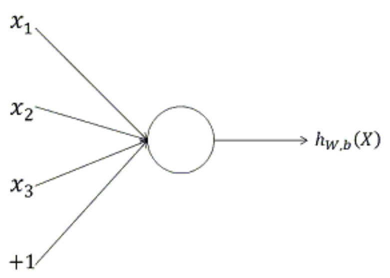
深度学习算法原理——神经网络的基本原理

一、神经网络

1、神经元概述

神经网络是由一个个的被称为“神经元”的基本单元构成，单个神经元的结构如下图所示：



对于上述的神经元，其输入为x_1，x_2，x_3以及截距$+1$，其输出为：

$$h_{\mathbf{W},b}\left(\mathbf{x}\right)=f\left(\mathbf{W}^T\mathbf{x}\right)=f\left(\sum_{i=1}^{3}W_ix_i+b\right)$$

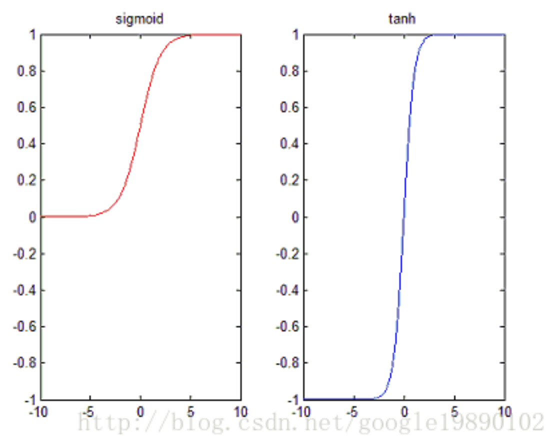
其中，$\mathbf{W}$表示的是向量，代表的是权重，函数f称为激活函数，通常激活函数可以选择为Sigmoid函数，或者tanh双曲正切函数，其中，Sigmoid函数

$$f\left(z\right)=\frac{1}{1+e^{-z}}$$

双曲正切函数的形式为：

$$f\left(z\right)=tanh\left(z\right)=\frac{e^z-e^{-z}}{e^z+e^{-z}}$$

以下分别是Sigmoid函数和tanh函数的图像，左边为Sigmoid函数的图像，右边为tanh函数的图像：



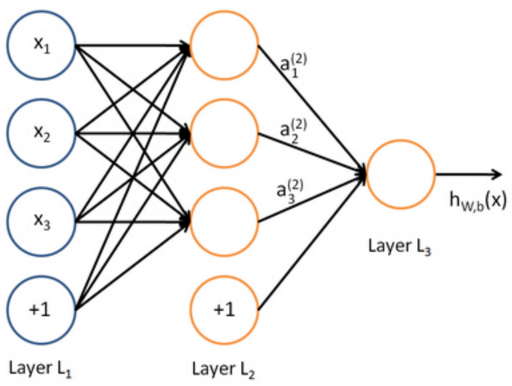
> Sigmoid函数的区间为$[0,1]$，而tanh函数的区间为$[-1,1]$。

若是使用sigmoid作为神经元的激活函数，则当神经元的输出为1时表示该神经元被激活，否则称为未被激活。同样，对于激活函数是tanh时，神经元
表示该神经元被激活，否则称为未被激活。

2、神经网络

2.1、神经网络的结构

神经网络是由很多的神经元联结而成的，一个简单的神经网络的结构如下图所示：



其中一个神经元的输出是另一个神经元的输入，$+1$项表示的是偏置项。上图是含有一个隐含层的神经网络模型，L_1层称为输入层，L_2层称为隐含层出层。

2.2、神经网络中的参数说明

在神经网络中，主要有如下的一些参数标识：

- 网络的层数n_1。在上述的神经网络中$n_l=3$，将第l层记为L_l，则上述的神经网络，输入层为L_1，输出层为L_3。
- 网络权重和偏置$(\mathbf{W},\mathbf{b})=\left(\mathbf{W}^{(1)},\mathbf{b}^{(1)},\mathbf{W}^{(2)},\mathbf{b}^{(2)}\right)$，其中$W_{ij}^{(l)}$表示的是第$l$层的第$j$个神经元和第$l+1$层的第$i$个神经元之间的连接参数，$l$ $l+1$层的第i个神经元的偏置项。在上图中，$\mathbf{W}^{(1)}\in\Re^{3\times 3}$，$\mathbf{W}^{(2)}\in\Re^{1\times 3}$。

