

上机实验报告

|  |  |
| --- | --- |
| 题目： | 运用多层感知神经网络识别手写数字 |

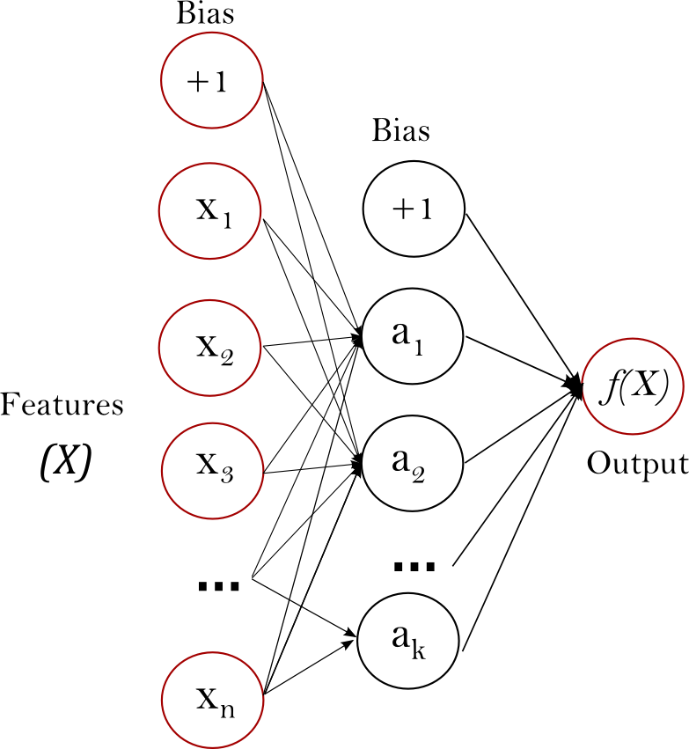
|  |  |
| --- | --- |
| 姓 名： | **李梓焓** |
| 学 号： | **1700011735** |
| 组 别： | **无** |
| 实验日期： | **2019.10.25** |

**摘要** 本实验运用多层感知神经网络，采用MMIST手写数字数据集进行训练，并对其进行测试，计算相应的混淆矩阵、准确率、召回率，从而得出结论：（1）仅用一层隐藏层的多层感知神经网络模型，其准确率和召回率达90%以上，基本满足手写数字识别的需求；（2）增大单层的节点数，能改善多层感知神经网络模型的表现，但训练时相应的计算量和计算时间也会增加；（3）在其他条件不变的情况下，改变最大迭代次数，对多层感知神经网络模型的表现没有明显改善；（4）在其他条件不变的情况下，改变学习速率，对多层感知神经网络模型的表现具有两面性：当学习速率从0.0001增大到0.001时，模型的准确率和召回率均有少量提升，当学习速率从0.001继续增大时，模型的准确率和召回率开始出现明显下滑；（5）在其他条件不变的情况下，若学习速率为0.0001，则采用多个隐藏层时，对多层感知神经网络模型的表现没有明显改善，反而导致了准确率和召回率的少量下降；而若学习速率为0.001，则采用多个隐藏层时，多层感知神经网络模型的表现先是出现了一定改善，随层数增加，改善效果逐渐下降。

**关键词** 多层感知神经网络，识别，手写数字

1. **引言**

多层感知器(MLP) 是一种监督学习算法，通过在数据集上训练来学习函数 ，其中是输入的维数，是输出的维数。给定一组特征 和标签，它可以学习用于分类或回归的非线性函数。与逻辑回归不同的是，在输入层和输出层之间，可以有一个或多个非线性层，称为隐藏层。下图展示了一个具有标量输出的单隐藏层 MLP。



最左层的输入层由一组代表输入特征的神经元 组成。每个隐藏层中的神经元将前一层的值进行加权线性求和转换，再通过非线性激活函数，如双曲正切函数tanh。输出层接收到的值由最后一个隐藏层的输出经变换得到。

本次上机实验采用作业附带的mlc\_mnist.py，从MMIST手写数字数据集提供的文件读入数据，然后用Scikit-learn自带的MLPClassifier，训练神经网络和测试，并分别用confusion\_matrix()、precision\_score()和recall\_score()计算混淆矩阵、准确率、召回率。

