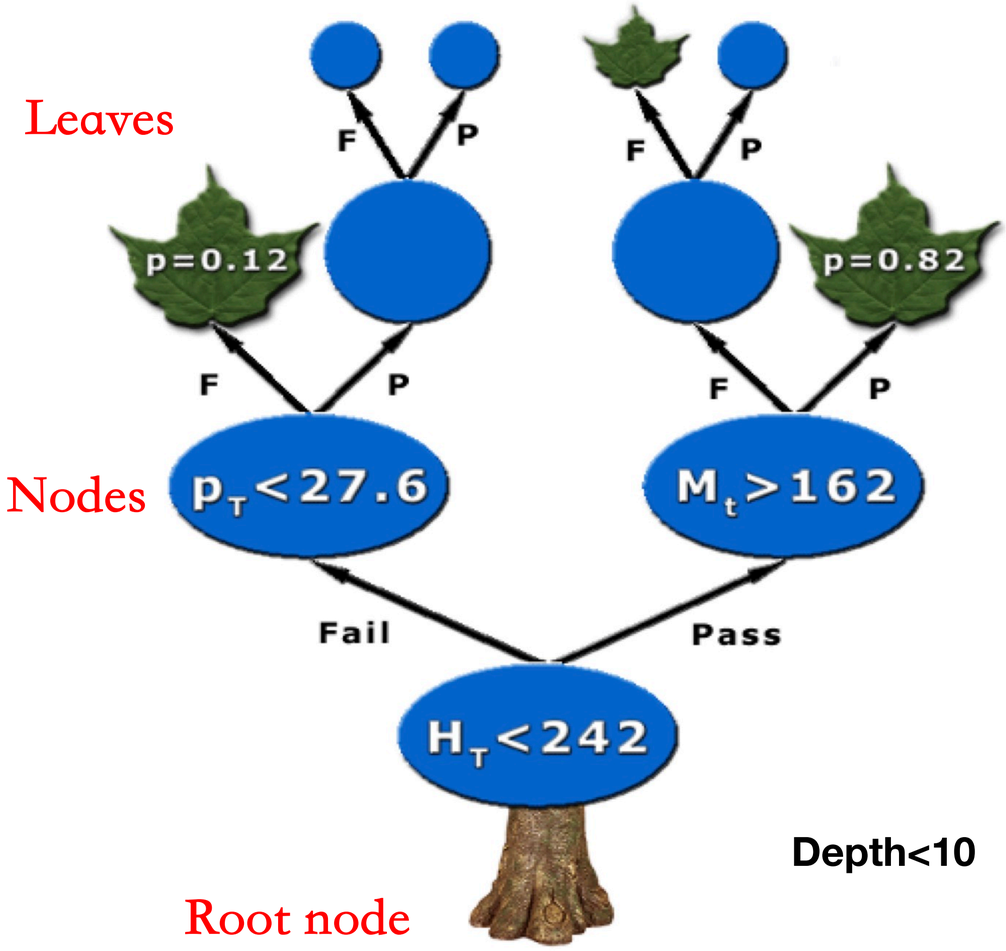
$$\hat{y}(x) = \sum_t w_t h_t(x)$$

权重，随迭代变化

The goal is to minimize an objective function

$$O(x) = \sum_i l(\hat{y}_i, y_i) + \sum_t \Omega(f_t)$$

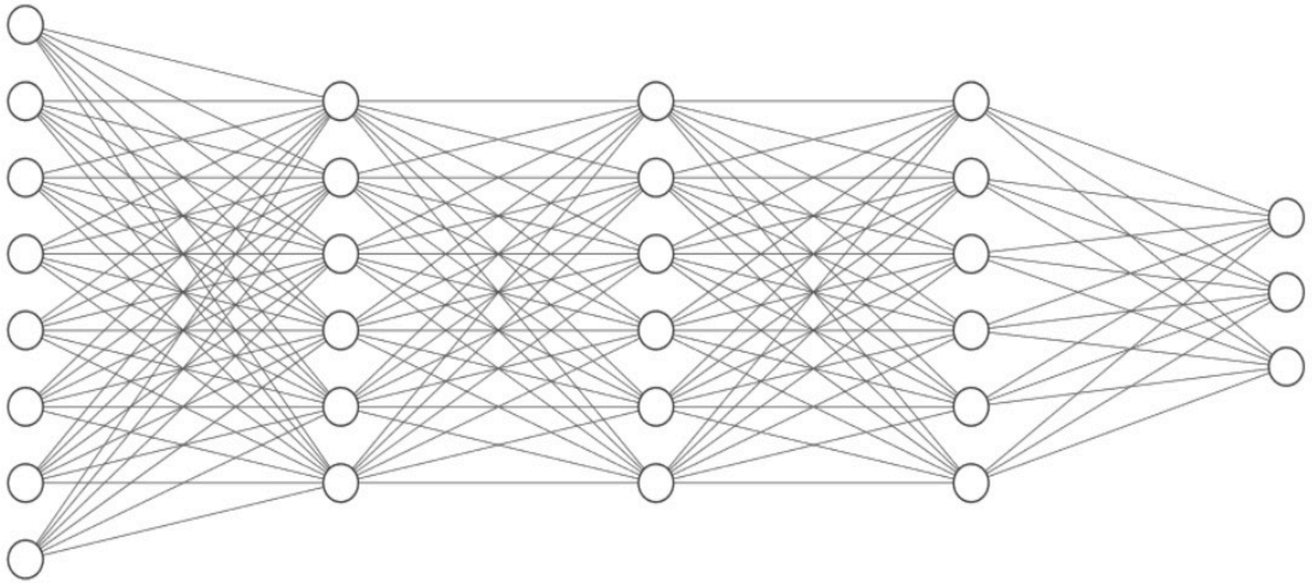
误差函数　　正规化函数



3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

全连接网络的问题



参数量过大

对输入节点数为 n，输出节点数为m的全连接层来说，$\boldsymbol{W}$张量包含的参数量共有$n \cdot m$个，$\boldsymbol{b}$向量包含的参数量有m个，则全连接层的总参数量为$n \cdot m + m$。

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

解决方法

（1）局部相关性：分析输入节点对输出节点的重要性分布，仅考虑较重要的一部分输入节点。

$$o_j = \sigma\left(\sum_{i \in top(I,j,k)} w_{ij}x_i + b_j\right)$$

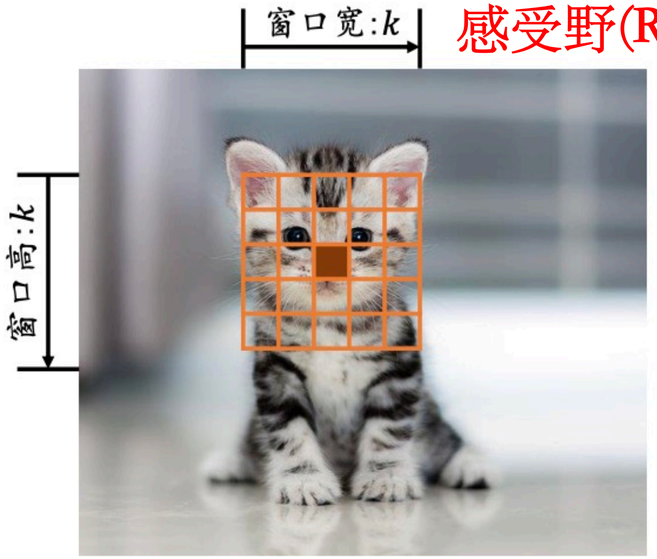
问题就转变为探索第I层中对于J层中的j号节点重要性最高的前k个节点集合 。

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

解决方法：

存在着大量以位置或距离作为重要性分布衡量标准的数据



感受野(Receptive Field)

e.g. 图片每个像素点和周边像素点的关联度更大(位置相关)

当前位置的节点与大小为k的窗口内的所有像素相连接，与窗口外的其它像素点无关

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

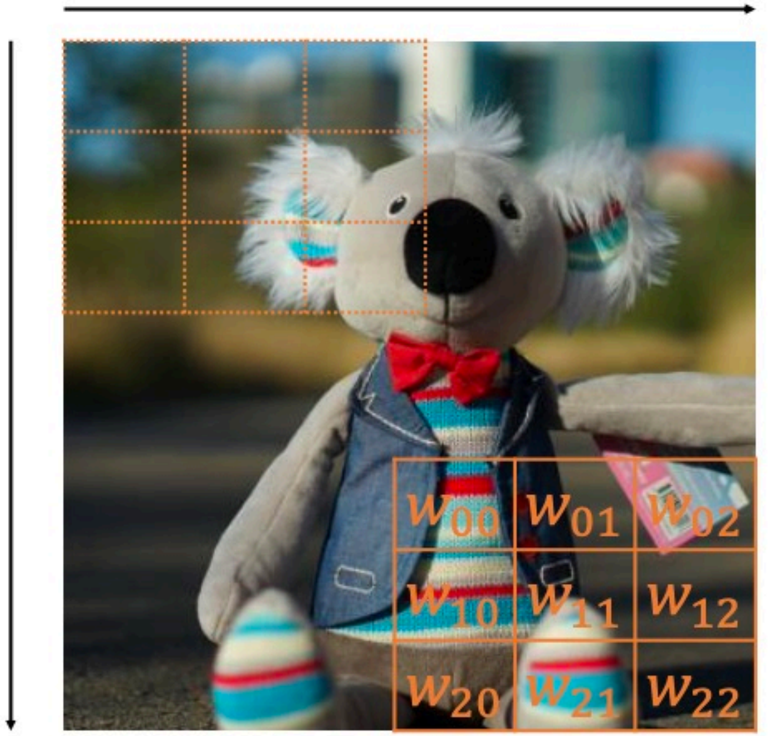
解决方法

（2）权值共享： 对于每个输出节点o，均使用相同的权值矩阵$\boldsymbol{W}$。

在计算左上角位置的输出像素时，使用权值矩阵

$$\boldsymbol{W}=\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

在计算右下方感受野区域时，共享权值参数$\boldsymbol{W}$。



3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

卷积运算

信号处理领域，卷积的“卷”是指翻转平移操作，“积”是指积分运算，

1D 连续卷积:

$$(f \otimes g)(n) = \int_{-\infty}^{\infty} f(\tau) g(n-\tau) \mathrm{d}\tau$$

1D离散卷积:

$$(f \otimes g)(n) = \sum_{\tau=-\infty}^{\infty} f(\tau) g(n-\tau)$$

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

卷积运算

2D离散卷积：

2D图片函数

$$[f \otimes g](m,n) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i,j)\, g(m-i, n-j)$$

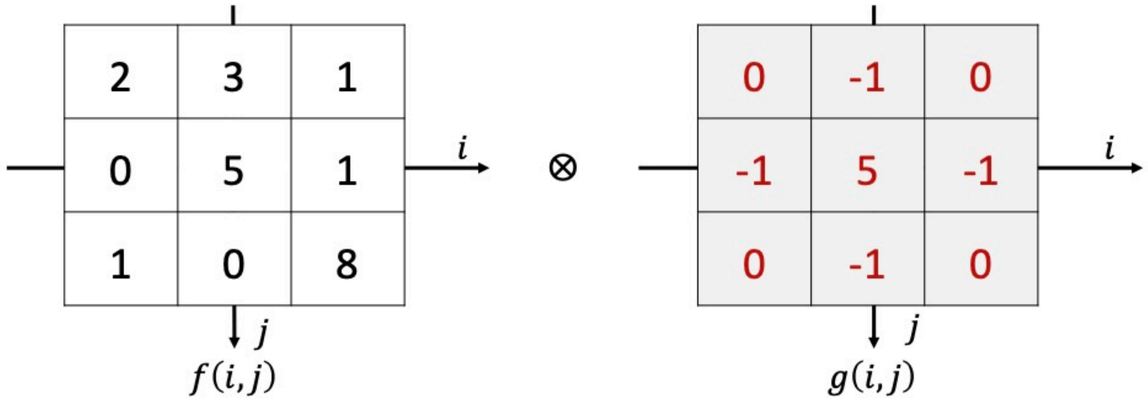
2D卷积核(即权值矩阵$\boldsymbol{W}$)