2.3、神经网络的计算

在神经网络中，一个神经元的输出是另一个神经元的输入。假设$z_i^{(l)}$表示的是第l层的第i个神经元的输入，假设$a_i^{(l)}$表示的是第l层的第i个神经元的输出 $l=1$时，$a_i^{(1)}=x_i$。根据上述的神经网络中的权重和偏置，就可以计算神经网络中每一个神经元的输出，从而计算出神经网络的最终的输出$h_{\mathbf{W},\mathbf{b}}$。

对于上述的神经网络结构，有下述的计算：

$$z_1^{(2)}=W_{11}^{(1)}x_1+W_{12}^{(1)}x_2+W_{13}^{(1)}x_3+b_1^{(1)}$$

$$a_1^{(2)}=f\left(W_{11}^{(1)}x_1+W_{12}^{(1)}x_2+W_{13}^{(1)}x_3+b_1^{(1)}\right)$$

$$z_2^{(2)}=W_{21}^{(1)}x_1+W_{22}^{(1)}x_2+W_{23}^{(1)}x_3+b_2^{(1)}$$

$$a_2^{(2)}=f\left(W_{21}^{(1)}x_1+W_{22}^{(1)}x_2+W_{23}^{(1)}x_3+b_2^{(1)}\right)$$

$$z_3^{(2)}=W_{31}^{(1)}x_1+W_{32}^{(1)}x_2+W_{33}^{(1)}x_3+b_3^{(1)}$$

$$a_3^{(2)}=f\left(W_{31}^{(1)}x_1+W_{32}^{(1)}x_2+W_{33}^{(1)}x_3+b_3^{(1)}\right)$$

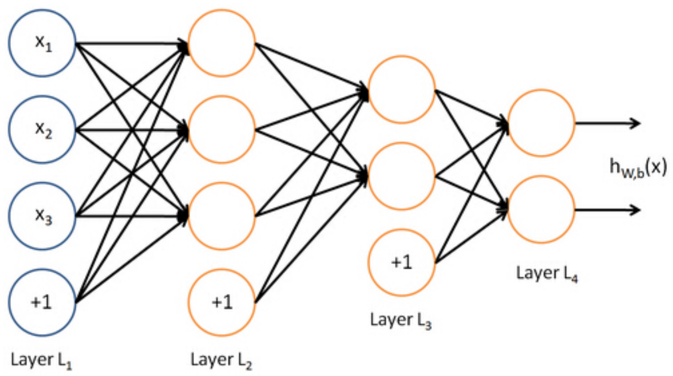
从而，上述神经网络结构的最终的输出结果为：

$$h_{\mathbf{W},\mathbf{b}}(\mathbf{x})=f\left(W_{11}^{(2)}a_1^{(2)}+W_{12}^{(2)}a_2^{(2)}+W_{13}^{(2)}a_3^{(2)}+b_1^{(2)}\right)$$

上述的步骤称为**前向传播**，指的是信号从输入层，经过每一个神经元，直到输出神经元的传播过程。

2.4、其他形式的神经网络模型

上述以单隐层神经网络为例介绍了神经网络的基本结构，在神经网络的结构中，可以包含多个隐含层，神经网络的输出神经单元也可以是多个，如下面输出单元的神经网络模型：



2.5、神经网络中参数的求解

对于上述神经网络模型，假设有m个训练样本$\left\{ \left(\mathbf{x}^{(1)}, y^{(1)}\right), \cdots, \left(\mathbf{x}^{(m)}, y^{(m)}\right) \right\}$，对于一个训练样本$(\mathbf{x}, y)$，其损失函数为：

$$J\left(\mathbf{W}, \mathbf{b}; \mathbf{x}, y\right) = \frac{1}{2}\left\| h_{\mathbf{W}, \mathbf{b}}\left(\mathbf{x}\right) - y \right\|^2$$

