逻辑回归和神经网络

Teacher：张明卫
Email：zhangmw@swc.neu.edu
Office: 东北大学浑南校区信息楼B434

CONTENTS

PART 01 逻辑回归

1.1 回归分析简介

- 回归，最初是遗传学中的一个名词，是由生物学家兼统计学家高尔顿首先提出来的。他在研究人类的身高时，发现高个子父母的孩子回归于人口的平均身高，而矮个子父母的孩子则从另一个方向回归于人口的平均身高。
- 预测是我们日常生活中离不开的活动，因而在机器学习中借助于历史数据构建高精确度的预测模型是一项基本任务。如果预测结果是离散的分类标号，我们称之为分类，如果是连续值则称之为预测，因而分类与预测是预测技术的两种类型。
- 回归分析是一种预测性的建模技术，研究自变量和因变量之间数量变化关系的一种分析方法。它主要是通过建立因变量 Y 与影响它的自变量 X 之间的回归模型，衡量自变量 X 对因变量 Y 的影响能力，进而可以预测因变量Y的发展趋势。

1.1 回归分析简介

- 回归分析方法的分类：
  - 按照自变量的多少，回归分析分为一元回归分析和多元回归分析；
  - 按照自变量和因变量之间的关系类型，可分为线性回归分析和非线性回归分析。

1.2 线性回归分析

- 训练数据集： $D = \{< X^{(i)}, y^{(i)} >\} (1 \le i \le m)$，即**D**中有**m**个实例（或称之为点）。$X = \{x_1, x_2, \dots, x_n\}$，即每个实例中包含**n**个自变量。
- 线性回归假设数据特征间满足线性关系，即根据训练数据集**D**，构建一个自变量的线性组合模型，并用此模型进行因变量的预测。其函数形式为：

$$h(X) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n \quad (1)$$

令x_0=1，即$X = \{1, x_1, x_2, \dots, x_n\}$，$W = \{w_0, w_1, w_2, \dots, w_n\}$，该线性函数可表示为：

$$h_w(\mathrm{X}) = W^T X \quad (2)$$

- 因而确定线性回归模型，就是确定权重系数向量**W**。

1.2 线性回归分析

■ 线性回归模型的训练：

模型根据什么进行训练呢？定义一个合理的目标函数（或称之为损失函数），使得损失函数取值最小。线性回归模型所用的损失函数一般是平方误差，即：

$$J(\mathrm{W}) = \min_{W} \sum_{i=1}^{m} (h_w(\mathrm{X}^{(i)}) - y^{(i)})^2 \quad (3)$$

基于最小二乘法，按下面的公式求解即可得W的值。

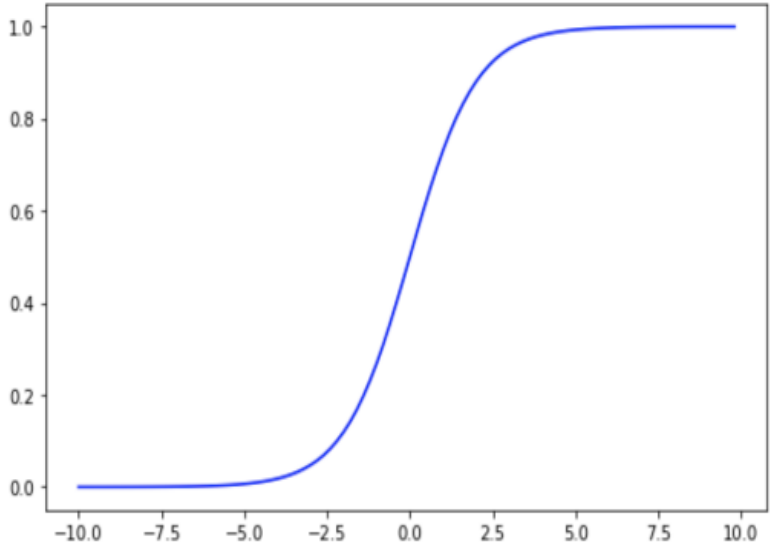
$$\mathrm{W} = (\mathrm{X}^T\mathrm{X})^{-1}X^TY \quad (4)$$

1.3 逻辑回归分析

- 逻辑回归(Logistic regression)为概率型非线性回归模型，是研究二分类观察结果与一些影响因素之间关系的一种多变量分析方法。通常的问题是，研究某些因素条件下某个结果是否发生，比如医学中根据病人的一些症状来判断它是否患有某种病。
- Logistic回归是一种广义线性回归，是在所建立的线性模型的基础上在进行Logistic函数的处理。Logistic函数是最典型的Sigmoid函数，其公式如下：

$$h(z) = \frac{1}{1+e^{-z}} \qquad (5)$$

- 函数图像：



1.3 逻辑回归分析

- 由于Logistic函数的定义域为(-∞,+∞)，值域为(0, 1)，因此最基本的LR分类器适合对两类目标进行分类（二分类）。当其输出大于0.5时，可以认为该实例属于甲类；小于0.5时，认为其属于乙类。
- 分类训练数据集： $D = \{X^{(i)}, y^{(i)}\}\,(1 \le i \le m)$，即**D**中有**m**个实例。若 $X = \{1, x_1, x_2, \dots, x_n\}$， $W = \{w_0, w_1, w_2, \dots, w_n\}$，则可建立逻辑回归模型为：

$$h(\mathrm{W}^T\mathrm{X}) = \frac{1}{1+e^{-W^T X}} \qquad (6)$$

- 模型的求解：上式逻辑回归分析模型的关键为权重向量**W**的取值，只有公式中的**W**已知，才能对一个未分类的数据进行二分类，那么该如何求得**W**呢？

1.3 逻辑回归分析

■逻辑回归模型求解的目标函数：

通常使用极大似然估计来建立目标函数，求得W的值。使用样本（即已知分类的数据），进行一系列的估算，得到W。这个过程在概率论中叫做参数估计。

(1) 首先我们将公式**(6)**中的$h(W^TX)$表示为$h_W(X)$，突出W为所要求得的参数。令：

$$P(y=1|X;W)=h_W(X)$$
$$P(y=0|X;W)=1-h_W(X)$$

(2) 将上述两式整合：

$$P(y|X;W)=(h_W(X))^y(1-h_W(X))^{1-y}$$

1.3 逻辑回归分析

■ 逻辑回归模型求解的目标函数：

(3) 求其似然函数：

$$l(W) = \prod_{i=1}^{m} (h_W(X^{(i)}))^{y^{(i)}} (1 - h_W(X^{(i)}))^{1-y^{(i)}}$$

(4) 对其似然函数求对数：

$$L(W) = \sum_{i=1}^{m} (y^{(i)} \ln h_W(X^{(i)}) + (1 - y^{(i)}) \ln(1 - h_W(X^{(i)})))$$