$f(i\,j)$ 和$g(i\,j)$仅在各自窗口有效区域存在值，其它区域视为0 。

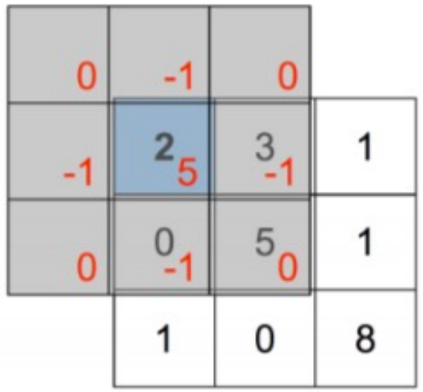
3 常见神经网络

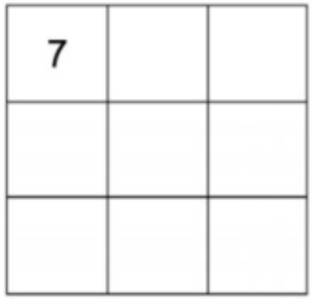
卷积神经网络（Conventional Neuron Network）：

卷积运算



$$[f \otimes g](-1,-1)$$





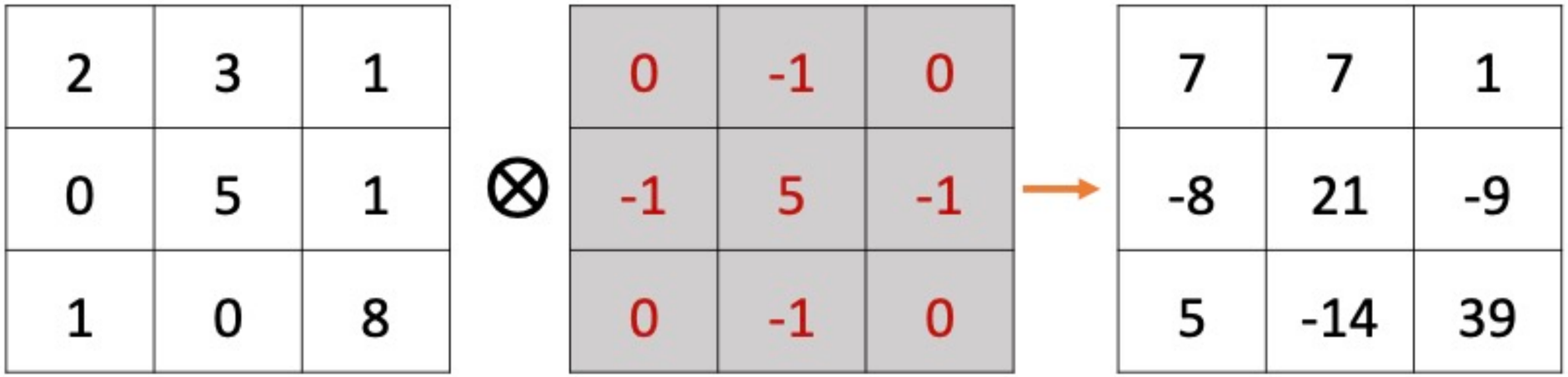
将卷积核$g(i\ j)$函数翻转(沿着x和y方向各翻转一次)

向左、向上各平移一个单元

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

按照此种方式循环计算，可以计算出函数$f \otimes g\ (m,n)$, -1<=m, n<=1，



始终使用$g(m,n)$函数完成卷积运算，卷积运算其实已经实现了权值共享的思想

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

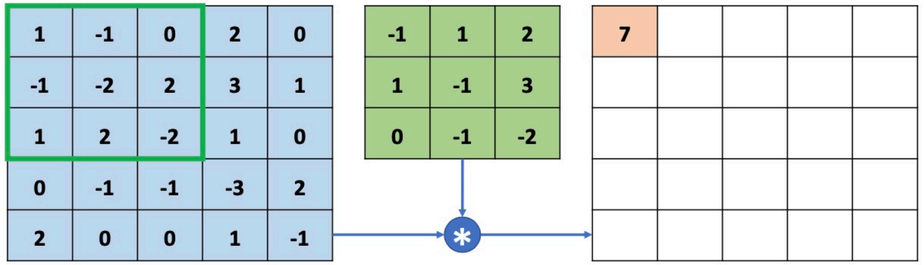
常见卷积核及其效果

$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&-1&0\\-1&5&-1\\0&-1&0\end{bmatrix}$	$\begin{bmatrix}0.0625&0.125&0.0625\\0.125&0.25&0.125\\0.0625&0.125&0.0625\end{bmatrix}$	$\begin{bmatrix}-1&-1&-1\\-1&8&-1\\-1&-1&-1\end{bmatrix}$
原图效果	锐化效果	模糊效果	边缘提取效果

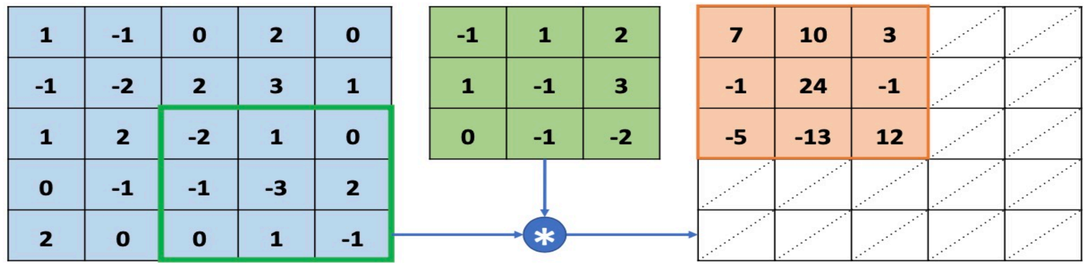
3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

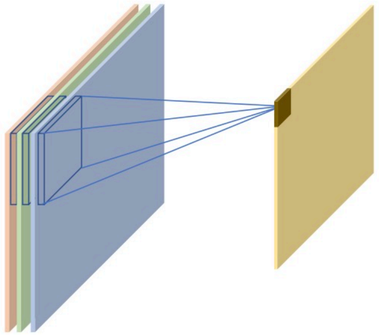
灰度图片，单个通道，单个卷积核



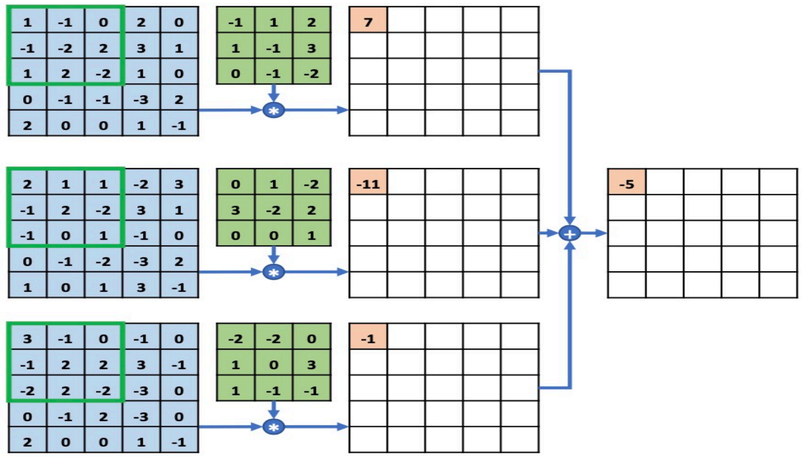
感受野窗口向下移动一个步长单位



彩色的图片，包含了 R/G/B 三个通道



多个通道，单个卷积核



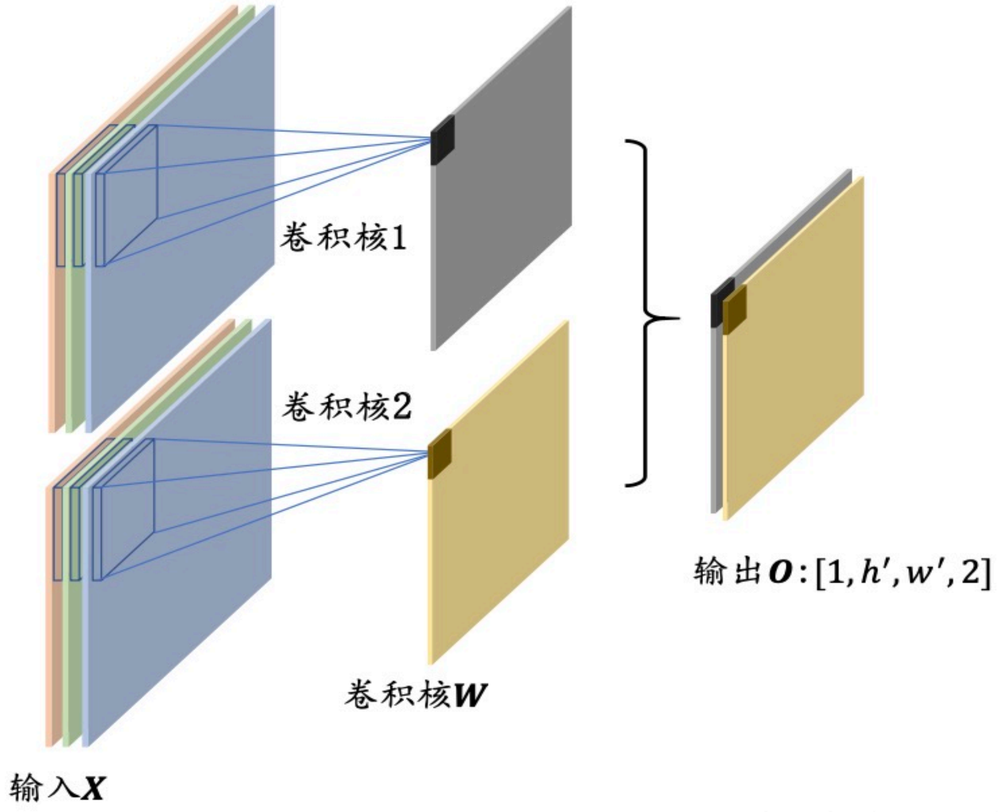
输入通道的通道数量决定了卷积核的通道数。一个卷积核只能得到一个输出矩阵，无论输入$\boldsymbol{X}$的通道数量。

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

多通道输入、多卷积核

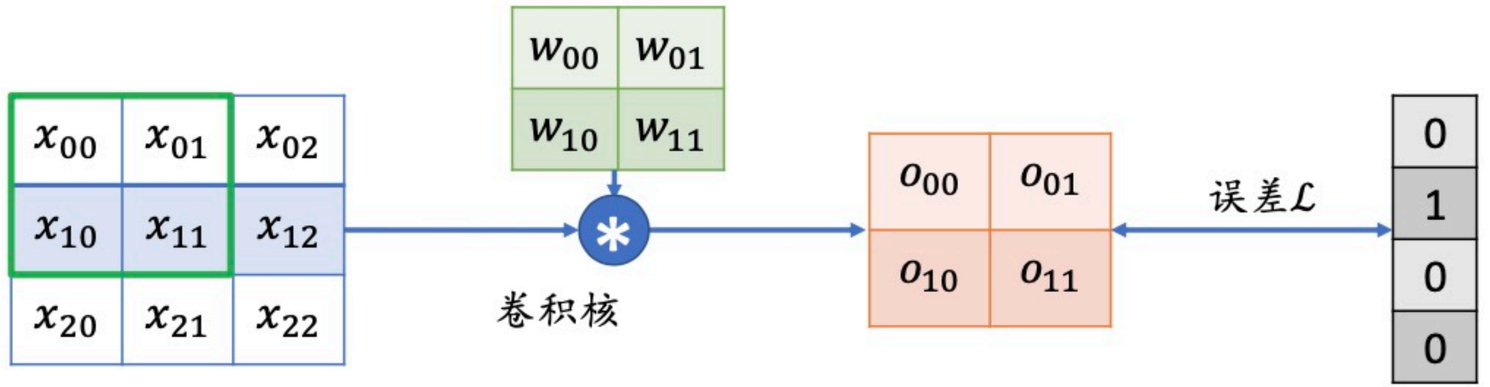
一个卷积核只能完成某种逻辑的特征提取，当需要同时提取多种逻辑特征，可通过增加多个卷积核来得到多种特征，提高神经网络的表达能力。



