**智能系统Lab1 Part1**

18302010026 吕昌泽

一、代码基本结构

BPNetwork类

初始化方法：传入指定的参数，构造不同结构的神经网络，实现了“可伸缩，易调整”



forwardPropagate方法：传入输入变量（输入层向量），经过神经网络的计算后，返回输出（输出层向量）。

backPropagate方法：反向传播，记录每层的delta，沿着权重反向富集，以此逐层更新网络的权重和bias。

train方法：传入输入以及其对应的输出（标签），与epoch次数（默认为1000），进行训练。



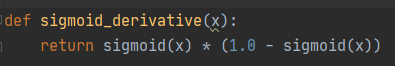
初始化方法

类外的全局方法：

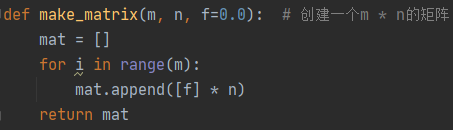
sigmoid函数：



sigmoid的导函数：



创建一个指定大小的矩阵：



二、不同网络结构、网络参数的实验比较

1）拟合sinx任务：

在[-pi，pi]到上随机取1000个样本点，一遍epoch对这1000个样本点进行训练，正向传播一遍，之后反向调整误差与bias。

注意：权重和bias的初始值都从小的开始取，为使网络收敛，bias默认为负值；weight decay用于防止过拟合

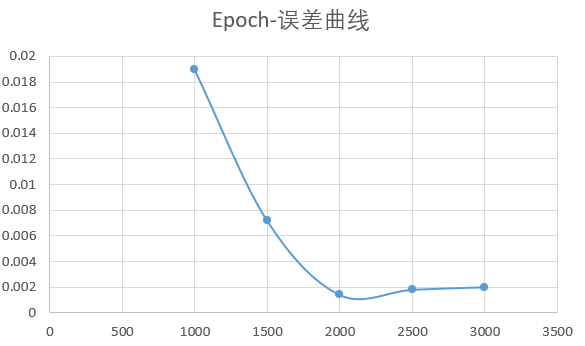
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验编号 | 输入结点个数 | 隐层层数 | 隐层结点个数 | 输出结点个数 | 权重初始值 | Bias初始值 | Weight Decay | 学习率 | Epoch | 拟合总误差 |
| 1 | 1 | 1 | 10 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.042 |
| 2 | 1 | 20 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.023 |
| 3 | 1 | 30 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.011 |
| 4 | 1 | 40 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.007 |
| 5 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.0014 |
| 6 | 1 | 60 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.005 |
| 7 | 1 | 2 | 10 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.057 |
| 8 | 1 | 20 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.043 |
| 9 | 1 | 30 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.029 |
| 10 | 1 | 40 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.011 |
| 11 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.009 |
| 12 | 1 | 60 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.013 |
| 13 | 1 | 3 | 10 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.137 |
| 14 | 1 | 20 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.096 |
| 15 | 1 | 30 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.026 |
| 16 | 1 | 40 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.024 |
| 17 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.022 |
| 18 | 1 | 60 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 2000 | 0.037 |

从实验1-18确定，隐藏层最好为**1层，50个结点**。

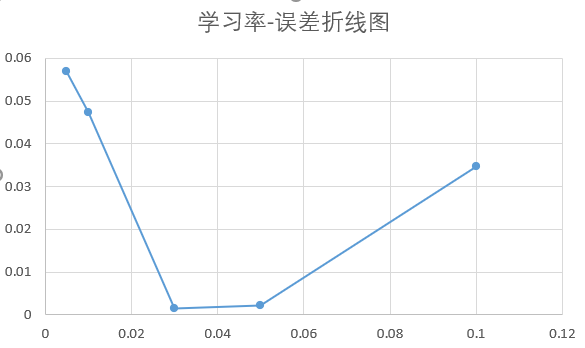
以下实验针对权重初始值、bias初始值、weight decay、学习率、epoch进行了探究