(5) 当似然函数为最大值时，得到的W即可认为是模型的参数。

1.3 逻辑回归分析

■ 基于梯度法的逻辑回归模型训练：

◆ 求似然函数的最大值，可以使用一种方法，梯度上升，但我们可以对似然函数稍作处理，使之变为梯度下降，然后使用梯度下降的思想来求解此问题，变换的表达式如下：

$$J(W) = -L(W)$$

◆在梯度下降法中，要使用当前的W（优化问题中，参数通常表示为θ）值更新得到新的W值，因而需要知道W更新的方向（即梯度）。对$J(W)$求偏导数可得到在当前点下的梯度。求梯度如下：

$$\nabla_{W_j} J(W) = \sum_{i=1}^{m} (h_W(X^{(i)}) - y^{(i)}) X_j^{(i)}$$

◆得到更新方向后便可使用下面的式子不断迭代更新得到最终结果。

$$W_t = W_{t-1} - \alpha \nabla W_{t-1}$$

1.3 逻辑回归分析

■ 基于牛顿法的逻辑回归模型训练：

(1) 基于逻辑回归模型第4步中定义的似然函数，求得一个维度的梯度计算公式为：

$$\nabla_{W_j} L(W) = X_j(y - h_W(X))$$

(2) n+1维的梯度向量为：

$$g = \{\nabla_{W_0} L(W), \nabla_{W_1} L(W), \ldots, \nabla_{W_n} L(W)\}^T$$

(3) 根据*L(W)*计算Hessian矩阵*H*，其中*j*行*k*列的值为：

$$\nabla_{W_j} \nabla_{W_k} L(W) = -X_j X_k h_W(X)(1 - h_W(X))$$

(4) (n+1)*(n+1)的Hessian矩阵为：

$$\begin{bmatrix} \nabla_{W_0}\nabla_{W_0} L(W) & \cdots & \nabla_{W_0}\nabla_{W_n} L(W) \\ \vdots & \ddots & \vdots \\ \nabla_{W_n}\nabla_{W_0} L(W) & \cdots & \nabla_{W_n}\nabla_{W_n} L(W) \end{bmatrix}$$

(5) 更新参数*W*，$W = W - H^{-1}g$

逻辑回归小结

■线性回归是利用数理统计中回归分析来确定两种或两种以上变量间相互依赖的定量关系的一种统计分析方法，是最为简单的一种预测分析模型：

◆ 模型： $h_w(\mathrm{X}) = W^T X$

◆ 求解： $J(\mathrm{W}) = \min_W \sum_{i=1}^{m} (h_w(\mathrm{X}^{(i)}) - y^{(i)})^2$

■逻辑回归分析是一种广义的线性回归分析模型，在线性回归分析的基础上添加了一个非线性变换，常用于数据挖掘，疾病自动诊断，经济预测等领域。

◆ 模型： $h(\mathrm{W}^T\mathrm{X}) = \frac{1}{1+e^{-W^T X}}$

◆ 求解： $L(W) = \sum_{i=1}^{m} (y^{(i)} \ln h_W(X^{(i)}) + (1 - y^{(i)}) \ln(1 - h_W(X^{(i)})))$

PART 02 神经网络

2.1 神经网络模型简介

- 人工神经网络（Artificial Neural Network，即ANN ），是20世纪80 年代以来人工智能领域兴起的研究热点，来源于人类对自身大脑认知能力的研究与模仿。它从信息处理角度对人脑神经元网络进行抽象，建立某种简单模型，按不同的连接方式组成不同的网络。
- 神经网络是一种运算模型，由大量的节点（或称神经元）之间相互联接构成。每个节点代表一种特定的输出函数，称为激活函数（activation function）。每两个节点间的连接都代表一个对于通过该连接信号的加权值，称之为权重，这相当于人工神经网络的记忆。网络的输出则依网络的连接方式，权重值和激励函数的不同而不同。而网络自身通常都是对自然界某种算法或者函数的逼近，也可能是对一种逻辑策略的表达。
- 最近十多年来，人工神经网络的研究工作不断深入，已经取得了很大的进展，其在模式识别、智能机器人、自动控制、预测估计、生物、医学、经济等领域已成功地解决了许多现代计算机难以解决的实际问题，表现出了良好的智能特性。