3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

梯度传播



首先推导出输出张量$\boldsymbol{O}$的表达形式:

$$o_{00} = x_{00}w_{00} + x_{01}w_{01} + x_{10}w_{10} + x_{11}w_{11} + b$$
$$o_{01} = x_{01}w_{00} + x_{02}w_{01} + x_{11}w_{10} + x_{12}w_{11} + b$$
$$o_{10} = x_{10}w_{00} + x_{11}w_{01} + x_{20}w_{10} + x_{21}w_{11} + b$$
$$o_{11} = x_{11}w_{00} + x_{12}w_{01} + x_{21}w_{10} + x_{22}w_{11} + b$$

$$\frac{\partial \mathcal{L}}{\partial w_{00}} = \sum_{i \in \{00,01,10,11\}} \frac{\partial \mathcal{L}}{\partial o_i} \frac{\partial o_i}{\partial w_{00}}$$

循环移动感受野的方式并没有改变网络层可导性

3 常见神经网络

卷积神经网络（Conventional Neuron Network）：

网络优化

步长值、卷积核大小等

卷积层变种

空洞卷积、转置卷积等

3 常见神经网络

循环神经网络（Recurrent Neuron Network）：

序列问题：文本识别、自然语言等

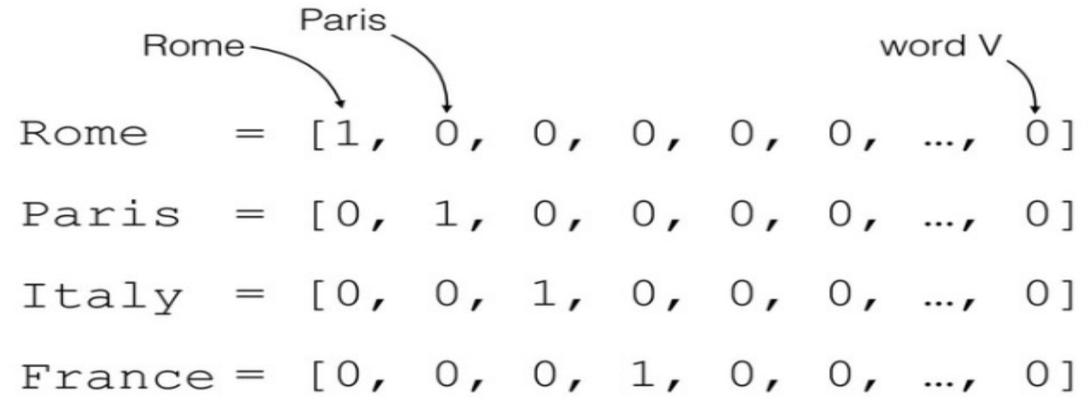
序列信号：需要一个 shape 为$[b, s]$的张量， 其中b 为序列数量，s为序列长度。

$$\left[\left[x_1^{(1)}, x_2^{(1)}, \cdots, x_6^{(1)}\right], \left[x_1^{(2)}, x_2^{(2)}, \cdots, x_6^{(2)}\right], \cdots, \left[x_1^{(b)}, x_2^{(b)}, \cdots, x_6^{(b)}\right]\right]$$

将单词或字符转化为数值，Word Embedding

One-hot (独热编码)

数据稀疏，且无相关性

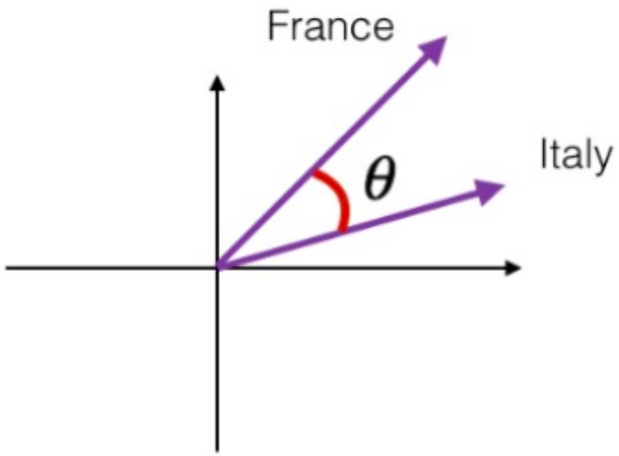


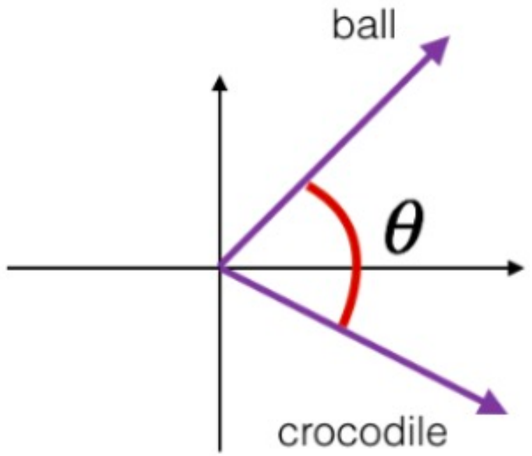
3 常见神经网络

循环神经网络（Recurrent Neuron Network）：

语义层面的相关性：余弦相关度

$$\text{similarity}(\boldsymbol{a},\boldsymbol{b}) \triangleq \cos(\theta) = \frac{\boldsymbol{a}\cdot\boldsymbol{b}}{|\boldsymbol{a}|\cdot|\boldsymbol{b}|}$$



