**华东理工大学 2019 至 2020 学年（第 1 学期）**

**研究生《模式识别原理与应用》课程论文 2019 年 11 月**

开课学院： 信息学院 任课教师： 赵海涛

考生姓名： 廖磊 学号：\_Y30190696\_ 成绩：

|  |
| --- |
| 论文题目： 数字识别器 |
| 论文要求：   1. 方法部分要针对问题对使用的方法进行较为详细的描述 2. 实验部分要先给出实验数据的描述，实验结果有相关的图表和分析 3. 给出自己的结论，对使用方法的优缺点进行总结 |
| 教师评语：  教师签字：  年 月 日 |

**数字识别器**

廖磊

1. **数字识别器比赛简介**

MNIST（“国家标准技术研究院”）是计算机视觉的事实上的“ hello world”数据集。自从1999年发布以来，这个经典的手写图像数据集就成为基准分类算法的基础。随着新的机器学习技术的出现，MNIST仍然是研究人员和学习者的可靠资源。

在这场比赛中，您的目标是从数万个手写图像的数据集中正确识别数字。我们策划了一系列教程风格的内核，涵盖了从回归到神经网络的所有内容。我们鼓励您尝试不同的算法，以第一手了解什么有效，以及如何比较技术。

**二、整体解决方案**

本实验主要是通过创建一个简单的神经网络用来对图片进行识别。

**2.1 数据导入**

首先导入程序必要的包，

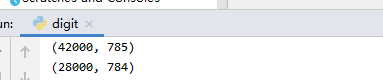
|  |
| --- |
| **import** pandas **as** pd **import** numpy **as** np **import** tensorflow **as** tf |

然后导入数据，数据包括训练数据train.csv，和测试数据test.csv。

同时查看训练数据和测试数据的形状。

|  |
| --- |
| train = pd.read\_csv(**'train.csv'**) test = pd.read\_csv(**'test.csv'**) print(train.shape) print(test.shape) |

运行结果如图所示



为了方便处理，把输入和结果分开。并分别对这些数据进行处理。

|  |
| --- |
| images\_train = train.iloc[:,1:].values labels\_train = train.iloc[:,0].values images\_test = test.iloc[:,:].values  images\_train = images\_train.astype(np.float) images\_train = np.multiply(images\_train,1.0/255) images\_test = images\_test.astype(np.float) images\_test = np.multiply(images\_test,1.0/255) images\_size = images\_train.shape[1] images\_width = images\_height = np.ceil(np.sqrt(images\_size)).astype(np.uint8) labels\_count = np.unique(labels\_train).shape[0] |

**2.2 one-hot编码**

one hot编码是将类别变量转换为机器学习算法易于利用的一种形式的过程。神经网络的输出中，一个属性如果有N个可取值，它就可以扩充为N个属性，每个样本的这N个属性中，只能有一个为1，表示该样本的该属性属于这个类别，其余扩展属性都为0。

|  |
| --- |
| **def** dense\_to\_ont\_hot(labels\_dense,num\_classes):  num\_labels = labels\_dense.shape[0]  index\_offset = np.arange(num\_labels) \* num\_classes  labels\_one\_hot = np.zeros((num\_labels,num\_classes))  labels\_one\_hot.flat[index\_offset + labels\_dense.ravel()] = 1  **return** labels\_one\_hot  labels = dense\_to\_ont\_hot(labels\_train,labels\_count) labels = labels.astype(np.uint8) |

**2.3 神经网络**

神经网络是一种模拟人脑的神经网络以期能够实现类人工智能的机器学习技术。一个基本的神经网络的结构，有输入层，隐藏层，和输出层。这里是使用Tensorflow来构建神经网络。**Tensorflow**是谷歌开发的深度学习系统，用它可以很快速地入门神经网络。

创建一个简单的神经网络用来对图片进行识别，该神经网络使用的是梯度下降法来优化优化参数的。

|  |
| --- |
| x = tf.placeholder(**'float'**,shape=[**None**,images\_size]) y = tf.placeholder(**'float'**,shape=[**None**,labels\_count])  weights = tf.Variable(tf.zeros([784,10])) biases = tf.Variable(tf.zeros([10])) result = tf.matmul(x, weights) + biases predictions = tf.nn.softmax(result)  loss = tf.reduce\_mean(tf.nn.softmax\_cross\_entropy\_with\_logits(labels=y,logits=predictions))  train\_step = tf.train.GradientDescentOptimizer(0.1).minimize(loss) |

**2.4 计算预测值**

进行训练和预测，这里训练了100次。

|  |
| --- |
| **with** tf.Session() **as** sess:  *#初始化* sess.run(init)  *#循环100次* **for** epoch **in** range(100):  **for** batch **in** range(n\_batch-1):  batch\_x = images\_train[batch\*batch\_size:(batch+1)\*batch\_size]  batch\_y = labels[batch\*batch\_size:(batch+1)\*batch\_size]  *#进行训练* sess.run(train\_step,feed\_dict={x:batch\_x,y:batch\_y})  batch\_x = images\_train[n\_batch\*batch\_size:]  batch\_y = labels[n\_batch\*batch\_size:]  sess.run(train\_step,feed\_dict={x:batch\_x,y:batch\_y})  *#计算预测* myPrediction = sess.run(predictions,feed\_dict={x:images\_test}) |

最后将生成的结果产生result.csv文件。

|  |
| --- |
| label\_test = np.argmax(myPrediction,axis=1) pd.DataFrame(label\_test).to\_csv(**'result.csv'**) |

**三 实验总结**

最终的准确率是0.918，由于这是一个单层的神经网络，丢失了手写图片的二维结构，所以最终的效果并不好。而且初次使用神经网络来识别图像问题，很多细节都没有考虑到。通过本次实验，我对使用神经网络来解决实际问题的方法有了初步的了解，同时对python语言更加熟悉。

**附：文件说明**

本次附件一共包含有：

1. 大作业报告；
2. 最终的python程序源码：digit.py；
3. 训练数据：train.csv；
4. 测试数据;test.csv；（由于数据过大，所以train和test文件在一个压缩包里面）
5. 最终预测数据：result.csv