为了防止模型的过拟合，在损失函数中会加入正则项，即：

$$J = loss + R$$

其中，$loss$表示的是损失函数，R表示的是正则项。则对于上述的含有m个样本的训练集，其损失函数为：

$$J\left(\mathbf{W}, \mathbf{b}\right) = \left[\frac{1}{m}\sum_{i=1}^{m} J\left(\mathbf{W}, \mathbf{b}; \mathbf{x}^{(i)}, y^{(i)}\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l - 1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ij}^{(l)}\right)^2$$

> 通常，偏置项并不放在正则化中，因为在正则化中放入偏置项只会对神经网络产生很小的影响。

我们的目标是要求得参数$\mathbf{W}$和参数$\mathbf{b}$以使得损失函数$J\left(\mathbf{W}, \mathbf{b}\right)$达到最小值。首先需要对参数进行随机初始化，即将参数初始化为一个很小的接近0的

> 参数的初始化有很多不同的策略，基本的是要在0附近的很小的邻域内取得随机值。

在随机初始化参数后，利用**前向传播**得到预测值$h_{\mathbf{W}, \mathbf{b}}\left(\mathbf{x}\right)$，进而可以得到损失函数，此时需要利用损失函数对其参数进行调整，可以使用梯度下降的
降对参数的调整如下：

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha\frac{\partial}{\partial W_{ij}^{(l)}} J\left(\mathbf{W}, \mathbf{b}\right)$$

$$b_i^{(l)} = b_i^{(l)} - \alpha\frac{\partial}{\partial b_i^{(l)}} J\left(\mathbf{W}, \mathbf{b}\right)$$

其中，α称为学习率，在计算参数的更新公式中，需要使用到**反向传播算法。**

而$\frac{\partial}{\partial W_{ij}^{(l)}} J\left(\mathbf{W}, \mathbf{b}\right)$，$\frac{\partial}{\partial b_i^{(l)}} J\left(\mathbf{W}, \mathbf{b}\right)$的具体形式如下：

$$\frac{\partial}{\partial W_{ij}^{(l)}} J\left(\mathbf{W}, \mathbf{b}\right) = \left[\frac{1}{m}\sum_{i=1}^{m}\frac{\partial}{\partial W_{ij}^{(l)}} J\left(\mathbf{W}, \mathbf{b}; \mathbf{x}^{(i)}, y^{(i)}\right)\right] + \lambda W_{ij}^{(l)}$$

$$\frac{\partial}{\partial b_i^{(l)}} J\left(\mathbf{W}, \mathbf{b}\right) = \frac{1}{m}\sum_{i=1}^{m}\frac{\partial}{\partial b_i^{(l)}} J\left(\mathbf{W}, \mathbf{b}; \mathbf{x}^{(i)}, y^{(i)}\right)$$

反向传播算法的思路如下：对于给定的训练数据$(\mathbf{x}, y)$，通过**前向传播算法**计算出每一个神经元的输出值，当所有神经元的输出都计算完成后，对每一
其“残差”，如第l层的神经元i的残差可以表示为$\delta_i^{(l)}$。该残差表示的是该神经元对最终的残差产生的影响。这里主要分为两种情况，一是神经元为输
二是神经元为非输出神经元。这里假设$z_i^{(l)}$表示第l层上的第i个神经元的输入加权和，假设$a_i^{(l)}$表示的是第l层上的第i个神经元的输出，即$a_i^{(l)} = f\left(\right.$