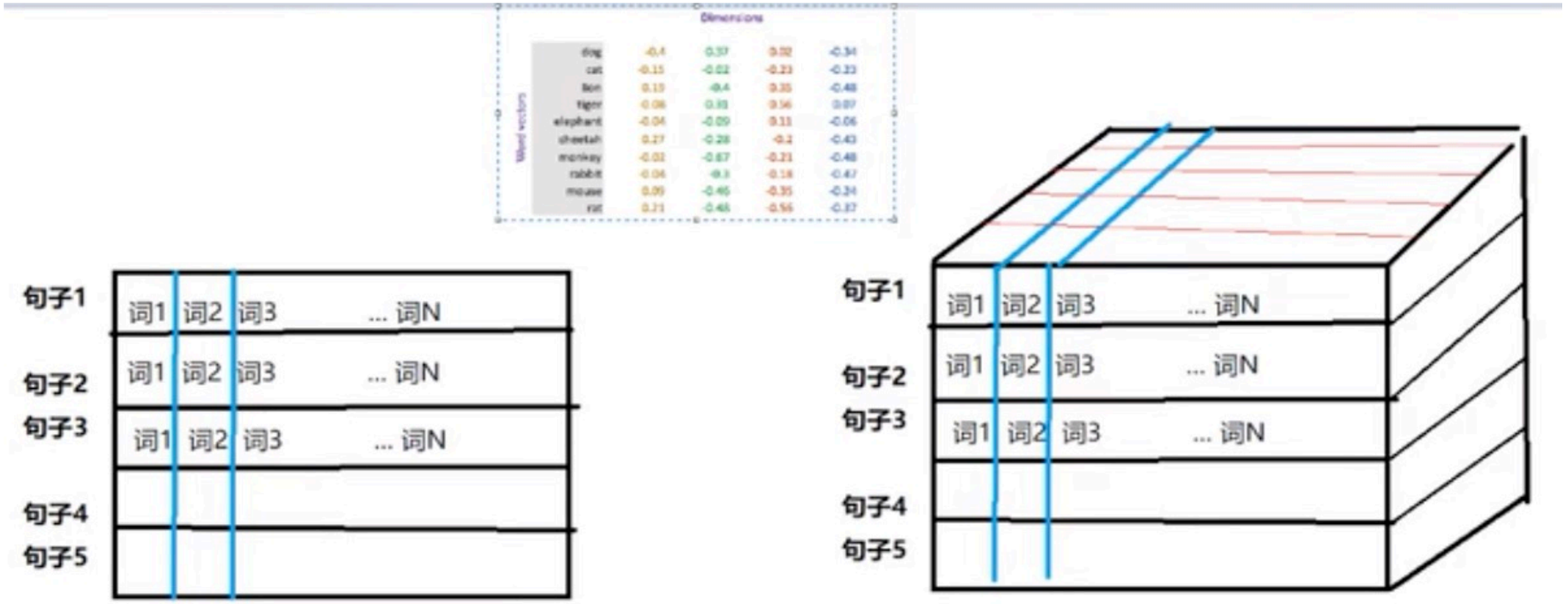


θ为两个词向量之间的夹角

3 常见神经网络

循环神经网络（Recurrent Neuron Network）：

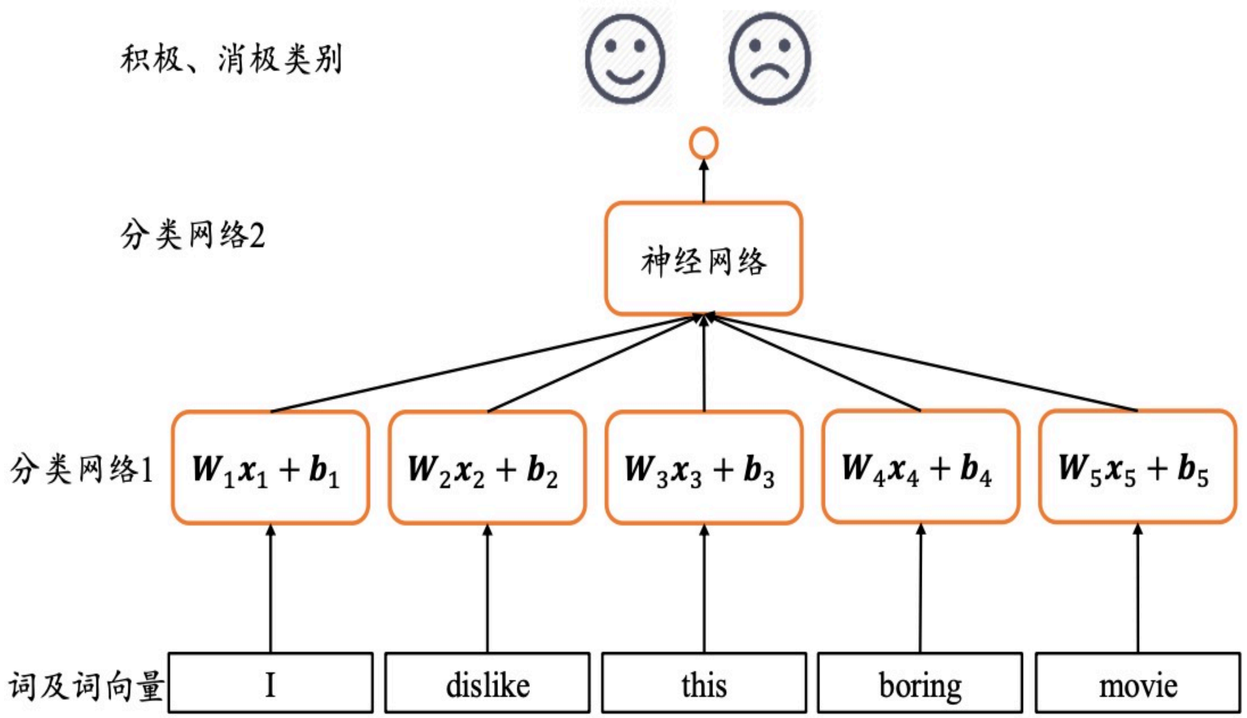
Embedding层：单词的表示层，把单词编码为某个词向量。



3 常见神经网络

循环神经网络（Recurrent Neuron Network）：

网络实现



全连接

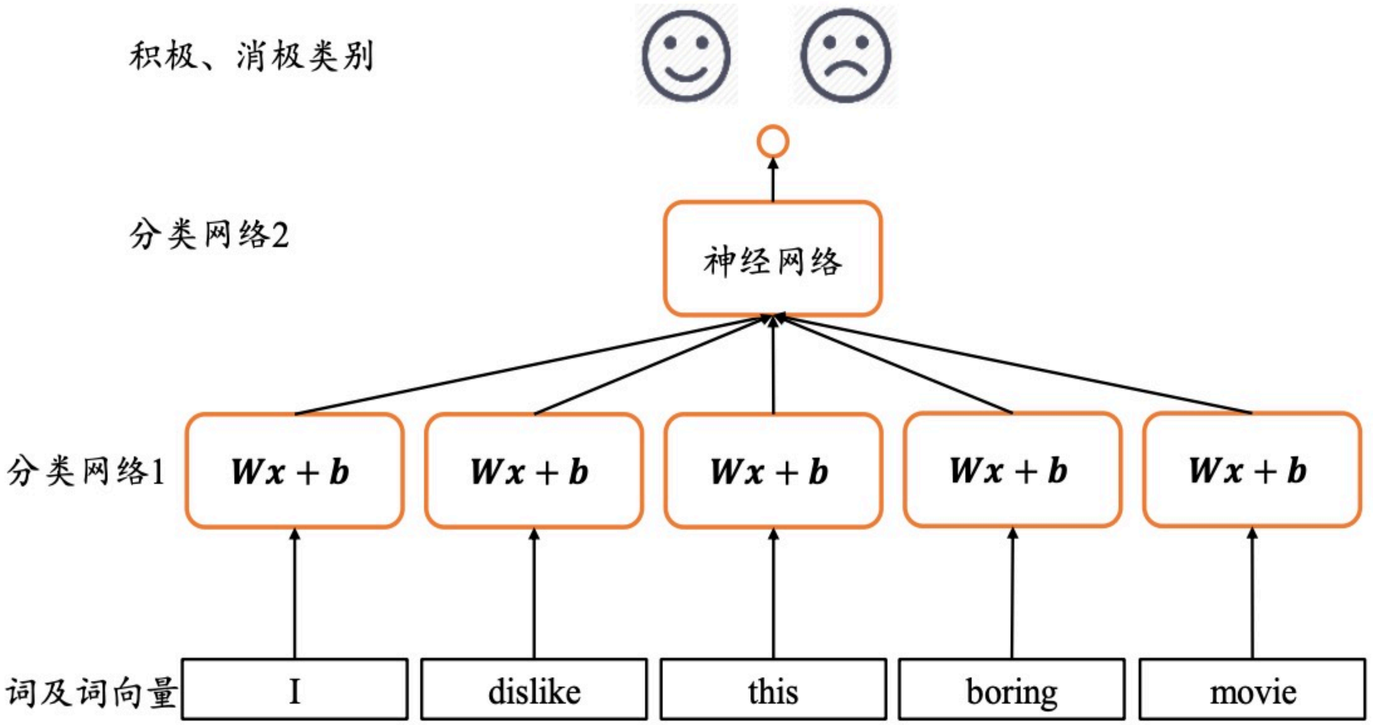
1、网络参量相当可观

2、只能感受当前词向量的输入

3 常见神经网络

循环神经网络（Recurrent Neuron Network）：

网络实现

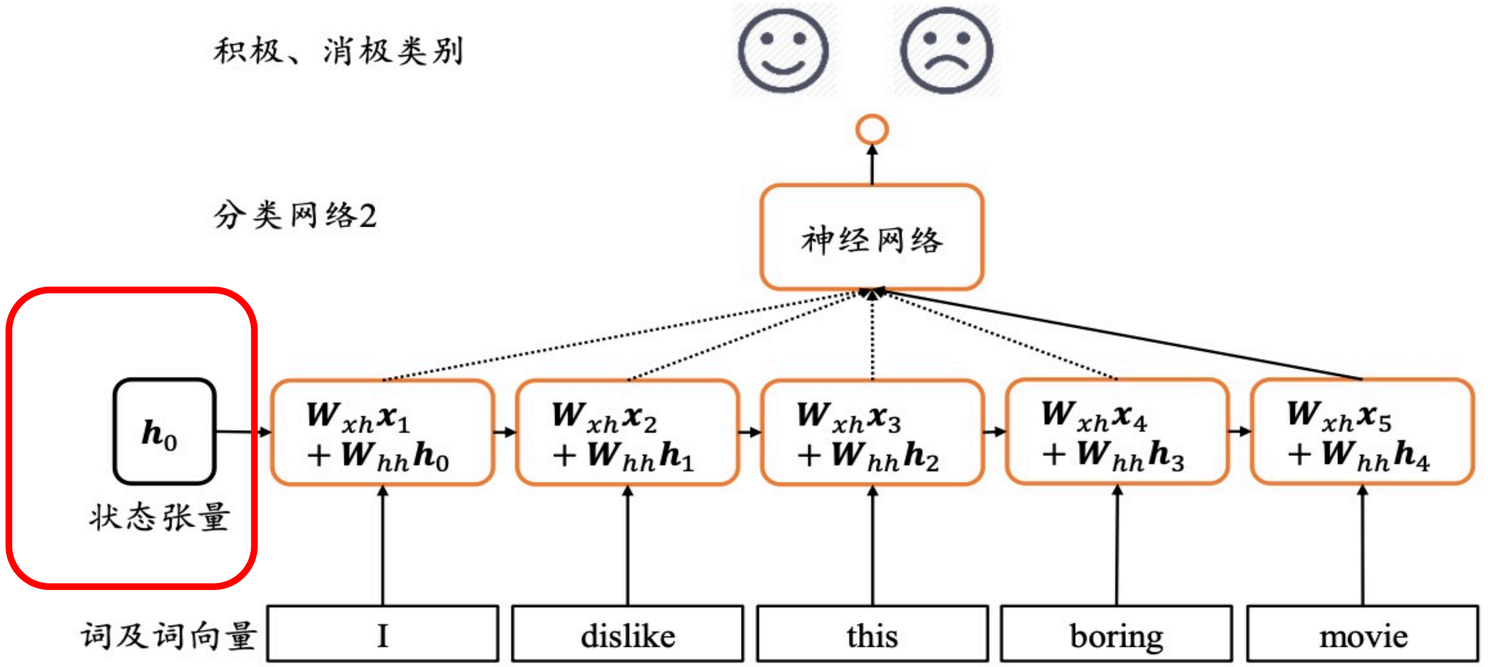


权值共享

3 常见神经网络

循环神经网络（Recurrent Neuron Network）：

网络实现



全局语义

3 常见神经网络

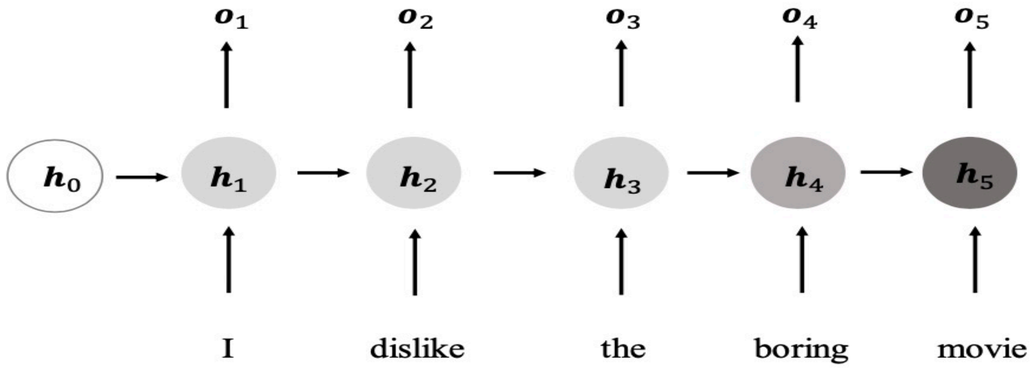
循环神经网络（Recurrent Neuron Network）：

网络实现

在每个时间戳t，网络层接受当前时间戳的输入$\boldsymbol{x}(t)$和上一个时间戳的网络状态向量$h(t-1)$，经过

$$\boldsymbol{h}_t = f_\theta(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$$

变换后得到当前时间戳的新状态向量，并写入记忆状态中，在每个时间戳上，网络层均有输出产生$\boldsymbol{o}(t)$，$\boldsymbol{o}(t) = g(\mathrm{h}t)$，即将网络的状态向量变换后输出。



全局语义

3 常见神经网络

循环神经网络（Recurrent Neuron Network）：

梯度传播

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_i} = \prod_{j=i}^{t-1} diag\left(\sigma'\left(\boldsymbol{W}_{xh}\boldsymbol{x}_{j+1} + \boldsymbol{W}_{hh}\boldsymbol{h}_j + \boldsymbol{b}\right)\right)\boldsymbol{W}_{hh}$$

梯度弥散或者爆炸

链式法则展开

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_{hh}} = \sum_{i=1}^{t} \frac{\partial \mathcal{L}}{\partial \boldsymbol{o}_t}\frac{\partial \boldsymbol{o}_t}{\partial \boldsymbol{h}_t}\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_i}\frac{\partial^+ \boldsymbol{h}_i}{\partial \boldsymbol{W}_{hh}}$$

$$\frac{\partial^+ \boldsymbol{h}_i}{\partial \boldsymbol{W}_{hh}} = \frac{\partial \sigma(\boldsymbol{W}_{xh}\boldsymbol{x}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{b})}{\partial \boldsymbol{W}_{hh}}$$

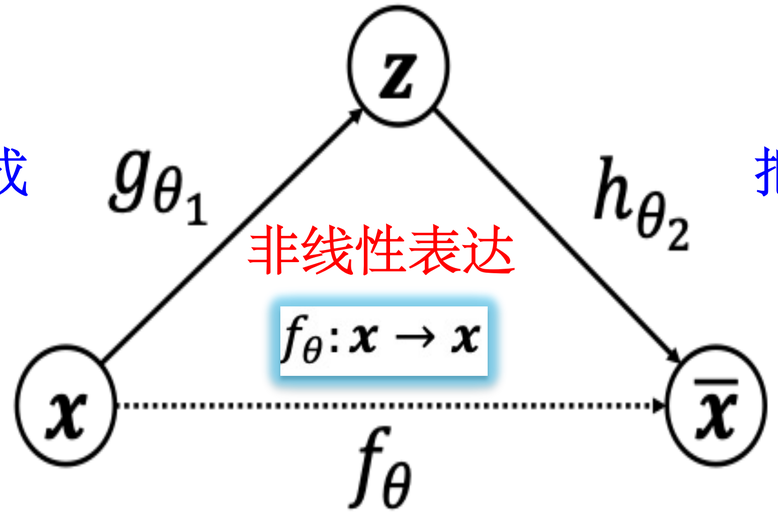
短时记忆

3 常见神经网络

自编码器(Auto-Encoder)：

一种把样本x作为监督信号来学习的神经网路（自监督学习）

把高维度的输入x编码成
低维度的隐变量z

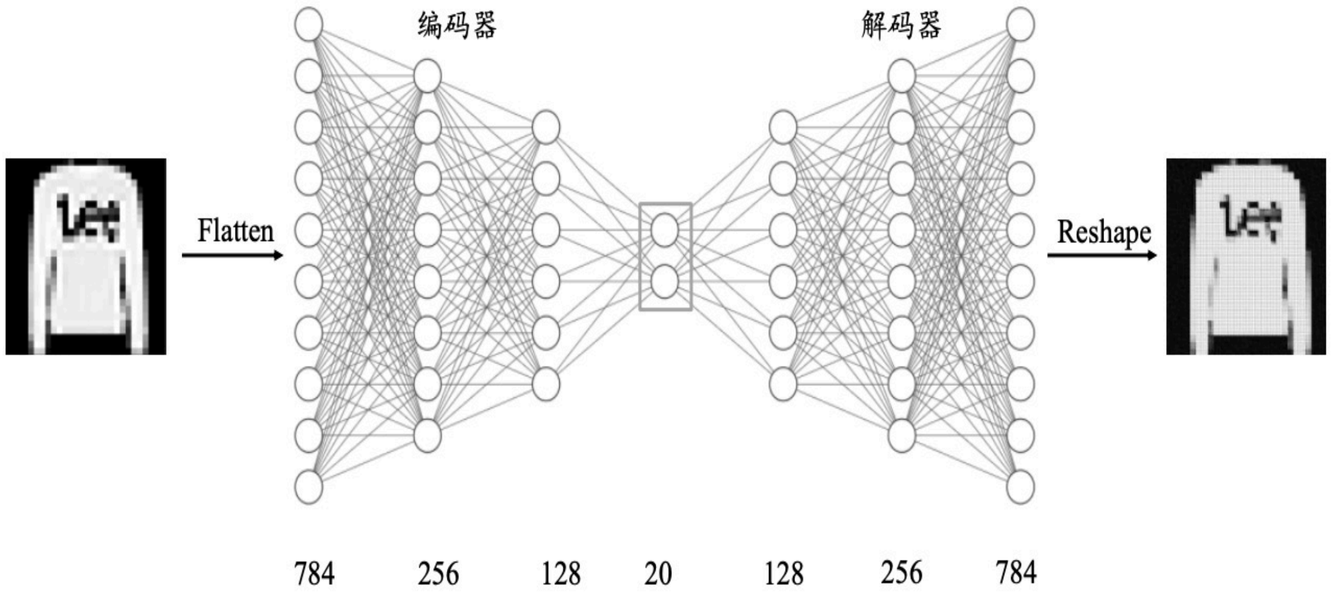


把编码过后的输入z解码为
高维度的x

整个网络模型f叫做自动编码器(Auto-Encoder)