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验编号 | 输入结点个数 | 隐层层数 | 隐层结点个数 | 输出结点个数 | 权重初始值 | Bias初始值 | Weight Decay | 学习率 | Epoch | 拟合总误差 |
| 19 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 1000 | 0.019 |
| 20 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 1500 | 0.0072 |
| 21 | 1 | 1 | 50 | 1 | [0, 0.002] | [-0.001,1] | 0 | 0.03 | 2500 | 0.0054 |
| 22 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.03 | 3000 | 0.002 |
| 23 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.005 | 2000 | 0.0569 |
| 24 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.01 | 2000 | 0.0473 |
| 25 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.05 | 2000 | 0.0022 |
| 26 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0 | 0.1 | 2000 | 0.0347 |
| 27 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.0001,0] | 0 | 0.03 | 2000 | 0.0114 |
| 28 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.01,0] | 0 | 0.03 | 2000 | 0.0127 |
| 29 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.1,0] | 0 | 0.03 | 2000 | 0.0259 |
| 30 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0.00001 | 0.03 | 2000 | 0.0016 |
| 31 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0.0001 | 0.03 | 2000 | 0.048 |
| 32 | 1 | 1 | 50 | 1 | [0, 0.001] | [-0.001,0] | 0.001 | 0.03 | 2000 | 0.103 |
| 33 | 1 | 1 | 50 | 1 | [0, 0.0001] | [-0.001,0] | 0 | 0.03 | 2000 |  |
| 34 | 1 | 1 | 50 | 1 | [0, 0.01] | [-0.001,0] | 0 | 0.03 | 2000 | 0.023 |
| 35 | 1 | 1 | 50 | 1 | [0,0.1] | [-0.001,0] | 0 | 0.03 | 2000 | 0.37 |
| 36 | 1 | 1 | 50 | 1 | [0, 1] | [-0.001,0] | 0 | 0.03 | 2000 | 1.63 |

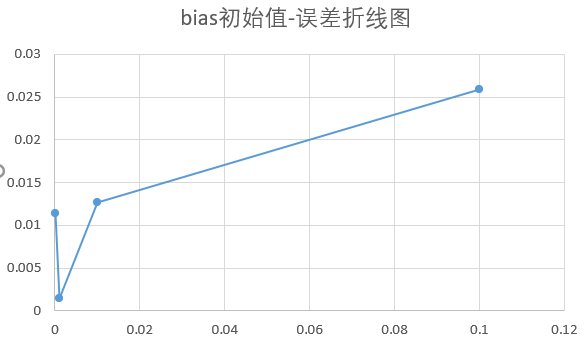
由实验5、19-22可以得出：epoch次数为2000次为宜，2500-3000次区别不大，1000次训练不充分。这个结果是显而易见的，当训练次数增加，误差必然在一定程度内先减小后增加，增加是因为存在过拟合。



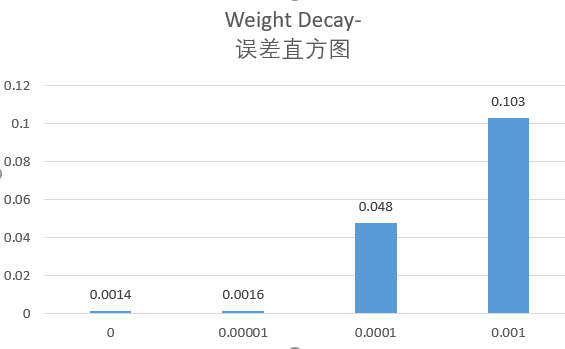
由实验5、23-26可以得出：学习率0.03为最佳。



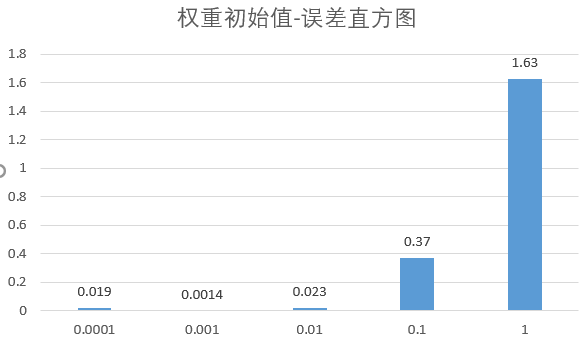
由实验5、27-29可以得出，bias的初值应该取在0.001量级，所以我取了[-0.001, 0]



