这一周我把重点放在了对算法概念及神经网络基本概念的学习上，对神经网络有了更深刻的理解，代码实现方面，我完成了theano安装，Anaconda3，MinGW，和theano的安装

1. 人工神经元( Artificial Neuron )模型

       人工神经元是神经网络的基本元素，其原理可以用下图表示：

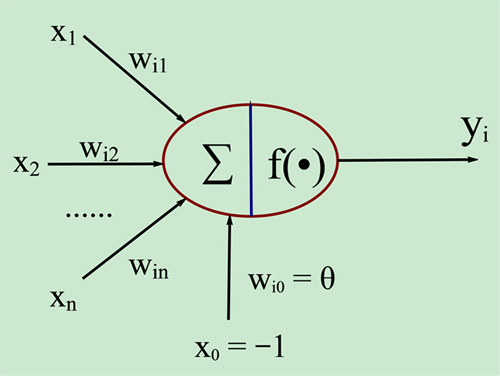


图1. 人工神经元模型

       图中x1~xn是从其他神经元传来的输入信号，wij表示表示从神经元j到神经元i的连接权值，**θ**表示一个阈值 ( threshold )，或称为偏置( bias )。则神经元i的输出与输入的关系表示为：

http://pic002.cnblogs.com/images/2011/63234/2011030721501442.png

http://pic002.cnblogs.com/images/2011/63234/2011030721502781.png

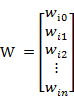
图中 yi表示神经元i的输出，函数f称为**激活函数** ( Activation Function )或转移函数 ( Transfer Function ) ，net称为净激活(net activation)。若将阈值看成是神经元i的一个输入x0的权重wi0，则上面的式子可以简化为：

http://pic002.cnblogs.com/images/2011/63234/2011030721504430.png

http://pic002.cnblogs.com/images/2011/63234/2011030721505337.png

　　若用X表示输入向量，用W表示权重向量，即：

X = [ x0 , x1 , x2 , ....... , xn ]



