人工智能实验报告

----BP 神经网络在函数拟合和手写数字识别中的应用

姓名：姚文韬　　学号: 13351061　日期：2016.1.1

摘要

作为当前人工智能研究最热的方向之一，人工神经网络具有不可估量的前景。大规模的非线性自适应结构，使其拥有较好的学习能力和较好的鲁棒性，从而使其拥有类似于人的简单判断和识别功能。本次实验尝试将人工神经网络中较为基础的 BP 神经网络运用在简单函数拟合以及手写数字识别上，在将算法实例化的过程当中尝试更好地去理解人工神经网络。

1. 导言

1.1 问题背景介绍

函数拟合是数学建模与数值计算当中的重要环节。在解决实际数学问题时，我们常常需要对数据集进行函数拟合，以预测未来的数据走向；同时由于不少在纯数学模型当中的函数在实际生产中难以实现，因此我们需要利用函数拟合对一些复杂函数进行简单近似，以简化生产流程。

手写识别常常被当成学习神经网络的原型问题，因此我们聚焦在这个问题上。作为一个原型，它具备一个关键点：挑战性——识别手写数字并不轻松——但也不会难到需要极其复杂的解决方法，或者超大规模的计算资源。另外，这其实也是一种发展出诸如深度学习更加高级的技术的方法。

1.2 方法背景介绍

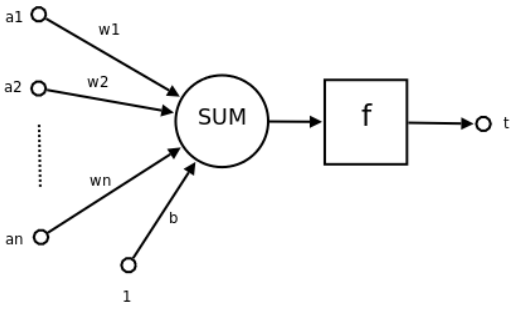
本实验采用误差反向传播（Back Propagation, BP）人工神经网络实现，其由大量的处理单元（即神经元）广泛的互相连接而形成，亦称为高度复杂的非线性动力学系统。其对人脑进行某种简化、抽象和模拟，反映了人脑许多基本特性，并且具备一定的判断识别功能。

2. 实验过程与结果描述

2.1 基本算法阐述

2.1.1 人工神经网络的基本组成单元：神经元

人工神经网络是由模拟神经元广泛互联而形成的系统，所谓的模拟神经元是指对生物神经元的抽象和模拟，包括对结构以及功能上的模拟。在网络当中，人工神经元又称为处理单元，其能对一个或多个输入进行处理，经过神经元功能函数 f 的计算后并经过阈值的判断后得到输出。在本实验中，神经元函数采用 Sigmoid 函数。



2.1.2 人工神经网络的基本层次结构

输入层（Input layer），众多神经元（Neuron）接受大量非线形输入信息。输入的信息称为输入向量。

隐层（Hidden layer），是输入层和输出层之间众多神经元和链接组成的各个层面。隐层可以有多层，本实验中采用一层。隐层的神经元数目不定，但数目越多神经网络的非线性越显著，从而神经网络的鲁棒性更显著。

输出层（Output layer），信息在神经元链接中传输、分析、权衡，形成输出结果。输出的信息称为输出向量。

同时，每一层之间相互连接的网络称为权值网络，其作用在于调节各个输入的比重，使得不同输入经过神经元后可以得到理想输出。

2.1.3 前馈网络以及误差反向传播（BP）算法

在以上所描述的网络当中，输入到输出中间不形成回路，计算出的结果一直前馈至输出层，因此称为前馈网络。而输出层输出与目的输出之间的误差将逐级反馈回每层网络，并通过梯度下降法调整权值网络，经过足够的迭代次数后得到理想的训练网络，因此称为误差反向传播算法。本实验中采用该算法以及加入动量项后的 BP 改进算法，其实现步骤如下：

1. 设置变量和参量 :令$\boldsymbol{x_k}$为输入向量(或称训练样本) ;N为训练样本的个数；两个权值矩阵$W_{MI}(n), W_{IP}(n)$，分别为输入层到隐层，以及隐层到输出层

的第n次迭代后的权值矩阵；$\boldsymbol{Y_k(n)}$为第n次迭代后网络的实际输出；$\boldsymbol{d_k(n)}$为期望输出；η为学习速率，n为迭代次数；