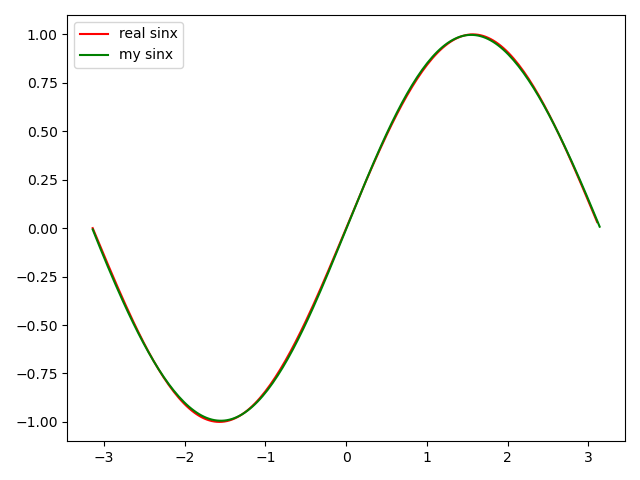
由实验5、30-32可以得出，weight decay取0最好。原因估计是数据量太少，不太会出现过拟合的情形。



由实验5、33-36可以得出，权重初始值为0.001量级最佳。



附上实验5所得截图：



2）识别手写图片任务：

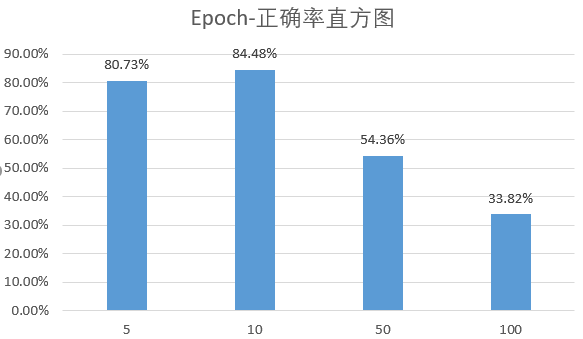
为了控制变量，每次实验我都选取了12 \* 510张图片作为训练集，12 \* 110张图片作为测试集。输入为28 \* 28 = 784 维的向量，输出为12维的向量。

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验编号 | 输入结点个数 | 隐层层数 | 隐层结点个数 | 输出结点个数 | 权重初始值 | Bias初始值 | Weight Decay | 学习率 | Epoch | 测试集识别正确率 |
| 1 | 784 | 1 | 100 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 54.81% |
| 2 | 784 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 84.48% |
| 3 | 784 | 400 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 38.74% |
| 4 | 784 | 2 | 100 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 67.83% |
| 5 | 784 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 62.91% |
| 6 | 784 | 400 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 8.33% |
| 7 | 784 | 3 | 100 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 8.33% |
| 8 | 784 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | 8.33% |
| 9 | 784 | 400 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 10 | NAN |

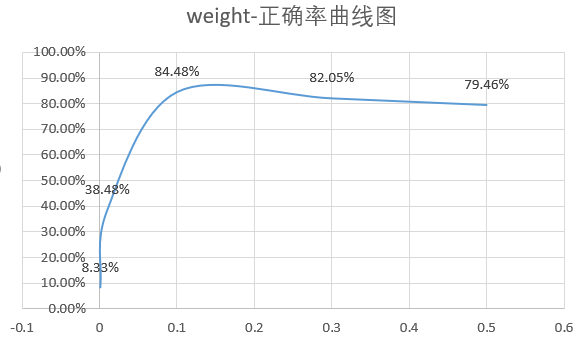
由实验1-9可以看出，网络结构为1层隐藏层，200结点为最佳。实验9训爆了，浮点数溢出。1-9说明，当网络太过复杂（实验6、7、）时，网络只会记住最后一次训练的类别的结果，使得最后只能分出一类，即正确率一直是1/12 = 8.33%