　　则神经元的输出可以表示为向量相乘的形式：

http://pic002.cnblogs.com/images/2011/63234/2011030721514231.png

http://pic002.cnblogs.com/images/2011/63234/2011030721520023.png

       若神经元的净激活net为正，称该神经元处于激活状态或兴奋状态(fire)，若净激活net为负，则称神经元处于抑制状态。

       图1中的这种“阈值加权和”的神经元模型称为M-P模型 ，也称为神经网络的一个处理单元。

**2. 常用激活函数**

       激活函数的选择是构建神经网络过程中的重要环节，下面我又了解到了几个常用的激活函数。

**(1) 线性函数 ( Liner Function )**

http://pic002.cnblogs.com/images/2011/63234/2011030721523837.png

**(2) 斜面函数 ( Ramp Function )**

http://pic002.cnblogs.com/images/2011/63234/2011030721525491.png

**(3) 阈值函数 ( Threshold Function )**

http://pic002.cnblogs.com/images/2011/63234/2011030721531558.png

       以上3个激活函数都属于线性函数，下面是两个常用的非线性激活函数。

**(4) S形函数 ( Sigmoid Function )**

http://pic002.cnblogs.com/images/2011/63234/2011030722052121.png

　　该函数的导函数：

http://pic002.cnblogs.com/images/2011/63234/2011030722054030.png

**(5) 双极S形函数**

http://pic002.cnblogs.com/images/2011/63234/2011030722060475.png

　　该函数的导函数：

http://pic002.cnblogs.com/images/2011/63234/2011030722062274.png

　　S形函数与双极S形函数的图像如下：

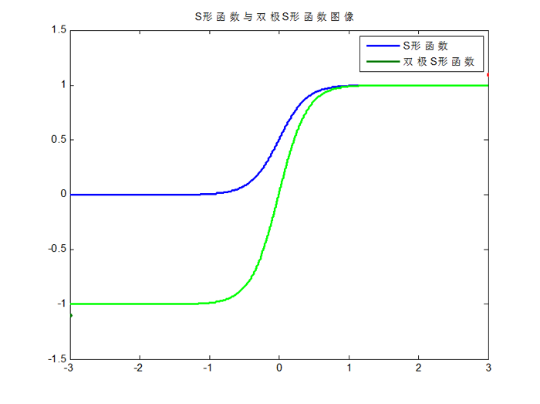


图3. S形函数与双极S形函数图像

　　双极S形函数与S形函数主要区别在于函数的值域，双极S形函数值域是(-1,1)，而S形函数值域是(0,1)。

　　由于S形函数与双极S形函数都是可导的(导函数是连续函数)，因此适合用在BP神经网络中。（BP算法要求激活函数可导）

**3. 神经网络模型**

       神经网络是由大量的神经元互联而构成的网络。根据网络中神经元的互联方式，常见网络结构主要可以分为下面３类：

**(1) 前馈神经网络　(　Feedforward Neural Networks )**

       前馈网络也称前向网络。这种网络只在训练过程会有反馈信号，而在分类过程中数据只能向前传送，直到到达输出层，层间没有向后的反馈信号，因此被称为前馈网络。感知机( perceptron)与BP神经网络就属于前馈网络。

       图4 中是一个3层的前馈神经网络，其中第一层是输入单元，第二层称为隐含层，第三层称为输出层（输入单元不是神经元，因此图中有2层神经元）。

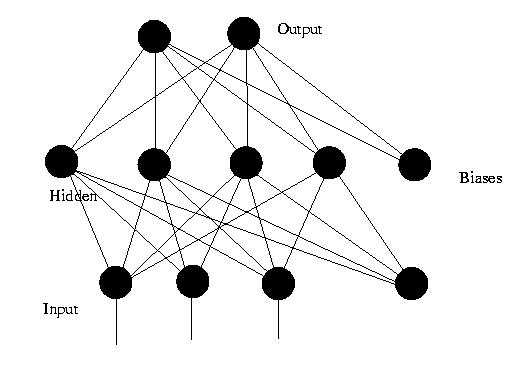


图4. 前馈神经网络

　　对于一个3层的前馈神经网络N，若用X表示网络的输入向量，W1~W3表示网络各层的连接权向量，F1~F3表示神经网络3层的激活函数。

　　那么神经网络的第一层神经元的输出为：

O1 = F1( XW1 )

　　第二层的输出为：

O2 = F2 ( F1( XW1 ) W2 )

　　输出层的输出为：

O3 = F3( F2 ( F1( XW1 ) W2 ) W3 )

       若激活函数F1~F3都选用线性函数，那么神经网络的输出O3将是输入X的线性函数。因此，若要做高次函数的逼近就应该选用适当的非线性函数作为激活函数。

**(2) 反馈神经网络　(　Feedback Neural Networks )**

       反馈型神经网络是一种从输出到输入具有反馈连接的神经网络，其结构比前馈网络要复杂得多。典型的反馈型神经网络有：Elman网络和Hopfield网络。

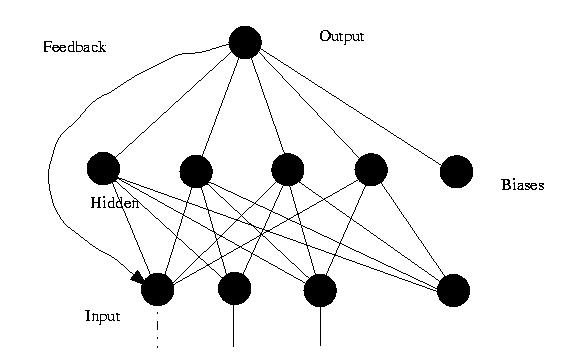


图5. 反馈神经网络

**(3) 自组织网络 ( SOM ,Self-Organizing Neural Networks )**

       自组织神经网络是一种无导师学习网络。它通过自动寻找样本中的内在规律和本质属性，自组织、自适应地改变网络参数与结构。

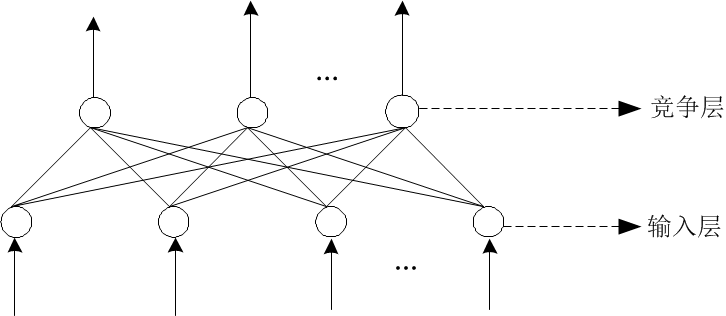


图6. 自组织网络

**4. 神经网络工作方式**

       神经网络运作过程分为学习和工作两种状态。

**(1)神经网络的学习状态**

       网络的学习主要是指使用学习算法来调整神经元间的联接权，使得网络输出更符合实际。学习算法分为**监督学习( Supervised Learning )**与**非监督学习( Unsupervised Learning )**两类。