3 常见神经网络

自编码器(Auto-Encoder)：



优化目标：

$$\text{Minimize } \mathcal{L} = \text{dist}(\boldsymbol{x}, \bar{\boldsymbol{x}})$$

$$\bar{\boldsymbol{x}} = h_{\theta_2}(g_{\theta_1}(\boldsymbol{x}))$$

自编码器的训练过程与分类器的基本一致，通过误差函数计算出重建向量$\boldsymbol{x}$与原始输入向量$\boldsymbol{x}$之间的距离，再利用自动求导机制同时求出encoder和decoder的梯度，循环更新即可。

3 常见神经网络

生成对抗神经网络（Generative Adversarial Network）：

GAN网络借鉴了博弈学习的思想

分别设立了两个子网络:负责生成样本的生成器G和负责鉴别真伪的鉴别器D。

老朝奉

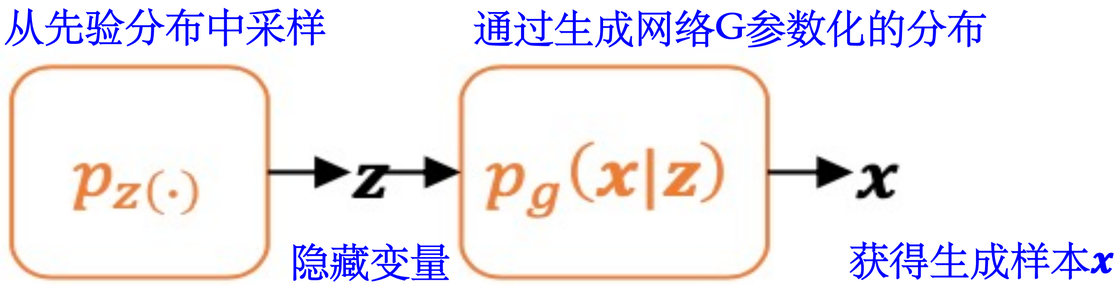
许愿

3 常见神经网络

生成对抗神经网络（Generative Adversarial Network）：

生成网络：类似解码器

判别网络：二分类网络

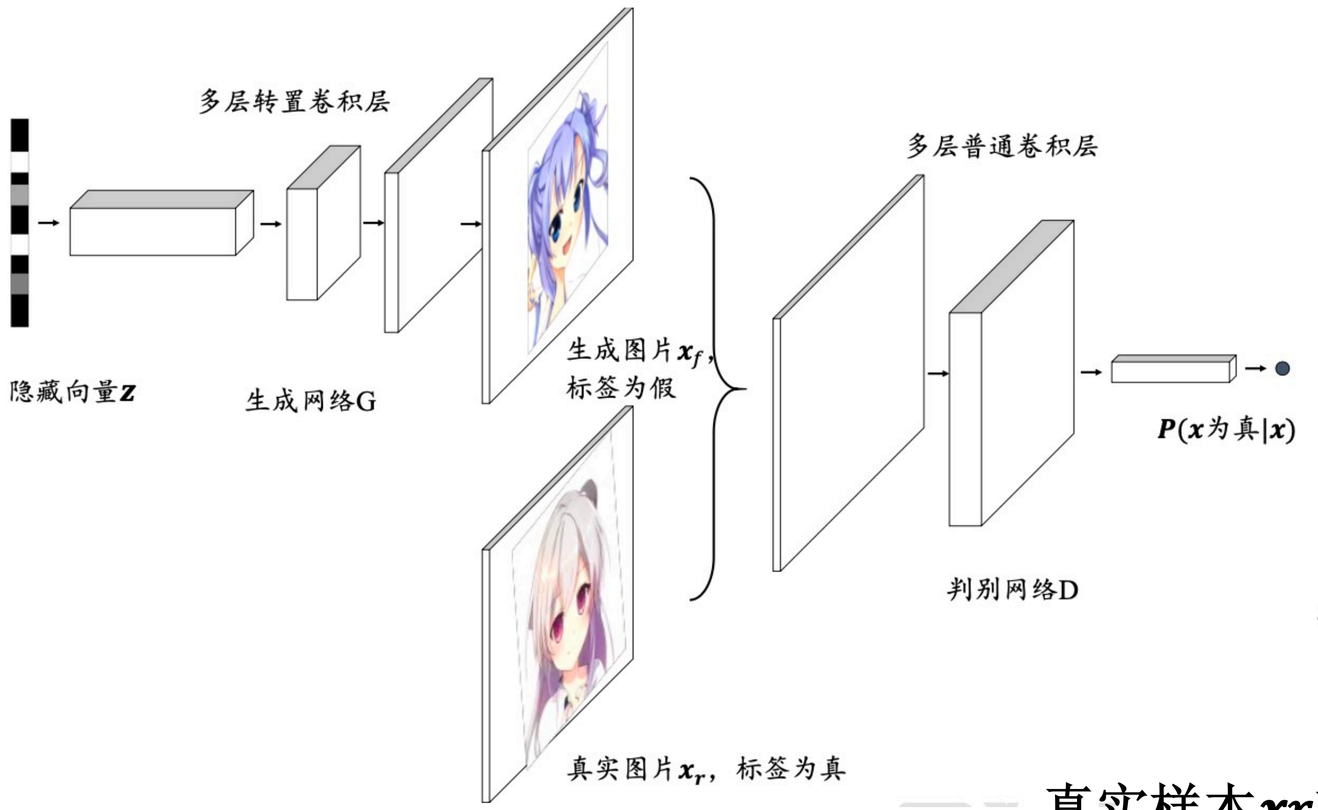


生成器G的功能是将隐向量z通过神经网络转换为样本向量x

接受输入样本x的数据集，包含采样自真实数据分布，也包含了采样自生成网络的假样本，共同组成了判别网络的训练数据集。

3 常见神经网络

生成对抗神经网络（Generative Adversarial Network）：



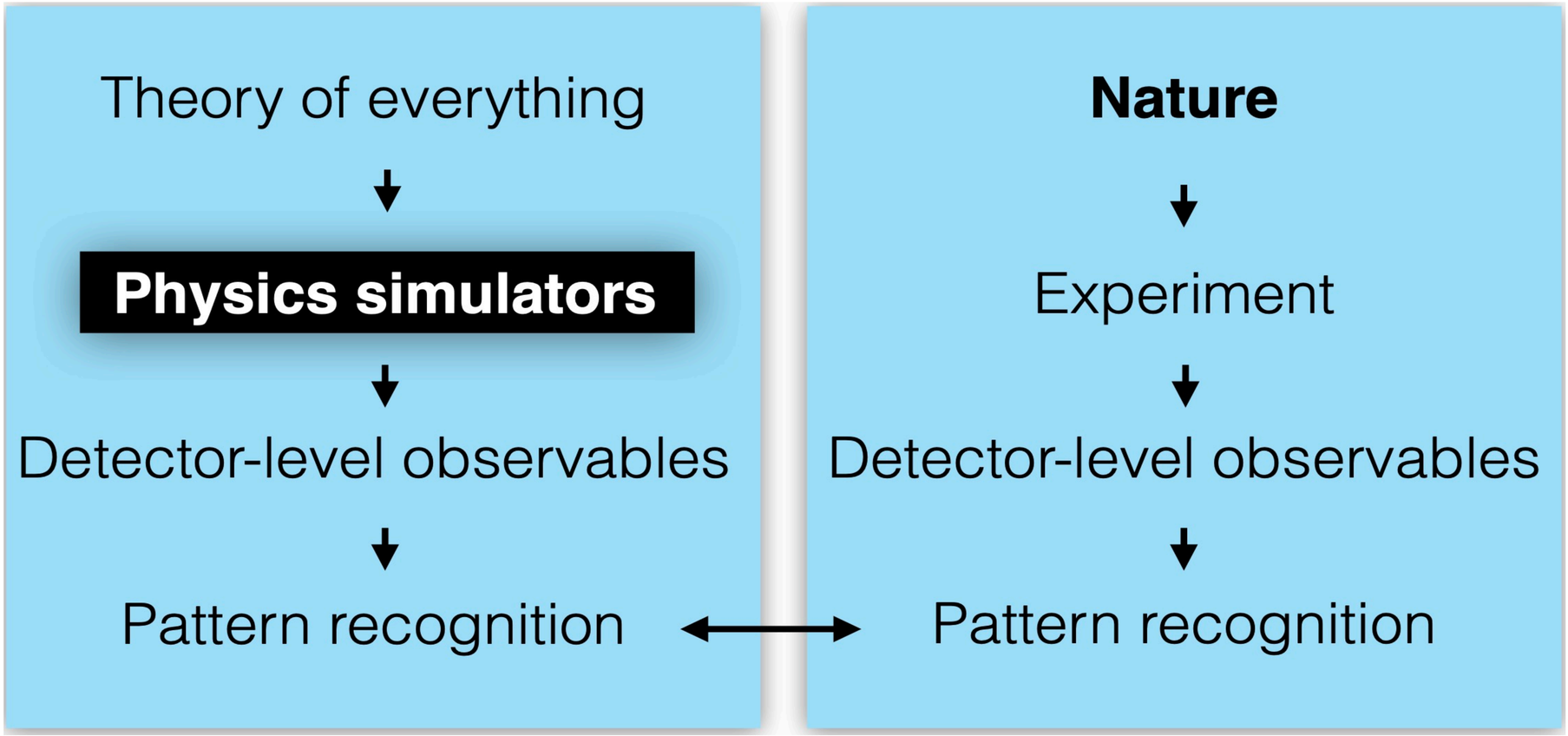
通过最小化判别网络**D**的预测值与标签之间的误差来优化判别网络参数。

$$\min_{\theta} \mathcal{L} = \mathrm{CE}\left(D_\theta(\boldsymbol{x}_r), y_r, D_\theta(\boldsymbol{x}_f), y_f\right)$$

$$\mathcal{L} = -\sum_{\boldsymbol{x}_r \sim p_r(\cdot)} \log D_\theta(\boldsymbol{x}_r) - \sum_{\boldsymbol{x}_f \sim p_g(\cdot)} \log\left(1 - D_\theta(\boldsymbol{x}_f)\right)$$