2. 初始化权值网络，赋给$W_{MI}(n), W_{IP}(n)$均匀分布的经验值，在$(-\frac{2.4}{\mathrm{F}}, \frac{2.4}{\mathrm{F}})$或$\left(-\frac{3}{\sqrt{F}}, \frac{3}{\sqrt{F}}\right)$之间，其中$F$为所连单元的输入端个数；
3. 输入训练样本$\boldsymbol{x_k}$，并置$n=0$；
4. 对于输入样本$\boldsymbol{x_k}$，前向计算 BP 网络每层神经元的输入信号u和输出信号v，其中$v_p^P(n)=y_{kp}(n)=1/(1+\exp(-u_p^P(n))),\ p=1,2,\ldots,P$;
5. 由期望输出$\boldsymbol{d_k}$和上一步求得的实际输出$y_{kp}(n)$计算误差$E(n)$，判断其是否满足要求，若满足转至第八步，若不满足转至第六步；
6. 判断第$n+1$步是否大于最大迭代次数，若大于转至第八步，若不大于则对输入样本$\boldsymbol{x_k}$反向计算每层神经元的局部梯度δ，其中

$$\delta_p^P(n)=y_p(n)\left(1-y_p(n)\right)\left(d_p(n)-y_p(n)\right)\quad p=1,2,\ldots,\mathrm{P}$$

$$\delta_i^I(n)=f'(u_i^I(n))\sum_{i=1}^{I}\delta_p^P(n)w_{ip}(n)\qquad i=1,2,\ldots,\mathrm{I}$$

7. 按照如下方法计算权值修正量Δw，并修正权值；$n=n+1$，转至第四步；

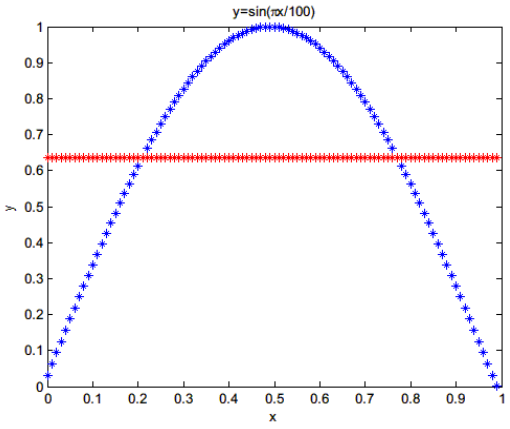
$$\Delta w_{ip}(n)=\eta\delta_p^P(n)v_i^I(n)\qquad w_{ip}(n+1)=w_{ip}(n)+\Delta w_{ip}(n)$$

$$\Delta w_{mi}(n)=\eta\delta_i^I(n)x_{km}(n)\qquad w_{mi}(n+1)=w_{mi}(n)+\Delta w_{mi}(n)$$

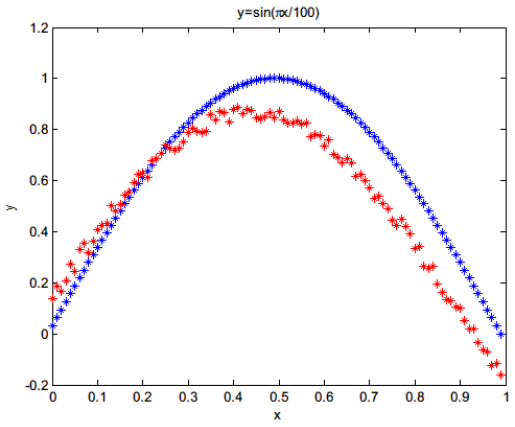
8. 判断是否学习完所有训练样本，若是则结束算法，否则转至第三步。

2.2 算法实现：函数拟合

在本次实验中采用了$y=\sin(\pi x)$作为拟合函数，给出一个随机数的序列，以函数的对应值作为期望输出，按照基本算法在 matlab 中建立人工神经网络（不采用工具箱）。当学习速率低至 0.01，并且迭代次数为 1000 时，有以下结果：



可以看出原始的拟合效果非常差，于是调整学习速率至 0.1，加入偏置b以及动量项，得到以下拟合结果：



可以看出此时的拟合结果较为接近函数值。

2.3 算法实现：手写数字识别

在利用 BP 神经网络实现函数拟合后，尝试进行手写数字的识别，流程如下。

2.3.1 图像预处理

本次实验采用的图像来自手写数字识别的标准库，均为灰度图像。若使用彩色图像作为输入，需要先进行灰度化和直方图均衡化。不同于人类视觉，机器视觉的本质是图像的像素矩阵的处理和运算，进而得到人们需要的结果；而有时我们仅仅使用灰度图像的信息就已经足够了。因此，为了加快运算速度，应对图像进行灰度化处理，将原图像矩阵变化为每个元素都位于 0-255 灰度值之间的一个灰度矩阵。而采用直方图均衡化的方法处理灰度图像是把原始图像的灰度直方图从比较集中的某个灰度区间变成在全部灰度范围内的均匀分布，即对图像进行非线性拉伸，重新分配图像像素值，使一定灰度范围内的像素数量大致相同，使得给定图像的直方图分布改变成“均匀”分布直方图分布。这种做法可以改善对比度，使得图像曝光均匀并增强局部的对比度。

2.3.2 特征提取

特征提取的目的在于减少网络的输入量以提高运算效率。本次实验采用的图像为 640*640 像素，每个图像有 10*10 个数字，因此每个数字大小为 64*64 像素。若将每个数字的全部像素放入网络中运算，运算速率将非常低。因此采用了特征提取的办法，计算每个数字的灰度矩阵中 15 个特定行