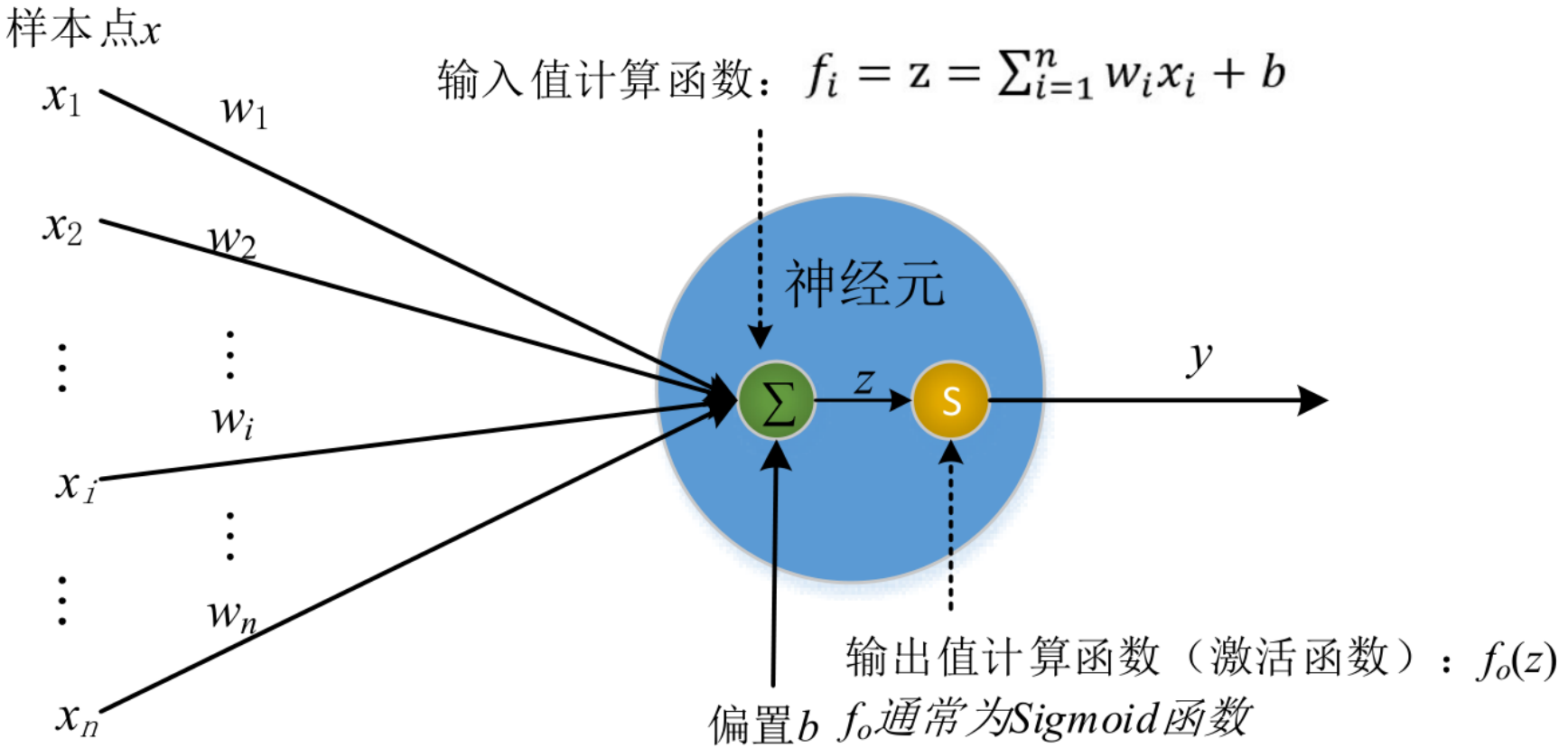
2.1 神经网络模型简介

- 单层感知网络（M-P模型）作为最初的神经网络，具有模型清晰、结构简单、计算量小等优点。但是其无法处理非线性问题。多层感知机(Multilayer Perceptron，MLP)网络（亦称之为深度前馈网络（deep feedforward network）或前馈神经网络（feedforward neural network））则是一种最为基础的深度神经网络模型。其可以逼近任意连续函数，具有很强的非线性映射能力，而且网络的中间层数、各层的处理单元数及网络的学习系数等参数可根据具体情况设定，灵活性很大，所以它在许多应用领域中起到重要作用，对人工神经网络的发展发挥了极大的推动作用。
- 误差反向传播算法（Error back propagation Training，简称BP网络）是一种最有效的多层神经网络参数学习（训练）方法，其主要特点是信号前向传递，而误差后向传播，通过不断调节网络权重值，使得网络的最终输出与期望输出尽可能接近，以达到训练的目的。BP神经网络收敛速度慢、不能保证收敛到全局最点，网络的中间层及它的单元数选取无理论指导及网络学习和记忆的不稳定性等缺陷，需要相应的改进算法。

2.2 M-P神经元网络模型

■ M-P神经元模型也称之为感知机（perceptron）模型，是最简单的神经网络模型，在其中仅包含一个神经元。其模型结构如下图所示：

■ M-P 神经元的输入值计算函数 f_i 是样本点向量加偏置（bias）的一个线性组合。而输出值计算函数 f_o 也称为激活函数，如果采用的是 Sigmoid 函数，M-P 神经元模型则可变为逻辑回归模型，进行二值分类。



2.2 M-P神经元网络模型

激活函数

- 激活函数（Activation Function）给神经元引入了非线性因素，使得神经网络可以任意逼近任何非线性函数，这样神经网络就可以应用到众多的非线性模型中。相反，如果不使用激活函数，则每一层输出都是上层输入的线性函数，无论神经网络有多少层，输出都是输入的线性组合。
- 激活函数的种类有很多，需要根据问题的性质，同时考虑训练性能和应用效果两个方面进行选择。一般来说：
  - 用于分类器时，Sigmoid函数及其组合通常效果更好。但由于梯度消失和训练性能问题，有时要避免使用sigmoid和tanh函数。
  - ReLU函数是一个通用的隐藏层中使用的激活函数，目前在大多数情况下使用。如果神经网络中出现死神经元，可以使用PReLU函数。
  - SoftMax函数是输出层常用的激活函数。

2.2 M-P神经元网络模型

激活函数

■ Logistic Sigmoid函数：

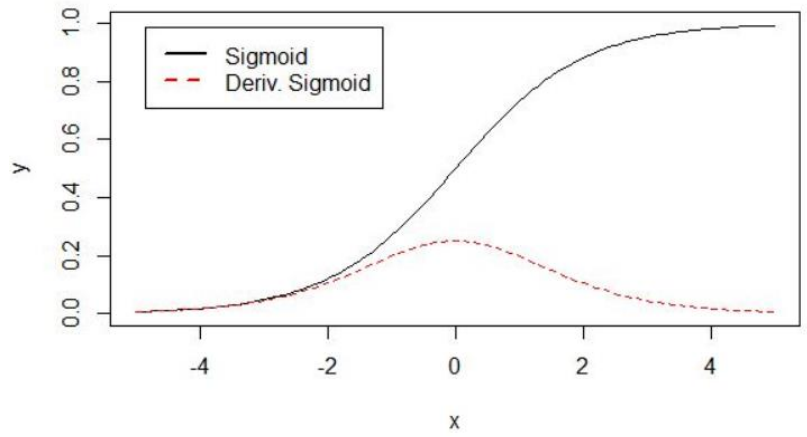
■函数式：

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

■导函数式：

$$f'(x) = f(x)(1 - f(x))$$

■**Logistic Sigmoid函数及其导函数的几何图像如下：**



■ **特点：Logistic sigmoid函数用于隐层神经元输出，取值范围为(0,1)，它可以将一个实数映射到(0,1)的区间，可以用来做二分类。在特征相差比较复杂或是相差不是特别大时效果比较好。**

■ **缺点：**

- ◆ 激活函数计算量大，反向传播求误差梯度时，求导涉及除法；
- ◆ 反向传播时，很容易就会出现梯度消失的情况，从而无法完成深层网络的训练；
- ◆ **Sigmoid函数的输出不是0均值。**

2.2 M-P神经元网络模型

激活函数

- 双曲正切函数（hyperbolic tangent function，tanh）：
- 函数式：

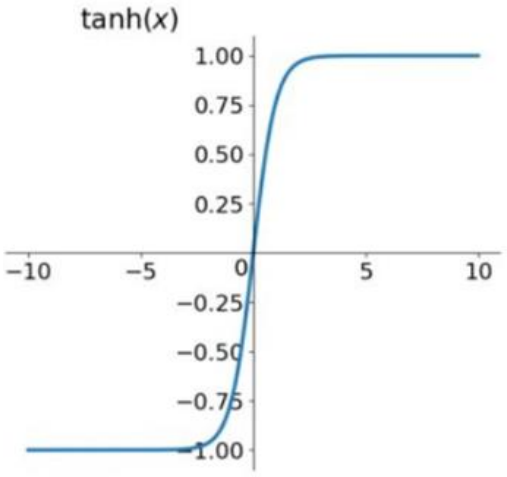
$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

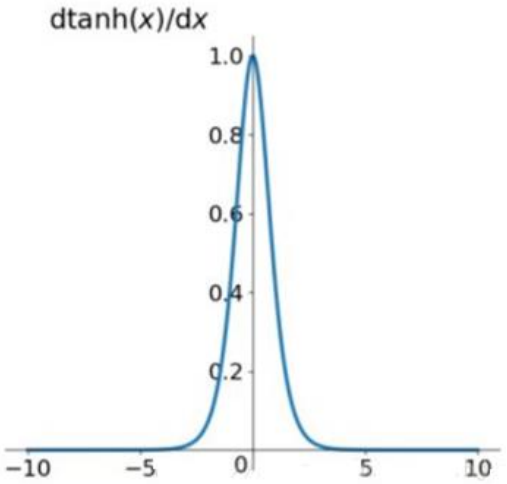
$$f(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} = 2sigmoid(2x) - 1$$

- 导函数式：

$$f'(x) = 1 - f(x)^2$$

- tanh函数及其导函数图像如下：





- 特点：取值范围为[-1,1]。tanh在特征相差明显时的效果会很好，在循环过程中会不断扩大特征效果。与sigmoid的区别是，tanh是0均值（zero-centered）的，因此实际应用中 tanh会比sigmoid 更好。然而，梯度消失（gradient vanishing）的问题和幂运算的问题仍然存在。

2.2 M-P神经元网络模型

激活函数

■ 线性整流函数（Rectified Linear Unit, ReLU）：又称修正线性单元, 是一种人工神经网络中常用的激活函数，通常指代以斜坡函数及其变种为代表的非线性函数。