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验编号 | 输入结点个数 | 隐层层数 | 隐层结点个数 | 输出结点个数 | 权重初始值 | Bias初始值 | Weight Decay | 学习率 | Epoch | 测试集识别正确率 |
| 10 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 5 | 80.73% |
| 11 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 50 | 54.36% |
| 12 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.01 | 100 | 33.82% |
| 13 | 784 | 1 | 200 | 12 | [-0.01, 0.01] | [-0.1, 0] | 0 | 0.01 | 10 | 38.48% |
| 14 | 784 | 1 | 200 | 12 | [-0.001, 0.001] | [-0.1, 0] | 0 | 0.01 | 10 | 8.33% |
| 15 | 784 | 1 | 200 | 12 | [-1, 1] | [-0.1, 0] | 0 | 0.01 | 10 | 73.48% |
| 16 | 784 | 1 | 200 | 12 | [-0.3, 0.3] | [-0.1, 0] | 0 | 0.01 | 10 | 82.05% |
| 17 | 784 | 1 | 200 | 12 | [-0.5, 0.5] | [-0.1, 0] | 0 | 0.01 | 10 | 79.46% |
| 18 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0.0001 | 0.01 | 10 | 70.90% |
| 19 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0.001 | 0.01 | 10 | 54.78% |
| 20 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0.01 | 0.01 | 10 | 12.73% |
| 21 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-1, 0] | 0 | 0.01 | 10 | 75.32% |
| 22 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.01, 0] | 0 | 0.01 | 10 | 40.00% |
| 23 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.3, 0] | 0 | 0.01 | 10 | 82.79% |
| 24 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.8, 0] | 0 | 0.01 | 10 | 75.16% |
| 25 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.1 | 10 | 68.33% |
| 26 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.001 | 10 | 27.68% |
| 27 | 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-0.1, 0] | 0 | 0.03 | 10 | 81.59% |

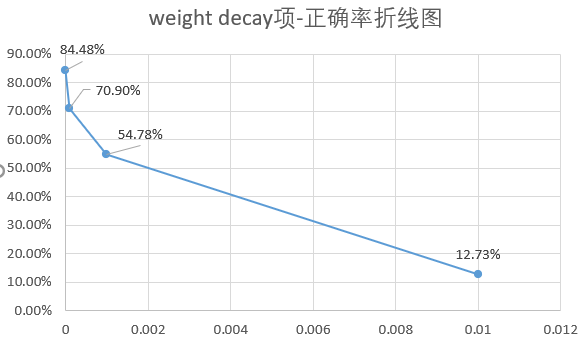
由实验2、10-12可以看出，训练次数epoch10次为最佳，5次和10次差别不大，50次与100次会导致过拟合，使正确率下降巨大。



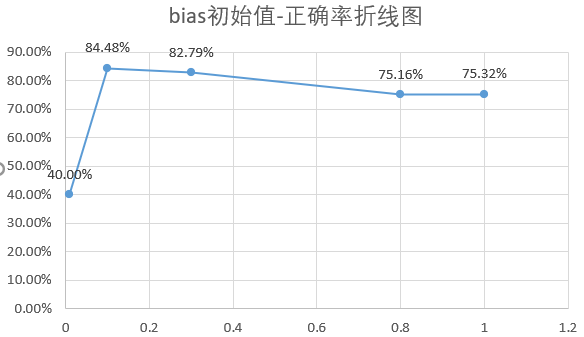
由实验2、13-17可以看出，初始的weight在0.1量级为最佳，[-0.1, 0.1]的正确率比[-0.3, 0.3]和[-0.5, 0.5]更加高，以此确定最佳的初始权重为[-0.1, 0.1]。这中间可能存在取随机数的偶然性，因为维持在0.1量级左右，正确率就可以大于80%