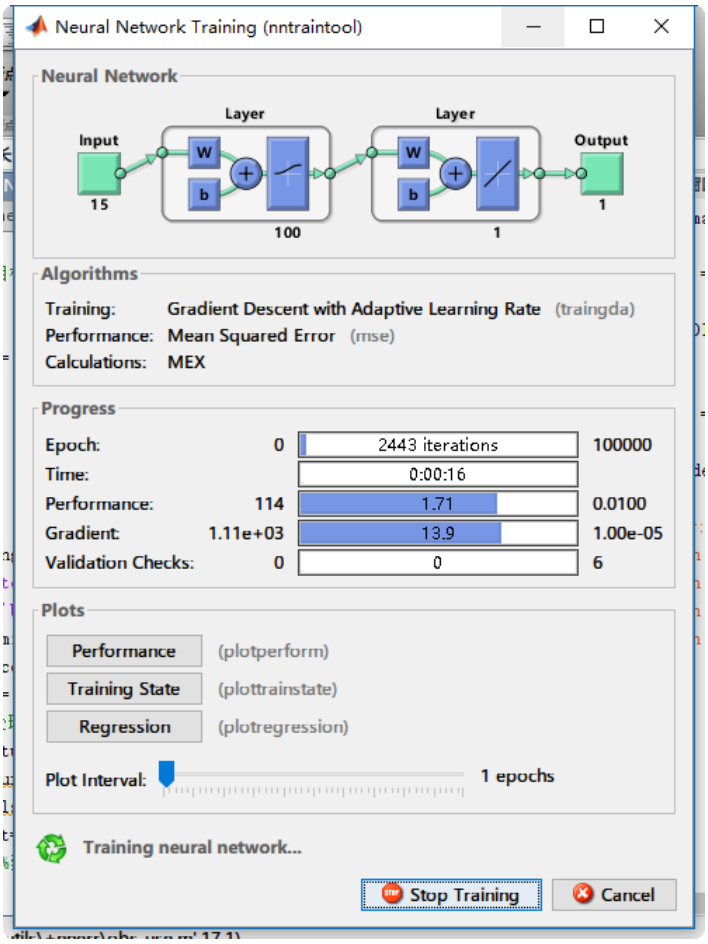
或列的像素值之和作为网络层的 15 个输入。

2.3.3 训练样本与识别样本

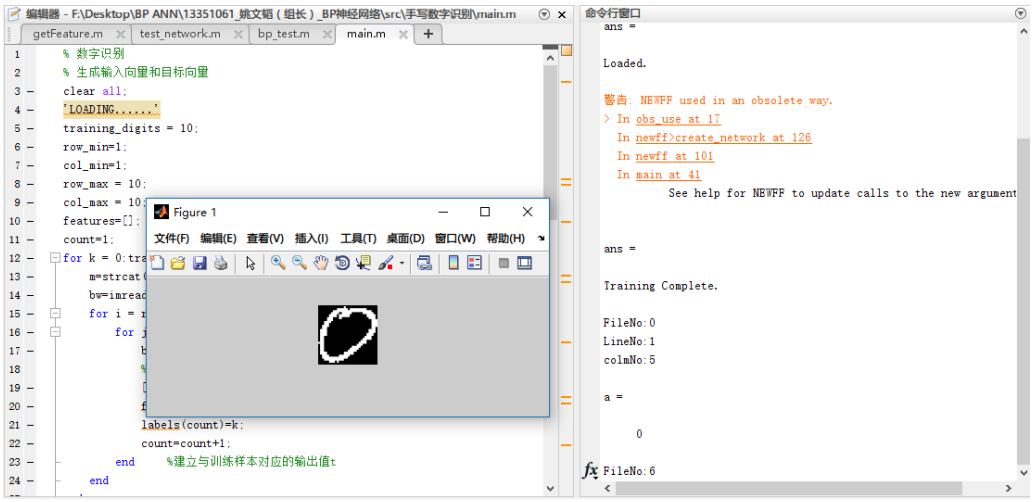
将图像进行分割和预处理后提取特征，将 15 个特征值作为一个数字的训练样本送入网络，以数字的实际值作为期望输出，按照以上的基本算法在 matlab 中建立人工神经网络（不采用工具箱），并进行多次训练和迭代。训练完毕后，检验识别效果。

2.3.4 效果检验

使用初始算法进行的网络训练未能准确识别数字；不幸的是，经过特征处理和算法优化后（加入偏置以及动量项），仍未能得到较好的结果。于是采用 matlab 工具箱进行操作，采用不同的权值调整算法，试图找出一个较好的识别效果的算法。经过研究对比，发现共轭最速梯度下降法收敛速度最快，学习效果最佳。但由于时间关系和水平有限，未能再都该进算法，只能用 matlab 工具箱实现手写数字识别。



正在训练的网络



识别效果。输入文件名称以及行列数，可以识别该数字的数值并显示图像。

3. 总结与展望

本次实验初步学习了 BP 人工神经网络，基本实现了函数拟合，但对于更复杂的问题如手写数字识别以及人脸识别，基本的梯度下降法不能满足收敛速度和学习精度的要求，有待进一步的算法优化；并且这也将我们引入了对于深度学习的初步认识，因为深度学习在人工神经网络上的优势非常瞩目。以后若有机会，会尝试编写深度学习的人工神经网络来实现以上功能，并比较优劣。