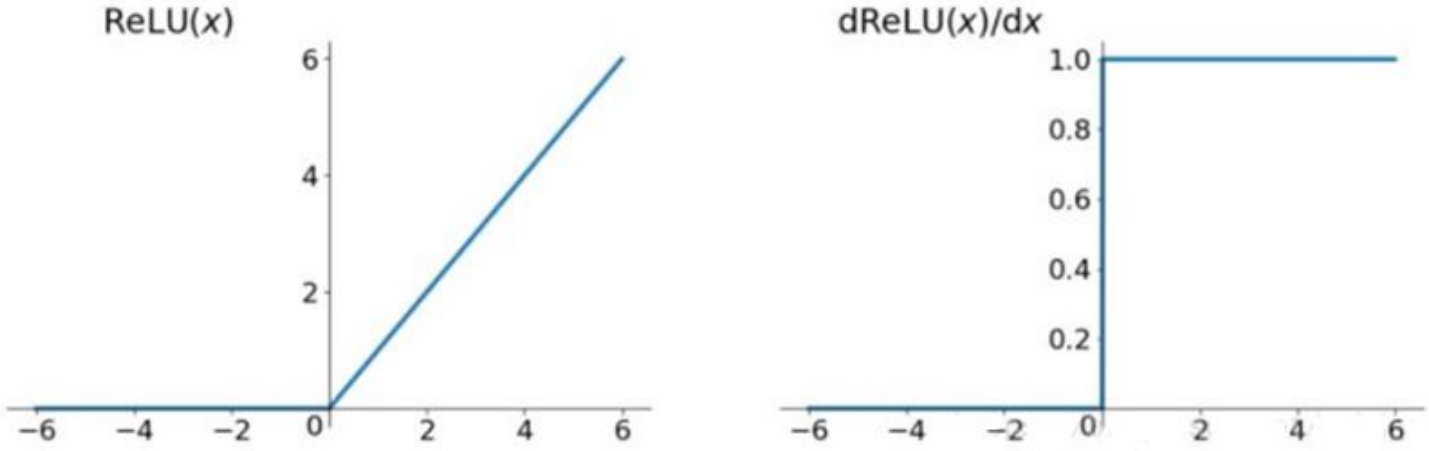
■ 函数式：

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

■ 导函数式：

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

■ Relu函数及其导函数图像如下所示：



2.2 M-P神经元网络模型

激活函数

- 线性整流函数（Rectified Linear Unit, ReLU）特点：输入信号<0时，输出都是0，>0的情况下，输出等于输入。虽然该函数不是全区间可导，但是可以取sub-gradient。当输入为正时，不会有梯度消失问题；计算速度与收敛速度快。
- 缺点：
  - 当输入为负时，梯度为0，会产生梯度消失问题。为了解决该问题，在Relu函数的负半区间引入一个泄露（Leaky）值，所以称为Leaky Relu函数。另外常用的变形还包括PReLU和Maxout函数；
  - ReLU的输出不是zero-centered；
  - Dead ReLU Problem，指的是某些神经元可能永远不会被激活，导致相应的参数永远不能被更新；
- Relu是最常用的默认激活函数，若不确定用哪个激活函数，就使用Relu或者Leaky Relu函数。

2.2 M-P神经元网络模型

激活函数

■归一化指数函数（Softmax Function）：Softmax激活函数被广泛用作输出层的激活函数，该函数的范围是[0, 1]。在多类分类问题中，它被用来表示一个类的概率。所有单位输出和总是1。

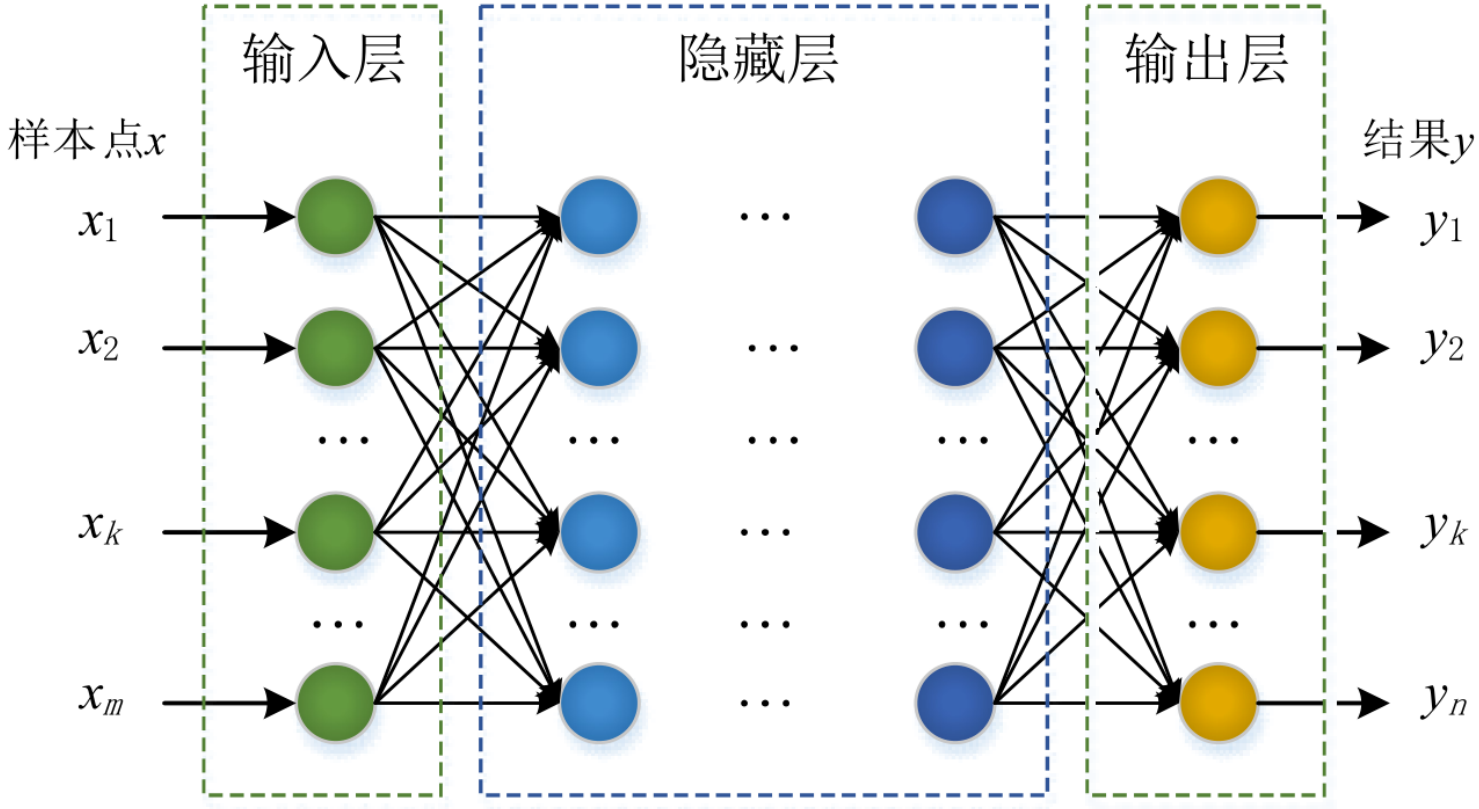
■ 函数式：

$$f(z)_j = \frac{e^{z_j}}{\sum_{i=1}^{m} e^{z_i}}$$

其中，$z \in \mathbb{R}^m$ 是一个m维实向量。z_j (j=1,2, ..., m)是向量z中的第j个元素。

2.3 多层感知机网络模型

■一个**MLP网络**通常由L层神经元组成，其中：第1层称为**输入层**，最后一层（第L层）被称为**输出层**，其它各层（第2层至第L-1层）被称为**隐含层**。