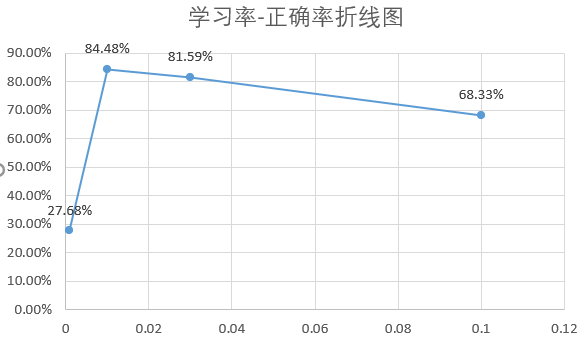
由实验4、18-20看出，我增加weight decay项，防止它过拟合，反而导致了正确率大幅下降。个人认为这是由于weight decay项用于过拟合的条件是epoch次数应该很大、训练数据很多，但是我的epoch只有2000，则decay项没有起到预期的效果。



由实验4、21-24可以看出，bias的初始值在-0.1量级最佳。与weight同理。



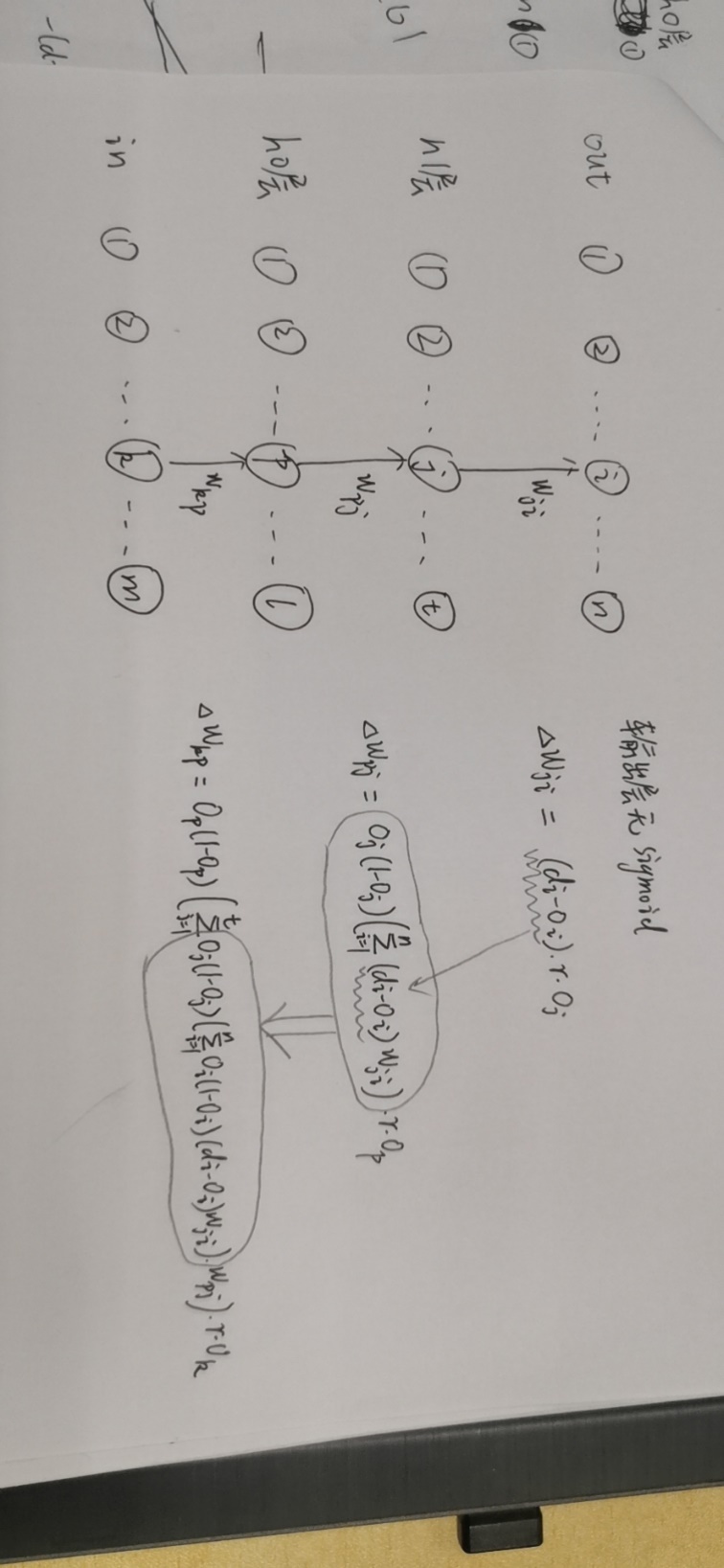
由实验4、25-27得出：在epoch为10次的条件下，学习率为0.01量级，0.03和0.01差别不大。0.001会导致训练不充分，0.1无法完全收敛。