$\boldsymbol{x}_r$ 真实样本$\boldsymbol{x}r$在判别网络D的输出，θ为判别网络的参数集

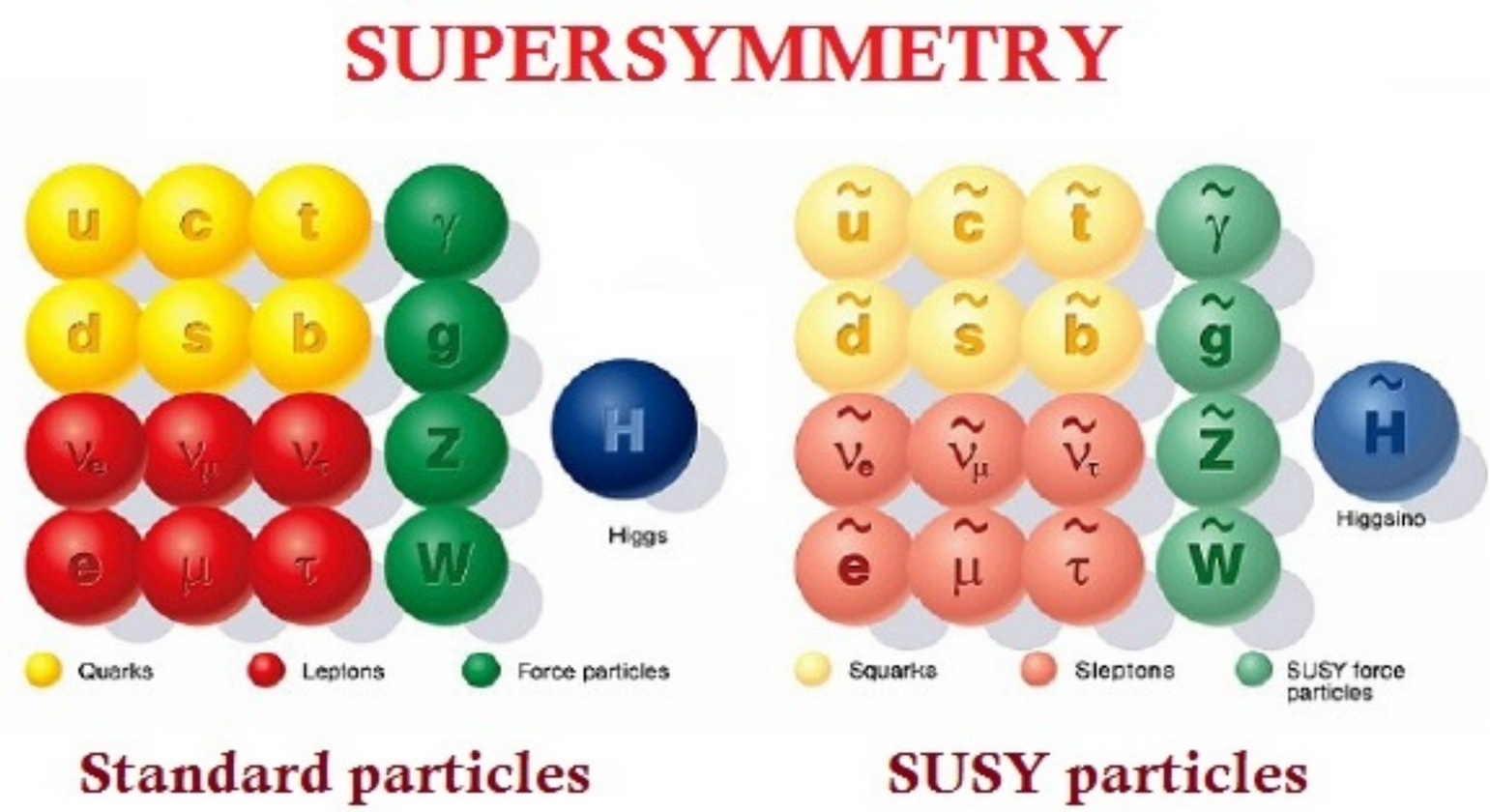
4 新物理物理研究中的ML



1、寻找新信号

2、检验新模型

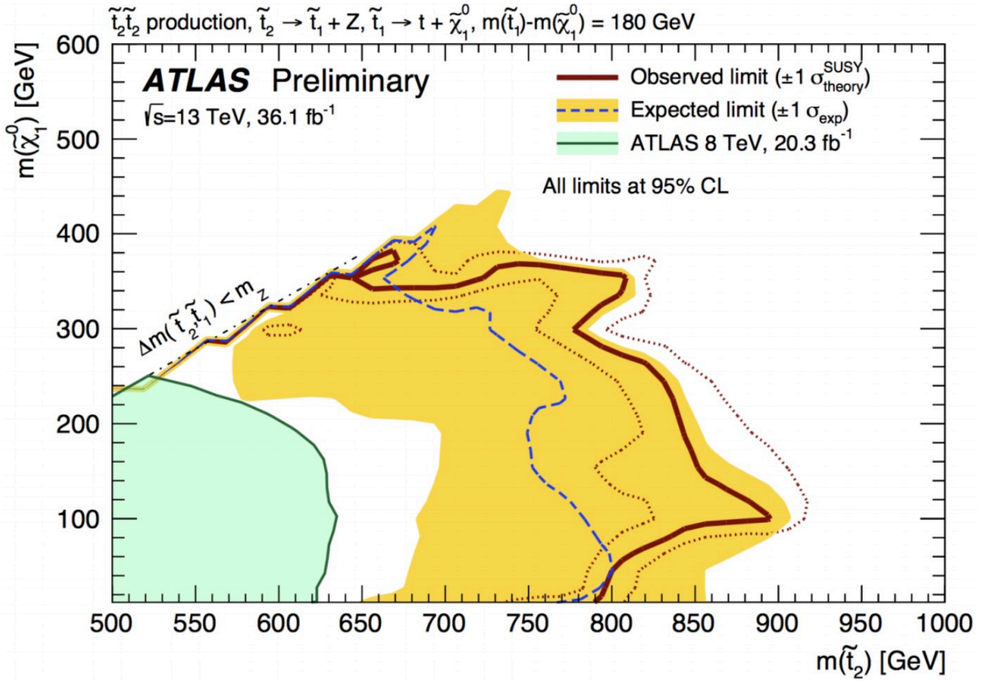
4 新物理物理研究中的ML

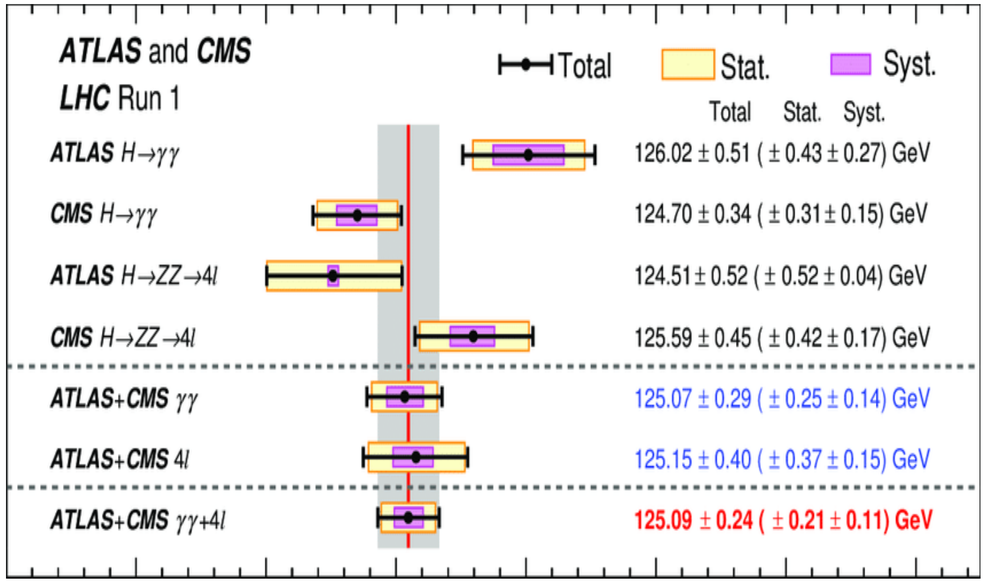


Extenson of space-time symmetry

4 新物理物理研究中的ML

检验新模型: Machine Learning Scan

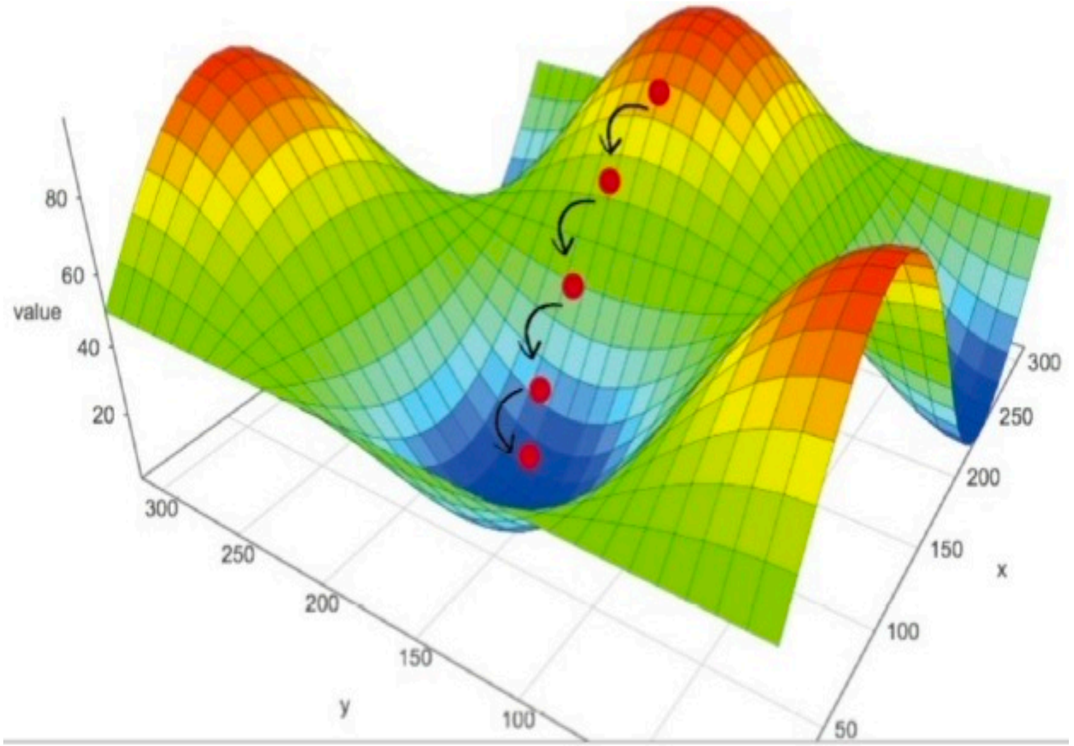




4 新物理物理研究中的ML

检验新模型: Machine Learning Scan

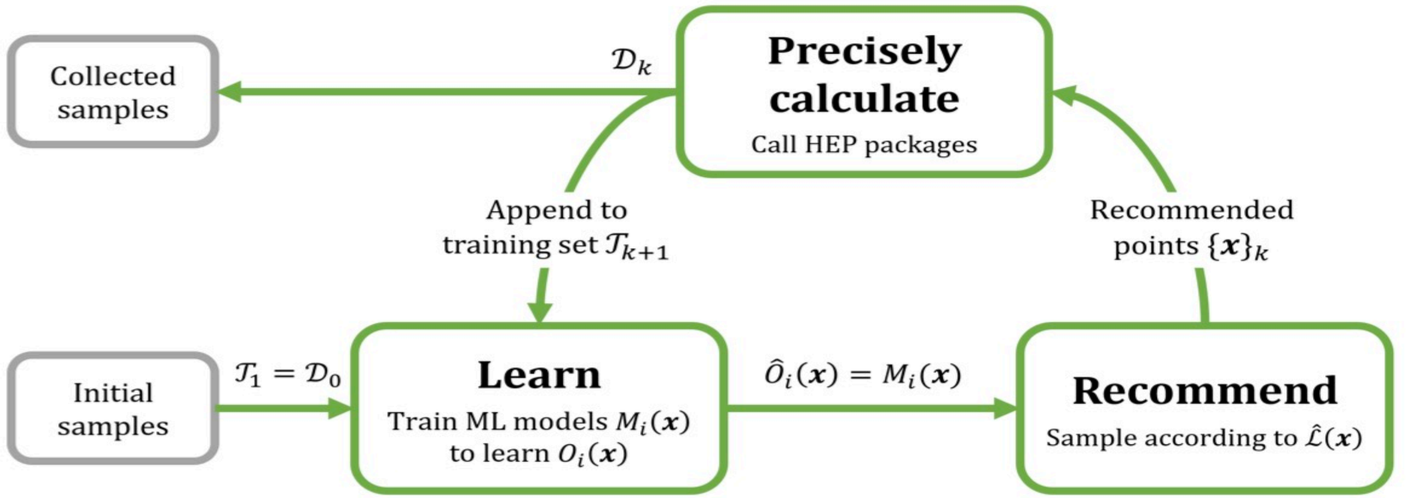
$$\mathcal{L}(\boldsymbol{x}) = \prod_i \mathcal{L}_i(O_i(\boldsymbol{x}); O_i^*, \sigma_i^*)$$