**监督学习**算法将一组训练集 ( training set )送入网络，根据网络的实际输出与期望输出间的差别来调整连接权。监督学习算法的主要步骤包括：

1）  从样本集合中取一个样本（Ai，Bi）；

2）  计算网络的实际输出O；

3）  求D=Bi-O；

4）  根据D调整权矩阵W；

5） 对每个样本重复上述过程，直到对整个样本集来说，误差不超过规定范围。

　　BP算法就是一种监督学习算法。

**无监督学习**抽取样本集合中蕴含的统计特性，并以神经元之间的联接权的形式存于网络中。

       Hebb学习律是一种经典的无监督学习算法。

**(2) 神经网络的工作状态**

       神经元间的连接权不变，神经网络作为分类器、预测器等使用。

　　下面简要介绍一下Hebb学习率与Delta学习规则 。

**(3) 无导师学习算法：Hebb学习率**

　　Hebb算法核心思想是，当两个神经元同时处于激发状态时两者间的连接权会被加强，否则被减弱。

       为了理解Hebb算法，有必要简单介绍一下条件反射实验。巴甫洛夫的条件反射实验：每次给狗喂食前都先响铃，时间一长，狗就会将铃声和食物联系起来。以后如果响铃但是不给食物，狗也会流口水。

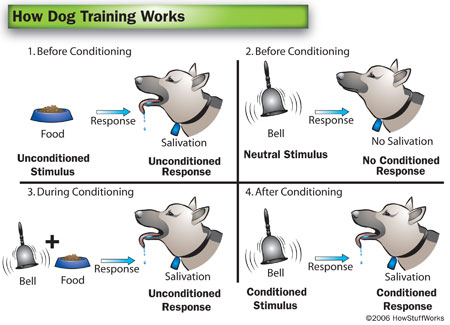


图7. 巴甫洛夫的条件反射实验

　　受该实验的启发，Hebb的理论认为在同一时间被激发的神经元间的联系会被强化。比如，铃声响时一个神经元被激发，在同一时间食物的出现会激发附近的另一个神经元，那么这两个神经元间的联系就会强化，从而记住这两个事物之间存在着联系。相反，如果两个神经元总是不能同步激发，那么它们间的联系将会越来越弱。

　　Hebb学习律可表示为：

http://pic002.cnblogs.com/images/2011/63234/2011030722144483.png

       其中wij表示神经元j到神经元i的连接权，yi与yj为两个神经元的输出，a是表示学习速度的常数。若yi与yj同时被激活，即yi与yj同时为正，那么Wij将增大。若yi被激活，而yj处于抑制状态，即yi为正yj为负，那么Wij将变小。

**(4) 有导师学习算法：Delta学习规则**

　　Delta学习规则是一种简单的有导师学习算法，该算法根据神经元的实际输出与期望输出差别来调整连接权，其数学表示如下：

http://pic002.cnblogs.com/images/2011/63234/2011030722150043.png

       其中Wij表示神经元j到神经元i的连接权，di是神经元i的期望输出，yi是神经元i的实际输出，xj表示神经元j状态，若神经元j处于激活态则xj为1，若处于抑制状态则xj为0或－1（根据激活函数而定）。a是表示学习速度的常数。假设xi为1，若di比yi大，那么Wij将增大，若di比yi小，那么Wij将变小。

       我对Delta规则的理解是：若神经元实际输出比期望输出大，则减小所有输入为正的连接的权重，增大所有输入为负的连接的权重。反之，若神经元实际输出比期望输出小，则增大所有输入为正的连接的权重，减小所有输入为负的连接的权重。这个增大或减小的幅度就根据上面的式子来计算。

由于本周回家之后要忙一些事情，所以周报未在周二按时上交。