**2 实验部分**

**2.1 仪器**

2.1.1硬件

Surface Pro（第5代，处理器参数：Intel® Core™ i5-7300U CPU @ 2.60GHz，2.71 GHz，2个内核，4个逻辑处理器；内存容量：8.00 GB）

2.2.2软件

操作系统：Windows 10家庭版，版本1903

开发环境：Visual Studio 2019 Community（已预装64位Miniconda 3，含64位Python 3.7.4、Conda 4.7.12、NumPy 1.16.5、Scikit-learn 0.21.2、joblib 0.13.2）

2.2.3训练和测试数据

train-images-idx3-ubyte和train-labels-idx1-ubyte：训练用手写数字和标签（均转存为二进制文件），共计6万个样本。

t10k-images-idx3-ubyte和t10k-labels-idx1-ubyte：测试用手写数字和标签（均转存为二进制文件），共计1万个样本。

**2.2 实验过程**

2.2.1训练和测试数据的读入

采用作业附带的mlc\_mnist.py，读取训练样本和测试样本的二进制文件，并转换为numpy的ndarray。

2.2.2训练多层感知神经网络

采用Scikit-learn自带的MLPClassifier，利用读入的训练样本，训练神经网络。其中求解器采用adam，激活函数采用logistic，正则化系数为，隐藏层仅用一层，包含50个节点，最大迭代数为500，学习率恒定为。

2.2.3预测测试集并评价预测结果

利用训练好的神经网络对测试样本进行预测，然后分别用confusion\_matrix()、precision\_score()和recall\_score()计算混淆矩阵、准确率、召回率。

2.2.4保存模型

采用joblib的dump()，保存训练模型。

2.2.5单层神经网络的节点数与训练效果的关系

使用固定的随机种子，重复2.2.1-2.2.4的流程，每次仅改变单层神经网络的节点数，然后计算神经网络预测结果的混淆矩阵、准确率、召回率。

2.2.6单层神经网络的最大迭代次数与训练效果的关系

使用固定的随机种子，重复2.2.1-2.2.4的流程，每次仅改变单层神经网络的最大迭代次数，然后计算神经网络预测结果的混淆矩阵、准确率、召回率。

2.2.7单层神经网络的最大迭代次数与训练效果的关系

使用固定的随机种子，重复2.2.1-2.2.4的流程，每次仅改变单层神经网络的最大迭代次数，然后计算神经网络预测结果的混淆矩阵、准确率、召回率。

2.2.8多层神经网络的层数及节点数与训练效果的关系

使用固定的随机种子，重复2.2.1-2.2.4的流程，每次仅改变多层神经网络的层数及节点数，然后计算神经网络预测结果的混淆矩阵、准确率、召回率。

**3 数据与结果分析**

**3.1固定随机状态后神经网络预测效果**

random\_state设定为1551，按2.2.1-2.2.4进行实验，得混淆矩阵，如表1所示：

**表1** 固定随机状态后神经网络预测结果的混淆矩阵

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实际\预测 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 | 965 | 0 | 3 | 1 | 0 | 2 | 6 | 0 | 2 | 1 |
| 1 | 0 | 1116 | 5 | 3 | 0 | 2 | 3 | 2 | 4 | 0 |
| 2 | 6 | 3 | 963 | 12 | 8 | 3 | 8 | 8 | 20 | 1 |
| 3 | 2 | 0 | 12 | 947 | 0 | 22 | 0 | 10 | 11 | 6 |
| 4 | 1 | 0 | 6 | 2 | 930 | 0 | 9 | 4 | 5 | 25 |
| 5 | 9 | 0 | 1 | 31 | 4 | 819 | 11 | 2 | 11 | 4 |
| 6 | 8 | 3 | 7 | 2 | 7 | 9 | 914 | 0 | 7 | 1 |
| 7 | 1 | 3 | 18 | 12 | 2 | 2 | 0 | 960 | 2 | 28 |
| 8 | 4 | 3 | 6 | 16 | 7 | 13 | 6 | 8 | 904 | 7 |
| 9 | 5 | 4 | 1 | 13 | 29 | 7 | 0 | 13 | 4 | 933 |