4 新物理物理研究中的ML

检验新模型: Machine Learning Scan

Ren, Wu, Yang, Zhao, Nucl.Phys.B 943 (2019) 114613



1、使用已收集的数据训练ML模型；

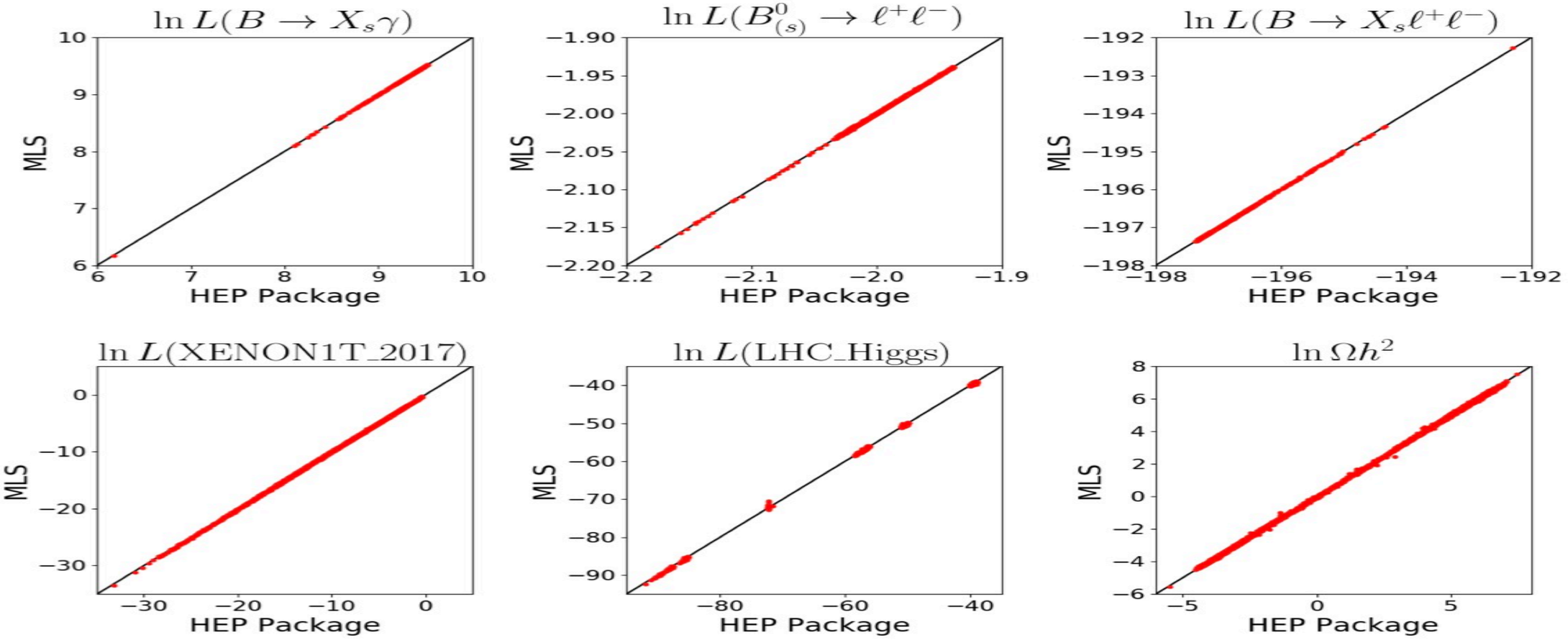
2、根据重建的likelihood区间重点抽样，并产生推荐数据；

3、利用程序包精确计算推荐数据的可观测量，将符合好的样本和新的随机样本重新加入训练集；

4、重复以上步骤，收集符合条件数据。

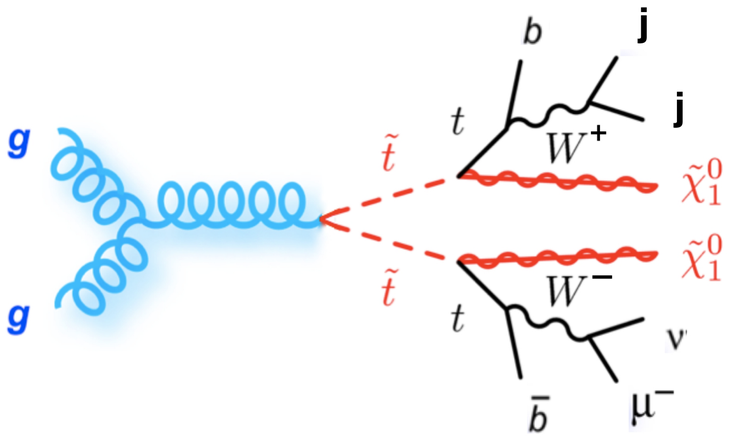
4 新物理物理研究中的ML

检验新模型: Machine Learning Scan

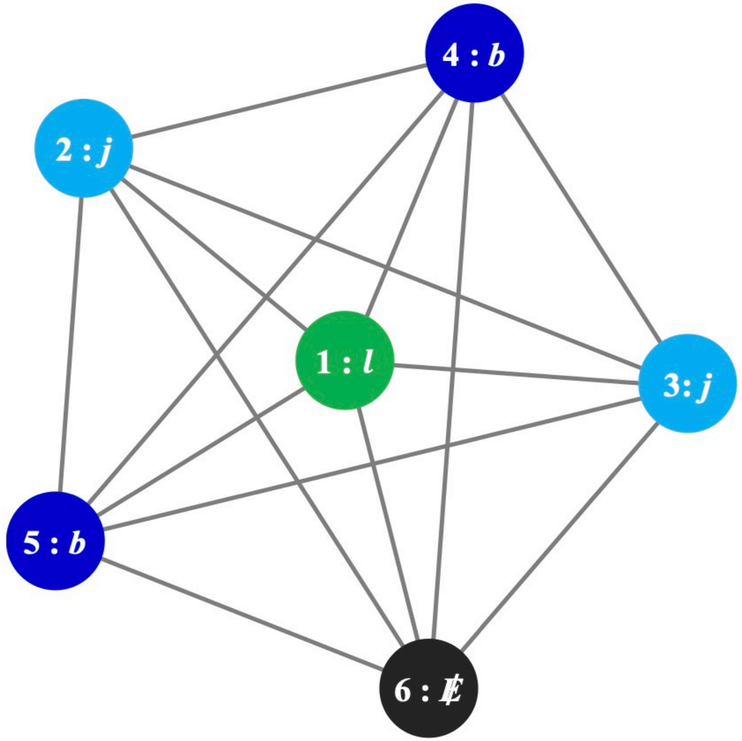


4 新物理物理研究中的ML

寻找新信号：事例图



top对产生过程



x

	Photon	lepton charge	b-jet or light jet	MET	p_T (TeV)	E (TeV)	m (TeV)	
$x_1 = ($	0	-1	0	0	0.0229	0.0289	0.0000	)
$x_2 = ($	0	0	-1	0	0.2637	0.3304	0.0373	)
$x_3 = ($	0	0	-1	0	0.1003	0.1888	0.0091	)
$x_4 = ($	0	0	1	0	0.0980	0.1146	0.0133	)
$x_5 = ($	0	0	1	0	0.0689	0.0773	0.0062	)
$x_6 = ($	0	0	0	1	0.2107	0.2107	0.0000	)

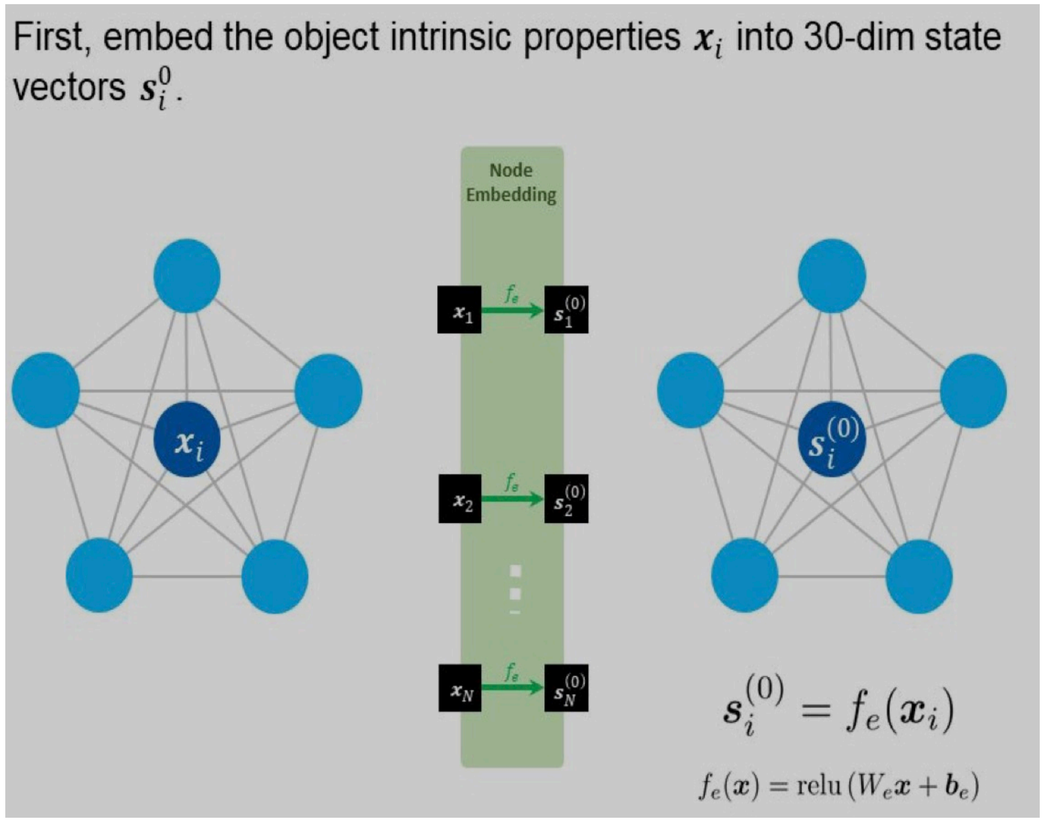
d

	1	2	3	4	5	6
1	0	1.3971	2.5649	1.2801	3.2752	3.0312
2	1.3971	0	1.9019	1.6688	3.0871	3.1717
3	2.5649	1.9019	0	3.4440	1.5805	1.7831
4	1.2801	1.6688	3.4440	0	2.2175	2.1387
5	3.2752	3.0871	1.5805	2.2175	0	0.4912
6	3.0312	3.1717	1.7831	2.1387	0.4912	0