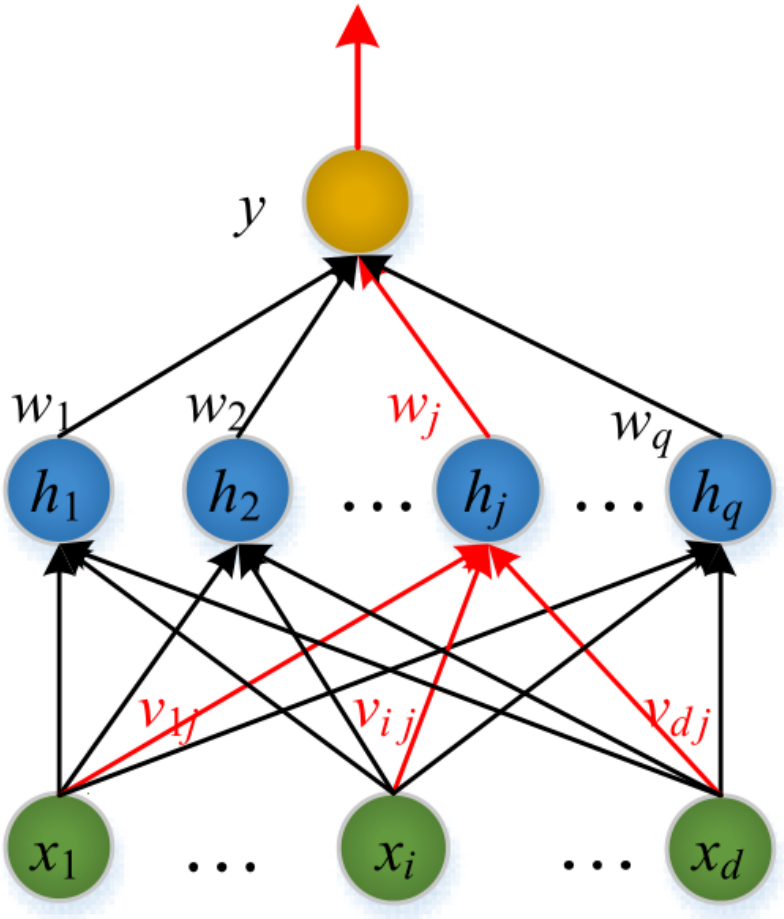
2.4 BP神经网络的训练

- 基本BP算法包括信号的前向传播和误差的反向传播两个过程。即计算误差输出时按从输入到输出的方向进行，而调整权值和阈值则从输出到输入的方向进行。
- 正向传播时，输入信号通过隐含层作用于输出节点，经过非线性变换，产生输出信号，若实际输出与期望输出不相符，则转入误差的反向传播过程。
- BP神经网络的误差逆传播算法需要根据问题所采用的目标函数和网络中所选择的激活函数进行推导。比如在进行二分类和多分类时，可以根据具体的目标函数和激活函数推导BP神经网络的训练过程。

2.4 BP神经网络的训练

二分类BP神经网络的训练

- 给定训练集$D=\{(x_1, y_1), (x_1, y_1), \ldots, (x_m, y_m)\}$，$x_i \in \mathbb{R}^d$，即输入实例由$d$个属性描述。所建立的网络模型如右图所示。仅包含单隐层，拥有d个输入神经元、q个隐层神经元和1个输出神经元。隐藏层第j个神经元的偏执为v_{0j}，输入层第i个神经元到隐层第j个神经元的连接权值为v_{ij}。
- 给定训练示例(x, y)，$x=<x_1, x_2, \ldots, x_d>$。则其输入为：$h_j^{in} = \sum_{i=0}^{d} v_{id}\, x_i$，其中$x_0$取值为1。其输出为：$h_j^{out} = Sigmoid(h_j^{in}) = 1/(1+e^{-h_j^{in}})$。输出层神经元的偏执阈值为$w_0$，和隐层各神经元的连接权值分别为$w_1, w_1, \ldots w_q$。则其输入为：$y^{in} = \sum_{i=0}^{q} w_i h_i^{out}$，其中$h_0^{out}$取值为1。其输出为：$\hat{y} = Sigmoid(y^{in}) = 1/(1+e^{-y^{in}})$。

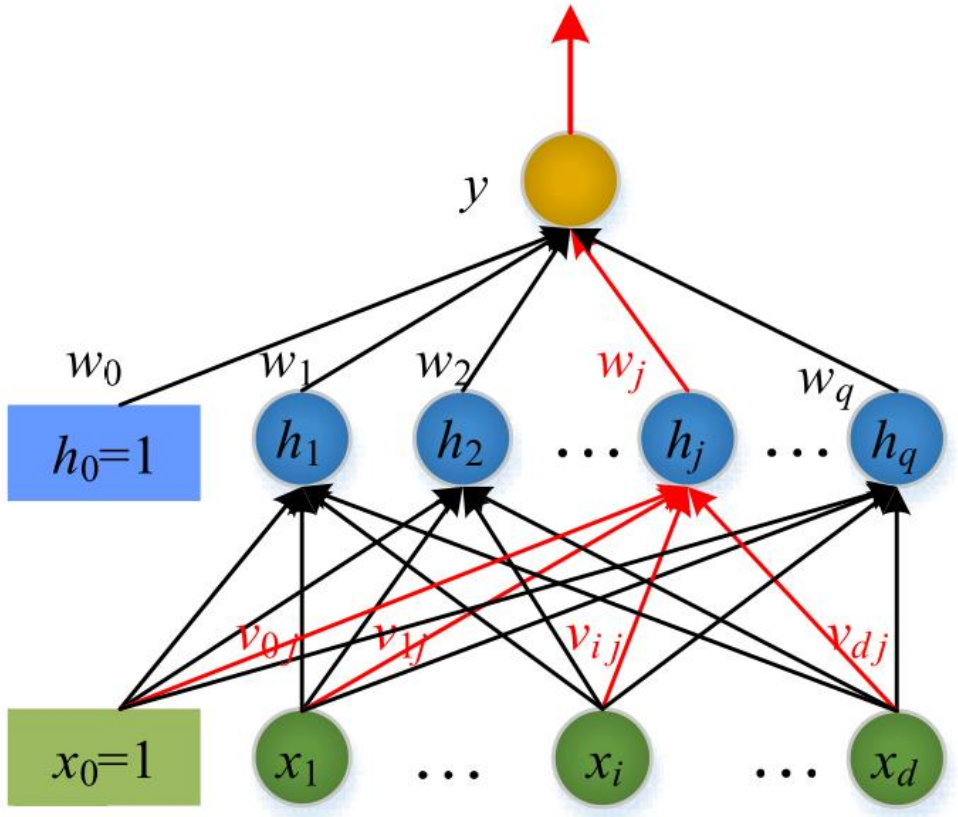


2.4 BP神经网络的训练

二分类BP神经网络的训练

■因而训练该神经网络模型，关键在于计算w_i ($i \in \{0, 1, \ldots, q\}$) 和$v_{ij}$ ($i \in \{0, 1, \ldots, d\}$, $j \in \{1, \ldots, q\}$)这些权值（统一表示为θ）。采用逻辑回归模型中的对数极大似然函数作为目标函数，采用误差反向传播算法对各权值进行训练。