**表2** 固定随机状态后神经网络预测结果的准确率和召回率

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 模型得分 | 微平均准确率 | 宏平均准确率 | 加权平均准确率 | 微平均召回率 | 宏平均召回率 | 加权平均召回率 |
| 0.9451 | 0.9451 | 0.9445 | 0.9452 | 0.9451 | 0.9445 | 0.9451 |

**3.2单层神经网络的节点数与训练效果的关系**

random\_state设定为1551，重复2.2.1-2.2.4的流程，每次仅改变单层神经网络的节点数，结果如表3所示。

**表3** 单层神经网络的节点数与训练效果关系表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 单层神经网络的节点数 | 50 | 100 | 150 | 200 |
| 模型得分 | 0.9451 | 0.9583 | 0.9648 | 0.9691 |
| 微平均准确率 | 0.9451 | 0.9583 | 0.9648 | 0.9691 |
| 宏平均准确率 | 0.9445 | 0.9578 | 0.9646 | 0.9689 |
| 加权平均准确率 | 0.9452 | 0.9583 | 0.9648 | 0.9692 |
| 微平均召回率 | 0.9451 | 0.9583 | 0.9648 | 0.9691 |
| 宏平均召回率 | 0.9445 | 0.9577 | 0.9643 | 0.9688 |
| 加权平均召回率 | 0.9451 | 0.9583 | 0.9648 | 0.9691 |

**3.3单层神经网络的最大迭代次数与训练效果的关系**

random\_state设定为1551，重复2.2.1-2.2.4的流程，每次仅改变单层神经网络的最大迭代次数，其他参数不变，结果如表4所示。

**表4** 单层神经网络的最大迭代次数与训练效果关系表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 单层神经网络的最大迭代次数 | 500 | 1000 | 1500 | 2000 |
| 模型得分 | 0.9451 | 0.9451 | 0.9451 | 0.9451 |
| 微平均准确率 | 0.9451 | 0.9451 | 0.9451 | 0.9451 |
| 宏平均准确率 | 0.9445 | 0.9445 | 0.9445 | 0.9445 |
| 加权平均准确率 | 0.9452 | 0.9452 | 0.9452 | 0.9452 |
| 微平均召回率 | 0.9451 | 0.9451 | 0.9451 | 0.9451 |
| 宏平均召回率 | 0.9445 | 0.9445 | 0.9445 | 0.9445 |
| 加权平均召回率 | 0.9451 | 0.9451 | 0.9451 | 0.9451 |

**3.4单层神经网络的学习速率与训练效果的关系**

random\_state设定为1551，重复2.2.1-2.2.4的流程，每次仅改变单层神经网络的学习速率，其他参数不变，结果如表4所示。

**表4** 单层神经网络的最大迭代次数与训练效果关系表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 单层神经网络的学习速率 | 0.0001 | 0.001 | 0.01 | 0.1 |
| 模型得分 | 0.9451 | 0.9488 | 0.9027 | 0.6744 |
| 微平均准确率 | 0.9451 | 0.9488 | 0.9027 | 0.6744 |
| 宏平均准确率 | 0.9445 | 0.9485 | 0.9025 | 0.6721 |
| 加权平均准确率 | 0.9452 | 0.9489 | 0.9032 | 0.6744 |
| 微平均召回率 | 0.9451 | 0.9488 | 0.9027 | 0.6744 |
| 宏平均召回率 | 0.9445 | 0.9481 | 0.9013 | 0.6676 |
| 加权平均召回率 | 0.9451 | 0.9488 | 0.9027 | 0.6744 |