- 对于输出层n_l上的神经元i，其残差为：

$$
\begin{aligned}
\delta_i^{(nl)} &= \frac{\partial}{\partial z_i^{nl}} J\left(\mathbf{W},\mathbf{b};\mathbf{x},y\right) \\
&= \frac{\partial}{\partial z_i^{nl}} \frac{1}{2}\left\|y-h_{\mathbf{W},\mathbf{b}}\left(\mathbf{x}\right)\right\|^2 \\
&= \frac{\partial}{\partial z_i^{nl}} \frac{1}{2}\sum_{i=1}^{s_{nl}}\left\|y_i-a_i^{nl}\right\|^2 \\
&= \left(y_i-a_i^{nl}\right)\cdot\left(-1\right)\cdot\frac{\partial}{\partial z_i^{nl}} a_i^{nl} \\
&= -\left(y_i-a_i^{nl}\right)\cdot f'\left(z_i^{nl}\right)
\end{aligned}
$$

-对于非输出层，即对于$l=n_{l-1},n_{l-2},\cdots,2$各层，第$l$层的残差的计算方法如下（以第$n_{l-1}$层为例）：

$$
\begin{aligned}
\delta_i^{(nl-1)} &= \frac{\partial}{\partial z_i^{nl-1}} J\left(\mathbf{W},\mathbf{b};\mathbf{x},y\right) \\
&= \frac{\partial}{\partial z_i^{nl-1}} \frac{1}{2}\left\|y-h_{\mathbf{W},\mathbf{b}}\left(\mathbf{x}\right)\right\|^2 \\
&= \frac{\partial}{\partial z_i^{nl-1}} \frac{1}{2}\sum_{j=1}^{s_{nl}}\left\|y_j-a_j^{nl}\right\|^2 \\
&= \frac{1}{2}\sum_{j=1}^{s_{nl}} \frac{\partial}{\partial z_i^{nl-1}}\left\|y_j-a_j^{nl}\right\|^2 \\
&= \frac{1}{2}\sum_{j=1}^{s_{nl}} \frac{\partial}{\partial z_j^{nl}}\left\|y_j-a_j^{nl}\right\|^2\cdot\frac{\partial}{\partial z_i^{nl-1}} z_j^{nl} \\
&= \sum_{j=1}^{s_{nl}} \delta_j^{(nl)}\cdot\frac{\partial}{\partial z_i^{nl-1}} z_j^{nl} \\
&= \sum_{j=1}^{s_{nl}}\left(\delta_j^{(nl)}\cdot\frac{\partial}{\partial z_i^{nl-1}}\sum_{k=1}^{s_{nl-1}} f\left(z_k^{nl-1}\right)\cdot W_{jk}^{nl-1}\right) \\
&= \sum_{j=1}^{s_{nl}}\left(\delta_j^{(nl)}\cdot W_{ji}^{nl-1}\cdot f'\left(z_i^{nl-1}\right)\right) \\
&= \left(\sum_{j=1}^{s_{nl}} \delta_j^{(nl)}\cdot W_{ji}^{nl-1}\right)\cdot f'\left(z_i^{nl-1}\right)
\end{aligned}
$$

因此有：

$$
\delta_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} \delta_j^{(l+1)}\cdot W_{ji}^{(l)}\right)\cdot f'\left(z_i^{(l)}\right)
$$

对于神经网络中的权重和偏置的更新公式为：

$$
\frac{\partial}{\partial W_{ij}^{(l)}} J\left(\mathbf{W},\mathbf{b};\mathbf{x},y\right) = a_j^{(l)}\delta_i^{(l+1)}
$$

$$
\frac{\partial}{\partial b_i^{(l)}} J\left(\mathbf{W},\mathbf{b};\mathbf{x},y\right) = \delta_i^{(l+1)}
$$

2.6、神经网络的学习过程

对于神经网络的学过程，大致分为如下的几步：

- 初始化参数，包括权重、偏置、网络层结构，激活函数等等
- 循环计算
    - 正向传播，计算误差
    - 反向传播，调整参数
- 返回最终的神经网络模型

参考文献

1、英文版：UFLDL Tutorial

2、中文版：UFLDL教程