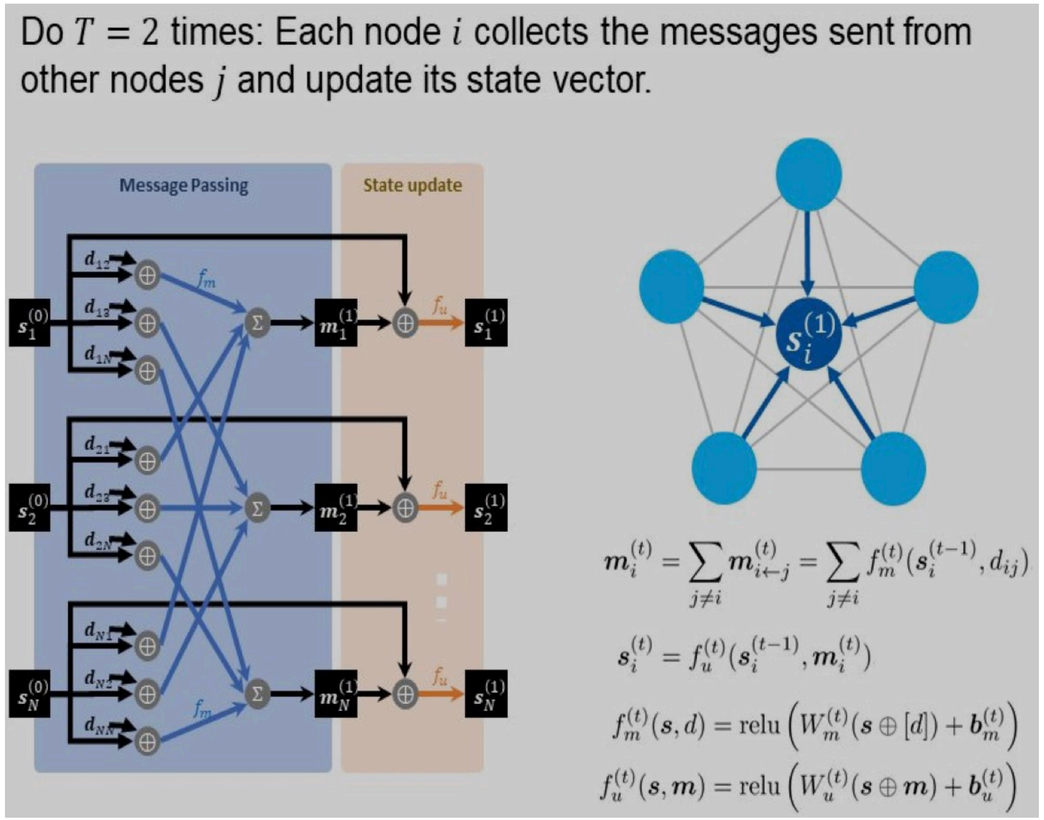
4 新物理物理研究中的ML

寻找新信号：事例图

仅包含编号为*i*的结点特征向量**xi**中的信息，
没有整张图的几何模式的任何信息。



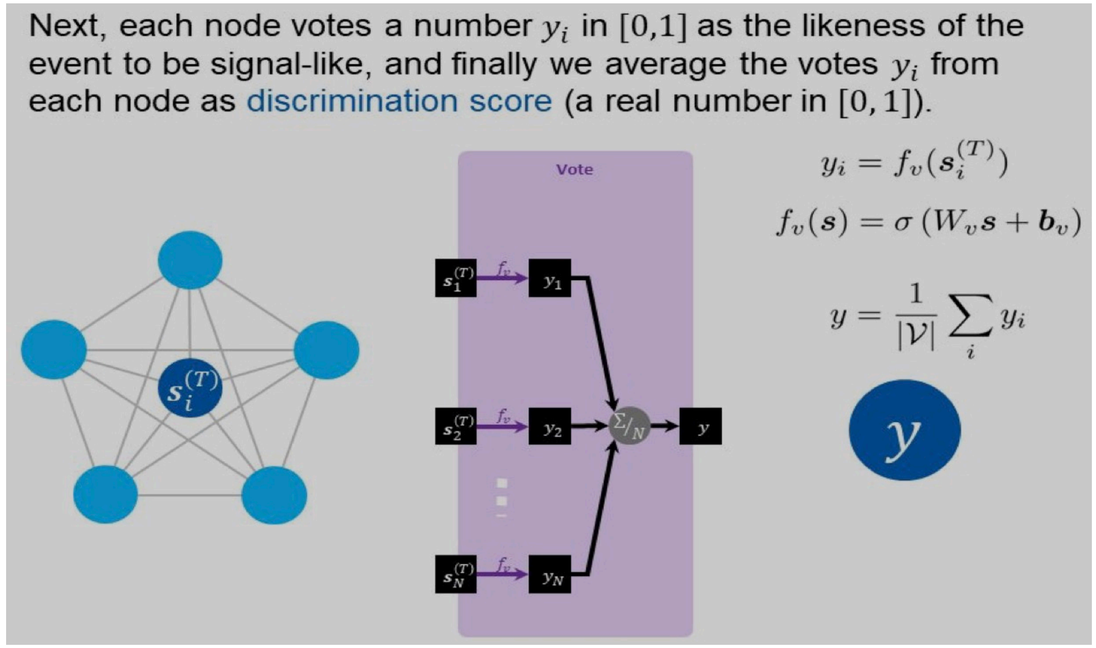
经过*T* 次迭代之后，
每一个结点的状态都是整张图的一个编码。

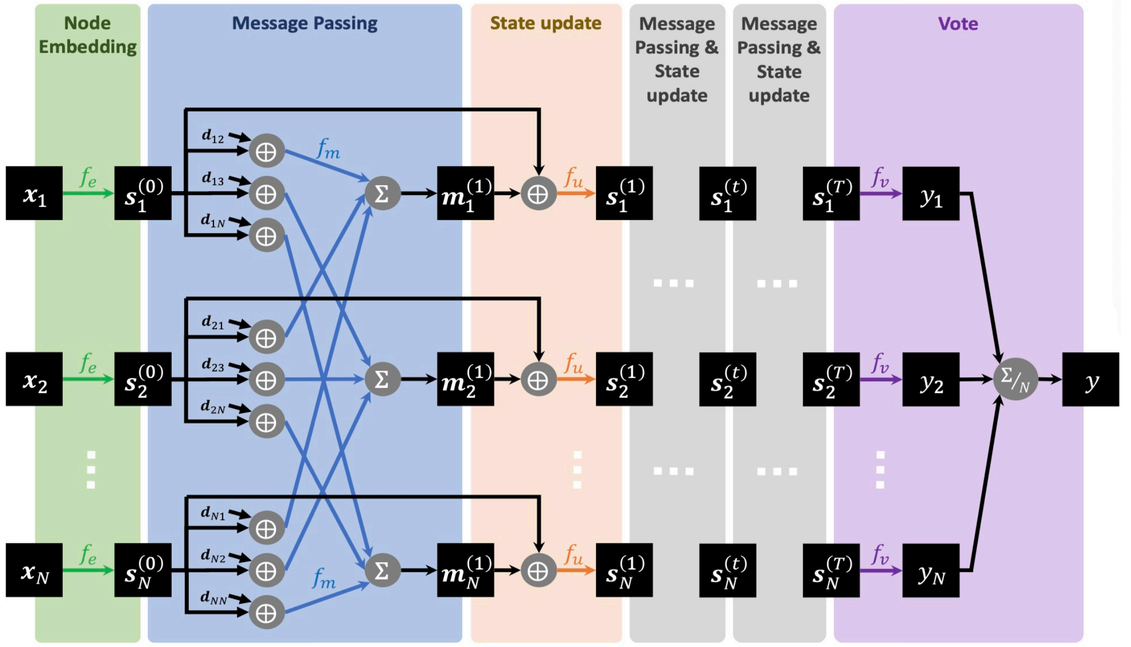


4 新物理物理研究中的ML

寻找新信号：事例图

每一个结点根据它的状态向量给出一个投票，
这个分数表示该结点认为的输入的事件图是信号的概率

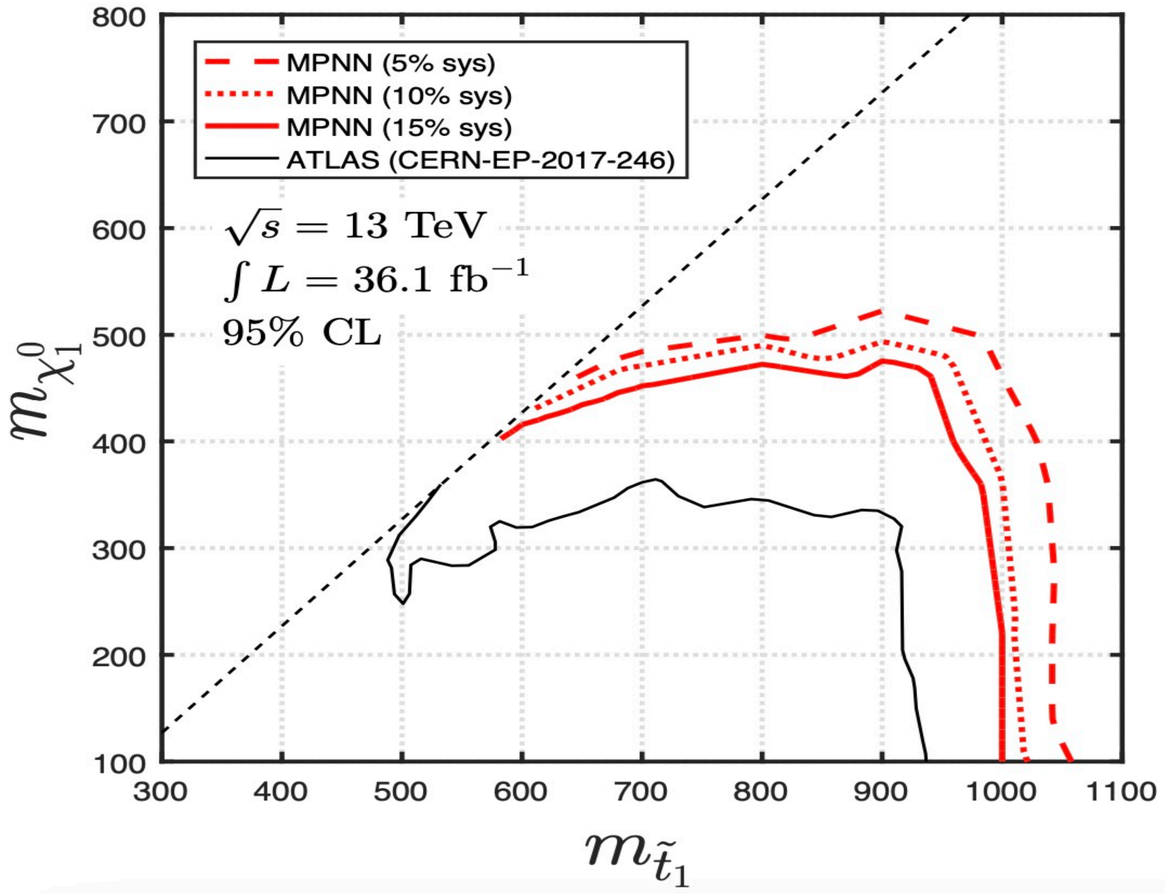




这种编码是所有末态的运动学特征以及它们之间的几何关系的完整信息的一种紧凑表示。它们可以看作是由神经网络自身从输入的事件图中自动提取到的事件特征。

4 新物理物理研究中的ML

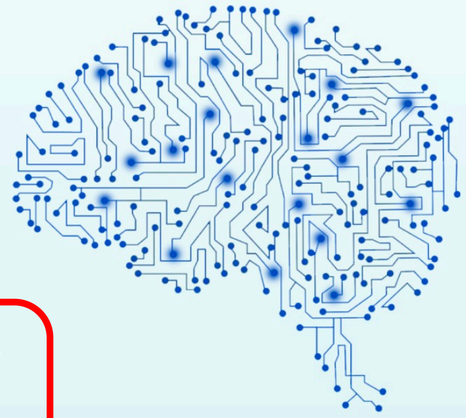
寻找新信号：事例图



4 新物理物理研究中的ML

The structure of the proton at the LHC

Higgs self-interactions



QCD-aware NNs for jet physics

Automated bSM exclusion limits

Boosting bSM searches

HEP detector simulation

Thank you!