**3.5多层神经网络的层数及节点数与训练效果的关系**

random\_state设定为1551，重复2.2.1-2.2.4的流程，每次仅改变多层神经网络的层数及节点数，其他参数不变，结果如表5所示。

**表5** 多层神经网络的层数及节点数与训练效果关系表（学习速率）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 多层神经网络的层数及节点数 | 1, (200) | 2, (200, 200) | 2, (200, 100) | 3, (200, 200, 200) | 3, (200, 100, 50) | 4, (200, 200, 200, 200) | 4, (200, 100, 50, 25) |
| 模型得分 | 0.9691 | 0.9691 | 0.9687 | 0.9646 | 0.9648 | 0.9620 | 0.9608 |
| 微平均准确率 | 0.9691 | 0.9691 | 0.9687 | 0.9646 | 0.9648 | 0.9620 | 0.9608 |
| 宏平均准确率 | 0.9689 | 0.9689 | 0.9685 | 0.9646 | 0.9646 | 0.9616 | 0.9604 |
| 加权平均准确率 | 0.9692 | 0.9692 | 0.9687 | 0.9647 | 0.9648 | 0.9620 | 0.9608 |
| 微平均召回率 | 0.9691 | 0.9691 | 0.9687 | 0.9646 | 0.9648 | 0.9620 | 0.9608 |
| 宏平均召回率 | 0.9688 | 0.9688 | 0.9683 | 0.9641 | 0.9644 | 0.9615 | 0.9604 |
| 加权平均召回率 | 0.9691 | 0.9691 | 0.9687 | 0.9646 | 0.9648 | 0.9620 | 0.9608 |

**表6** 多层神经网络的层数及节点数与训练效果关系表（学习速率）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 多层神经网络的层数及节点数 | 1, (200) | 2, (200, 200) | 2, (200, 100) | 3, (200, 200, 200) | 3, (200, 100, 50) | 4, (200, 200, 200, 200) | 4, (200, 100, 50, 25) |
| 模型得分 | 0.9634 | 0.9731 | 0.9698 | 0.9699 | 0.9676 | 0.9687 | 0.9664 |
| 微平均准确率 | 0.9634 | 0.9731 | 0.9698 | 0.9699 | 0.9676 | 0.9687 | 0.9664 |
| 宏平均准确率 | 0.9633 | 0.9728 | 0.9697 | 0.9701 | 0.9675 | 0.9685 | 0.9662 |
| 加权平均准确率 | 0.9635 | 0.9731 | 0.9699 | 0.9701 | 0.9676 | 0.9691 | 0.9665 |
| 微平均召回率 | 0.9634 | 0.9731 | 0.9698 | 0.9699 | 0.9676 | 0.9687 | 0.9664 |
| 宏平均召回率 | 0.9630 | 0.9729 | 0.9695 | 0.9696 | 0.9673 | 0.9684 | 0.9660 |
| 加权平均召回率 | 0.9634 | 0.9731 | 0.9698 | 0.9699 | 0.9676 | 0.9687 | 0.9664 |

**3.6实验结论**

（1）在手写数字识别中，仅用一层隐藏层的多层感知神经网络模型，其准确率和召回率达90%以上，基本满足手写数字识别的需求。

（2）在其他条件不变的情况下，增大单层的节点数，能改善多层感知神经网络模型的表现，但与此同时，训练时相应的计算量和计算时间也会增加。

（3）在其他条件不变的情况下，改变最大迭代次数，对多层感知神经网络模型的表现没有明显改善，这可能是因为在迭代次数小于500时，模型即已经收敛，继续增加最大迭代次数并不会影响模型的训练结果。

（4）在其他条件不变的情况下，改变学习速率，对多层感知神经网络模型的表现具有两面性：当学习速率从0.0001增大到0.001时，模型的准确率和召回率均有少量提升，这可能是因为适当的学习速率能使模型快速收敛到最优解；当学习速率从0.001继续增大时，模型的准确率和召回率开始出现明显下滑，及至0.1时，学习速率迅速下滑至不到70%，说明模型出现了劣化，这可能是因为过大的学习速率可能导致模型“囫囵吞枣”，出现过拟合现象，或是模型过早收敛，导致各参数优化不足。

（5）在其他条件不变的情况下，若学习速率为0.0001，则采用多个隐藏层时，对多层感知神经网络模型的表现没有明显改善，反而导致了准确率和召回率的少量下降；而若学习速率为0.001，则采用多个隐藏层时，多层感知神经网络模型的表现先是出现了一定改善，随层数增加，改善效果逐渐下降。这可能是因为迭代次数过少，导致采用多个隐藏层时，多层感知神经网络模型未能收敛到最优解，如果能适当调节最大迭代次数和模型学习速率，则加入多个隐藏层可以提升模型质量。