三、对反向传播算法的理解

反向传播算法就是通过计算最终误差与某个参数的梯度，来更新参数。例如，误差对权重的梯度，可以用于更新权重；误差对bias的梯度可以用于更新bias。

BP神经网络的核心是“链式法则”。它使得梯度可以沿着网络反向“富集”，每一层的权重、bias都可以通过对上一层的相关梯度迭代计算得到。



BP神经网络的优点在于：

1) 非线性映射能力：数学理论上，三层的神经网络就能够以任意精度逼近任何非线性连续函数。

2) 自学习和自适应能力：BP神经网络在训练时，能够通过学习自动提取输出、输出数据间的“合理规则”，并自适应的将学习内容记忆于网络的权值中。

3) 容错能力：在样本有极少错误的情况下，对预测的结果没有太大的影响。比如本实验的图片分类任务，“博”中混了个“学”，但是识别率还是可以高达83.48%

缺点在于：

1) 容易陷入局部极小值：梯度下降使得调整权重和bias成为可能。但是如果陷入局部极小值，而不是全局最小值，会使得回归、分类任务的误差变得很大。例如，在训练sinx函数的时候，由于初始权重取值的不同，得到的曲线有时会非常“平坦”，甚至于像一条直线。这就是陷入了局部极小值

2)收敛速度慢：由于BP神经网络算法本质上为梯度下降法，它所要优化的目标函数是非常复杂的，因此，必然会出现“锯齿形现象”。在误差极小或极小的情况下，它必然会在神经元输出接近0或1的情况下，出现一些平坦区，在这些区域内，权值误差改变很小，使训练过程几乎停顿。

3)神经网络结构选择尚无法则：BP神经网络结构的选择目前只能由经验选定。网络结构选择过大，训练中效率不高，可能出现过拟合现象，造成网络性能低，容错性下降；若选择过小，则又会造成网络可能不收敛。

四、小Trick

1）手动去噪：例如，“博”字里面混了个“学”字，那我就不把他作为训练集的一部分。此外还有一些肉眼难以辨别的字，都不将其作为训练数据。当然，这样子无意义。

2)第一遍epoch的时候，学习率取大一点，使得函数更容易收敛与全局最小值；之后的epoch再调小学习率。

这个方法在这次实验中并没有明显提高效率：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 输入结点个数 | 隐层层数 | 隐层结点个数 | 输出结点个数 | 权重初始值 | Bias初始值 | Weight Decay | 学习率  (第一次学习率/之后的学习率) | Epoch | 测试集识别正确率 |
| 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-1.0, 0] | 0 | 0.03/0.008 | 10 | 82.50% |
| 784 | 1 | 200 | 12 | [-0.1, 0.1] | [-1.0, 0] | 0 | 0.05/0.01 | 10 | 82.72% |