■为了方便编程，采用如右图所示的网络结构，将隐层和输出层各节点的偏执和其对应的输入连接权值统一考虑。



2.4 BP神经网络的训练

二分类BP神经网络的训练

■给定训练实例(x, y)，$x=<x_1, x_2,\ldots, x_d>$，将其对数似然函数作为目标函数：

$$L(\theta) = y \ln \hat{y} + (1-y)\ln(1-\hat{y})$$

■当似然函数为最大值时，得到的权值即为模型的参数，可采用梯度上升方法。首先，求隐层神经元到输出层神经元的各连接权值的梯度为：

$$\nabla_{W_j} L(\theta) = (y-\hat{y}) h_j^{out}$$

■然后沿着梯度的方向，不断迭代更新便可得到最终结果。

$$W_t = W_{t-1} + \eta \nabla W_{t-1}$$

■其中$\eta \in (0,1)$，称之为学习率或步长。

■其次，求输入层神经元到隐层神经元连接权值v_{ij}的梯度为：

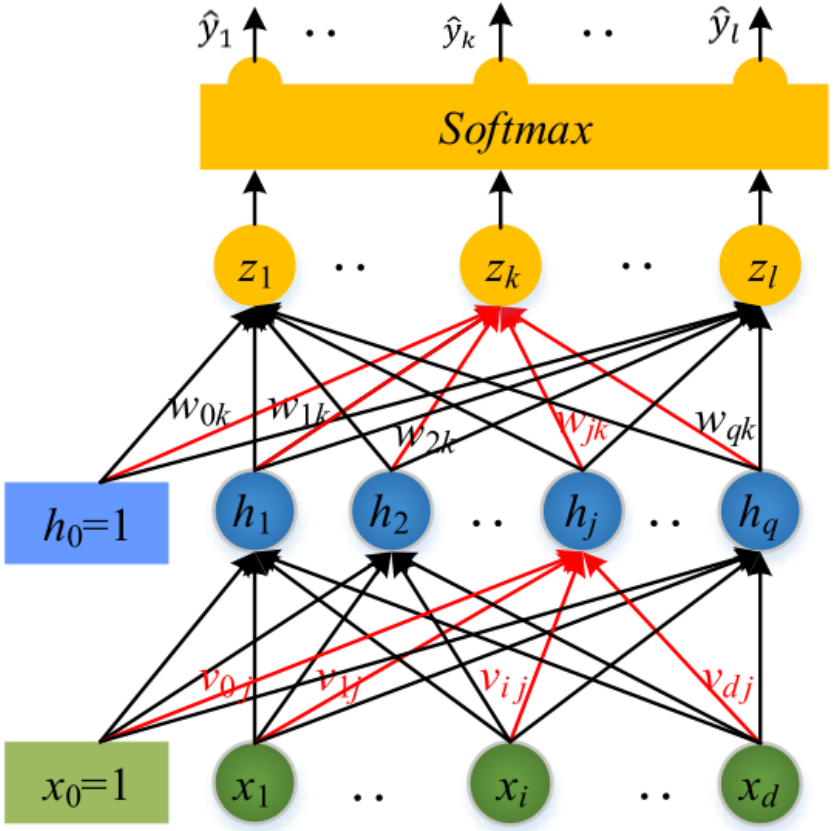
$$\nabla_{v_{ij}} L(\theta) = \frac{\partial L(\theta)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial y^{in}} \frac{\partial y^{in}}{\partial h_j^{out}} \frac{\partial h_j^{out}}{\partial h_j^{in}} \frac{\partial h_j^{in}}{\partial v_{ij}} = (y-\hat{y}) w_j h_j^{out}(1-h_j^{out}) x_i$$

■同样沿着梯度的方向，以给定的学习率η不断迭代更新便可得到最终结果。

2.4 BP神经网络的训练

多分类BP神经网络的训练

- 给定训练集D={$(x_1, y_1), (x_1, y_1), \ldots, (x_m, y_m)$}，$x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}^l$。即输入实例由d个属性描述，输出为l维实值向量。所建立的网络模型如右图所示：
- 该网络结构仅包含单隐层，拥有d个输入神经元、q个隐层神经元和l个输出神经元。隐藏层第j个神经元的偏执阈值为v_{0j}，输入层第i个神经元到隐层第j个神经元的连接权值为v_{ij}。给定训练示例(x, y)，$x=<x_1, x_2, \ldots, x_d>$，$y=<y_1, y_2, \ldots, y_l>$。则其输入为：$h_j^{in} = \sum_{i=0}^{d} v_{id}\, x_i$，其中$x_0$取值为1。隐层神经元采用Sigmoid激活函数，所以其输出为：$h_j^{out} = Sigmoid(h_j^{in}) = 1/(1+e^{-h_j^{in}})$。输出层采用softmax函数。具有$l$个输入节点$<z_1, z_2, \ldots, z_l>$和$l$个输出节点$<\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_l>$。其中，$z_k = \sum_{i=0}^{q} w_{ik} h_i^{out}$，$\hat{y}_k = e^{z_k} / \sum_{i=0}^{l} e^{z_i}$。



2.4 BP神经网络的训练

多分类BP神经网络的训练

■给定训练示例(x, y)，$x=<x_1, x_2,\ldots, x_d>$，$y=<y_1, y_2,\ldots, y_l>$，将交叉熵作为目标函数：

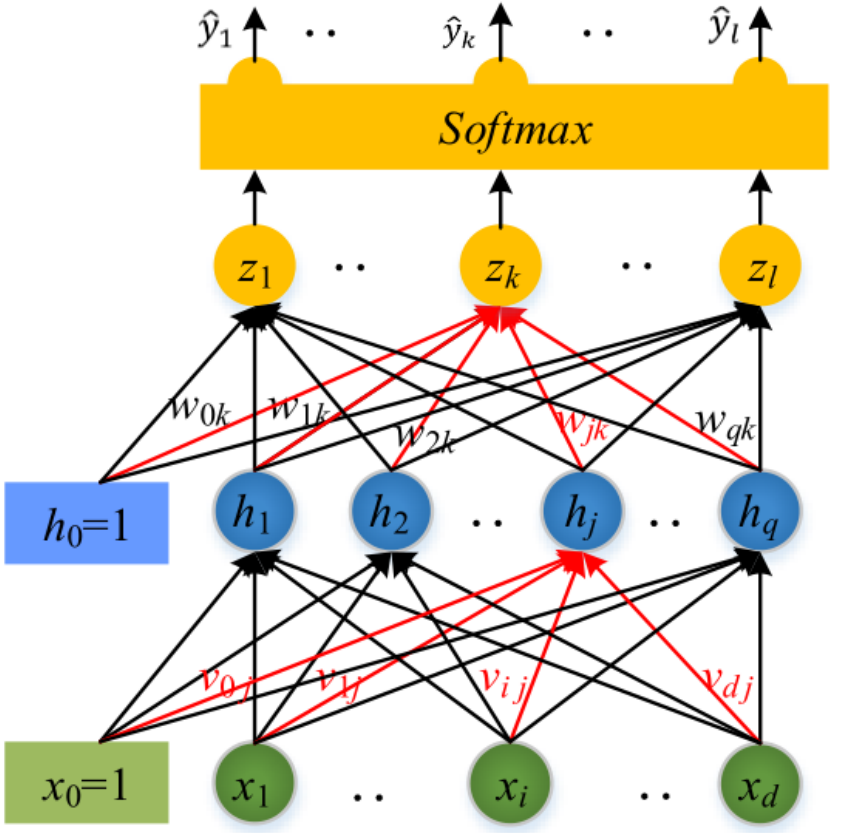
$$L(\theta) = -\sum_{i=1}^{l} y_i \ln \hat{y}_i$$

■首先求目标函数$L(\theta)$对输出层任一输入节点z_k的梯度。则：

$$\frac{\partial L(\theta)}{\partial z_k} = \sum_{i=1}^{l} \frac{\partial L(\theta)}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_k}$$

■其中：

$$\frac{\partial L(\theta)}{\partial \hat{y}_i} = \frac{\partial(-\sum_{j=1}^{l} y_j \ln \hat{y}_j)}{\partial \hat{y}_i} = -\frac{y_i}{\hat{y}_i}$$



2.4 BP神经网络的训练

多分类BP神经网络的训练

■当$i=k$时，有：

$$\frac{\partial \hat{y}_i}{\partial z_k}=\frac{\partial(e^{z_k}/\sum_{i=0}^{l}e^{z_i})}{\partial z_k}=\frac{e^{z_k}\sum_{i=0}^{l}e^{z_i}-e^{z_k}e^{z_k}}{(\sum_{i=0}^{l}e^{z_i})^2}=\hat{y}_k(1-\hat{y}_k)$$

■当$i\neq k$时，有：

$$\frac{\partial \hat{y}_i}{\partial z_k}=\frac{\partial(e^{z_i}/\sum_{i=0}^{l}e^{z_i})}{\partial z_k}=-\frac{e^{z_i}}{\left(\sum_{i=0}^{l}e^{z_i}\right)^2}e^{z_k}=-\hat{y}_i\hat{y}_k$$

■因而，有：

$$\frac{\partial L(\theta)}{\partial z_k}=\sum_{i=1}^{l}-\frac{y_i}{\hat{y}_i}\frac{\partial \hat{y}_i}{\partial z_k}=\sum_{i\neq k}-\frac{y_i}{\hat{y}_i}(-\hat{y}_i\hat{y}_k)+\sum_{i=k}-\frac{y_i}{\hat{y}_i}\hat{y}_k(1-\hat{y}_k)$$

$$=\sum_{i\neq k}y_i\hat{y}_k+\sum_{i=k}(y_i\hat{y}_k-y_k)=\hat{y}_k\sum_{i=1}^{l}y_i-y_k$$

2.4 BP神经网络的训练

多分类BP神经网络的训练

■针对多类分类问题，给定任一实例，分类结果会使得给定类别y_i为1，而其他类别都是0。即$\sum_{i=1}^{l} y_i = 1$。所以，对于分类问题，有：

$$\frac{\partial L(\theta)}{\partial z_k} = \hat{y}_k - y_k$$

■在此基础上，求隐层神经元到输出层神经元的各连接权值的梯度为：

$$\nabla_{W_{jk}} L(\theta) = \frac{\partial L(\theta)}{\partial z_k} \frac{\partial z_k}{\partial w_{jk}} = (\hat{y}_k - y_k) h_j^{out}$$

■然后逆着梯度的方向，不断迭代更新便可得到最终结果。

$$W_t = W_{t-1} - \eta \nabla W_{t-1}$$

■其中$\eta \in (0,1)$，称之为学习率或步长。

2.4 BP神经网络的训练

多分类BP神经网络的训练

■其次，求输入层神经元到隐层神经元连接权值v_{ij}的梯度为：

$$\nabla_{v_{ij}} L(\theta) = \frac{\partial L(\theta)}{\partial h_j^{out}} \frac{\partial h_j^{out}}{\partial h_j^{in}} \frac{\partial h_j^{in}}{\partial v_{ij}} = \sum_{i=1}^{l} \frac{\partial L(\theta)}{\partial z_i} \frac{\partial z_i}{\partial h_j^{out}} \frac{\partial h_j^{out}}{\partial h_j^{in}} \frac{\partial h_j^{in}}{\partial v_{ij}}$$

$$= h_j^{out}(1 - h_j^{out}) x_i \sum_{k=1}^{l} (\hat{y}_k - y_k) w_{jk}$$

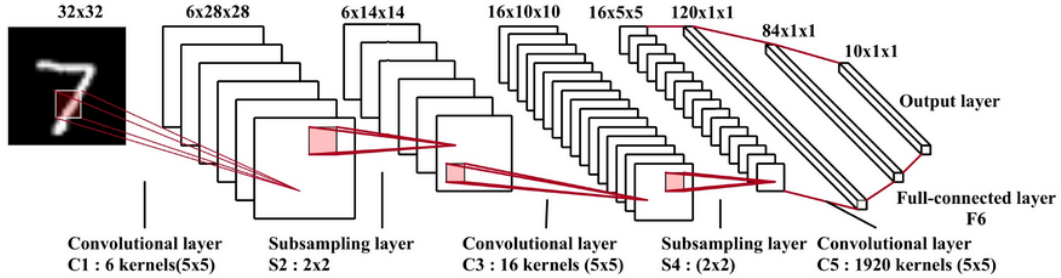
■同样逆着梯度的方向，以给定的学习率η不断迭代更新便可得到最终结果。

■BP神经网络是神经网络模型中较简单的一种。除了用了分类，还可以用到其它各个方面。应用时需要设计具体的网络结构、激活函数、目标函数等。

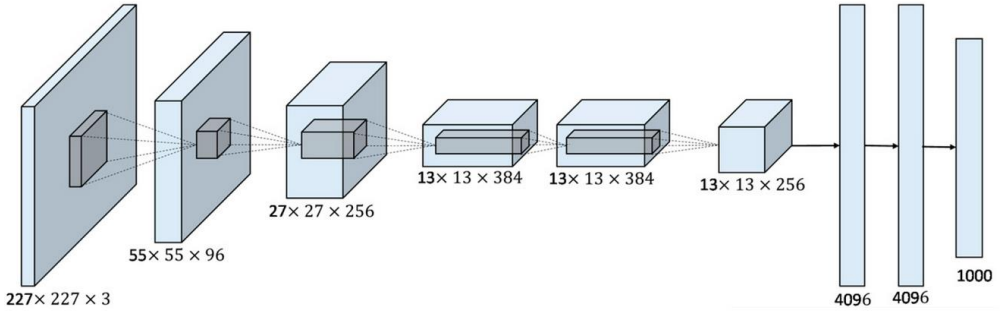
2.5 其它流行的神经网络结构

卷积神经网络

- 卷积神经网络(Convolutional Neural Networks: CNN) 利用空间关系减少需要学习的参数数目以提高一般前向BP算法的训练性能。CNNs作为一个深度学习架构提出是为了最小化数据的预处理要求。
- 在CNN中，图像的一小部分（局部感受区域）作为层级结构的最低层的输入，信息再依次传输到不同的层，每层通过一个数字滤波器去获得观测数据的最显著的特征。这个方法能够获取对平移、缩放和旋转不变的观测数据的显著特征，因为图像的局部感受区域允许神经元或者处理单元可以访问到最基础的特征，例如定向边缘。



LeNet-5 network

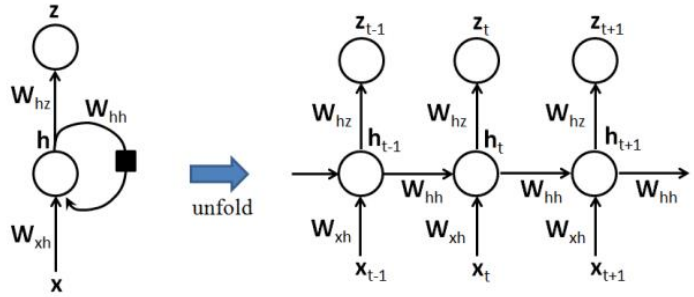


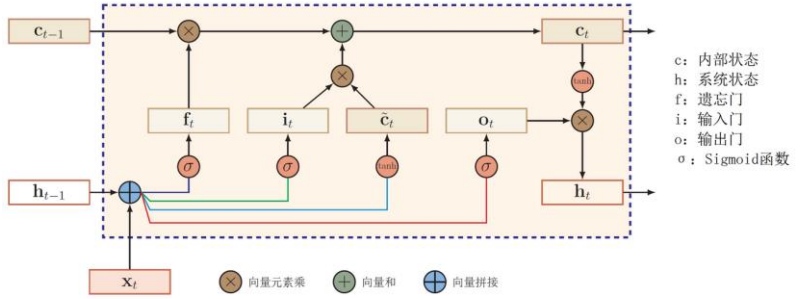
The AlexNet architecture.

2.5 其它流行的神经网络结构

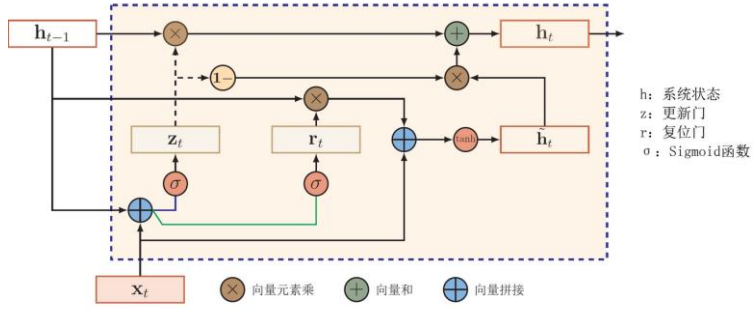
循环神经网络

- 循环神经网络（Recurrent Neural Network, RNN）是一类以序列（sequence）数据为输入，在序列的演进方向进行递归（recursion）且所有节点（循环单元）按链式连接的递归神经网络（recursive neural network）。
- 循环神经网络具有记忆性、参数共享的特点，在对序列的非线性特征进行学习时具有一定优势。循环神经网络在自然语言处理，例如语音识别、语言建模、机器翻译等领域有广泛应用，也被用于各类时间序列预报。





LSTM 循环单元结构

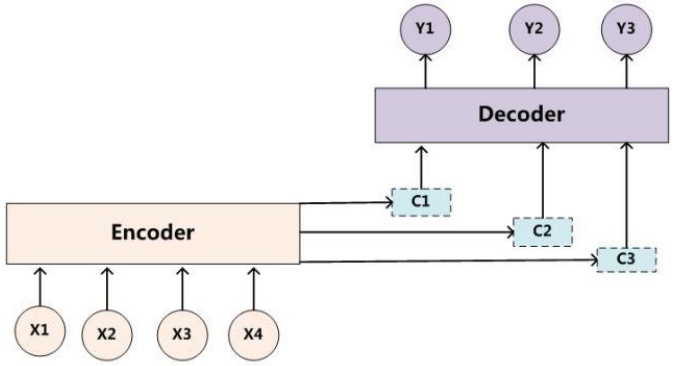


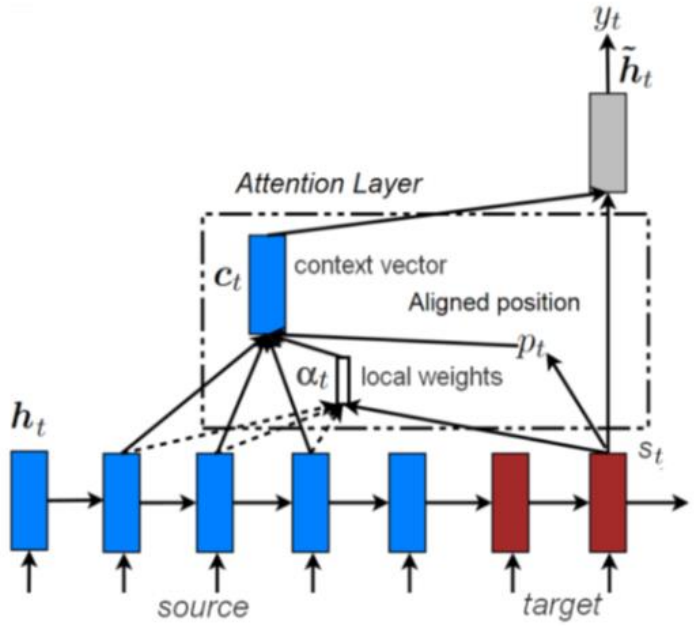
GRU 循环单元结构

2.5 其它流行的神经网络结构

注意力网络

- Encoder-Decoder最大的局限性就在于编码和解码之间的唯一联系就是一个固定长度的语义向量C。编码器要将整个序列的信息压缩进一个固定长度的向量中去。但是这样做有两个弊端：一是语义向量无法完全表示整个序列的信息；二是先输入的内容携带的信息会被后输入的信息稀释掉。输入序列越长，这个现象就越严重。
- Attention机制通过在每个时间输入不同的c来解决这个问题。相比于encoder-decoder模型，attention模型最大的区别就在于它不在要求编码器将所有输入信息都编码进一个固定长度的向量之中。
- 相反，此时编码器需要将输入编码成一个向量的序列，而在解码的时候，每一步都会选择性的从向量序列中挑选一个子集进行进一步处理。这样，在产生每一个输出的时候，都能够做到充分利用输入序列携带的信息。





神经网络小结

- 单层感知网络：最初的神经网络
- 多层感知机：一种最为基础的深度神经网络模型
- 其它经典的神经网络模型：卷积神经网络、循环神经网络、注意力网络等
- 神经网络的经典训练算法：误差反向传播算法。

THANK YOU FOR YOUR WATCHING

Teacher：张明卫
@东北大学 软件学院