**图像处理深度学习专题——实验1：mnist手写数字识别**

**实验报告**

**姓名 王昊 学号 201821061107 专业 计算机应用技术**

1. 运行demo1.py-以下是我的运行结果
   1. 网络结构：

Input：1@28\*28

layer1：10@3\*3 featuremap1：10@28\*28

layer2：max-pooling@2\*2 featuremap2：10@14\*14

layer3：全连接层激活函数RELU@20 featuremap3：20

layer4：softmax featuremap4：10

* 1. 网络训练

训练集样本数量：60000 测试集样本数量：10000

损失函数：crossentropy(交叉熵)

优化算法：Adadelta

迭代次数：10

demo2：完成demo2.py

* 1. 找出10幅判错的图像（自己完成demo2.py）



from \_\_future\_\_ import print\_function

import keras

from keras.datasets import mnist

from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten

from keras.layers import Conv2D, MaxPooling2D

from keras import backend as K

import numpy as np

import matplotlib.pyplot as plt

*#batch\_size, num\_classes and epochs*

batch\_size = 128

num\_classes = 10

epochs = 10

*# input image dimensions*

img\_rows, img\_cols = 28, 28

*# the data, shuffled and split between train and test sets*

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

x\_train = x\_train.reshape(x\_train.shape[0], img\_rows, img\_cols, 1)

x\_test = x\_test.reshape(x\_test.shape[0], img\_rows, img\_cols, 1)

input\_shape = (img\_rows, img\_cols, 1)

x\_train = x\_train.astype('float32')

x\_test = x\_test.astype('float32')

x\_train /= 255

x\_test /= 255

print('x\_train shape:', x\_train.shape)

print(x\_train.shape[0], 'train samples')

print(x\_test.shape[0], 'test samples')

*# convert class vectors to binary class matrices*

y\_train = keras.utils.np\_utils.to\_categorical(y\_train, num\_classes)

y\_test = keras.utils.np\_utils.to\_categorical(y\_test, num\_classes)

*#construct a convolutional neuron network*

*#if you change the architecture, you should change the following code*

*#####################################################################*

model = Sequential()

model.add(Conv2D(10,(3,3),activation='relu',input\_shape=input\_shape))

model.add(MaxPooling2D(pool\_size=(2, 2)))

model.add(Flatten())

model.add(Dense(20, activation='relu'))

model.add(Dense(num\_classes, activation='softmax'))

*####################################################################*

*#model compile:loss function, optimiazer and metrics*

model.compile(loss=keras.metrics.categorical\_crossentropy,

              optimizer=keras.optimizers.Adadelta(),

              metrics=['accuracy'])

*#visualize the architecture of CNN*

model.summary()

*#load the well-trained parameters, including weights and bias*

model.load\_weights('mnist\_demo1.h5')

*#obtain the model prediction in testing set*

model\_prediction = model.predict(x\_test)

*#try to find the differencees between  model\_prediction and y\_test to find the miss judgement*

*###################################################################*

plt.figure(figsize=(20, 4))  *#画一张画布*

b = [2,0,1,8,2,1,0,6,1,1,0,7]  *#学号*

n = **len**(b)  *#学号长度*

for j in **range**(**len**(b)):  *#对学号进行循环*

    for i in **range**(10000):  *#对验证集的样本进行循环*

        pos\_model = np.argmax(model\_prediction[i,:])*#真实标签,最大的神经元索引位置*

        pos\_y = np.argmax(y\_test[i,:])*#模型输出*

        if pos\_model != pos\_y:  *#如果判错*

                if pos\_y == b[j]: *#如果判错且正好等于学号*

                    ax = plt.subplot(1, n, j + 1)

                    plt.imshow(x\_test[i].reshape(28, 28))

                    plt.gray()

                    break

plt.show()

*#################################################################*

1. 完成demo3.py，请自己设计网络结构（至少10层且准确率在99%以上），提高识别的准确率

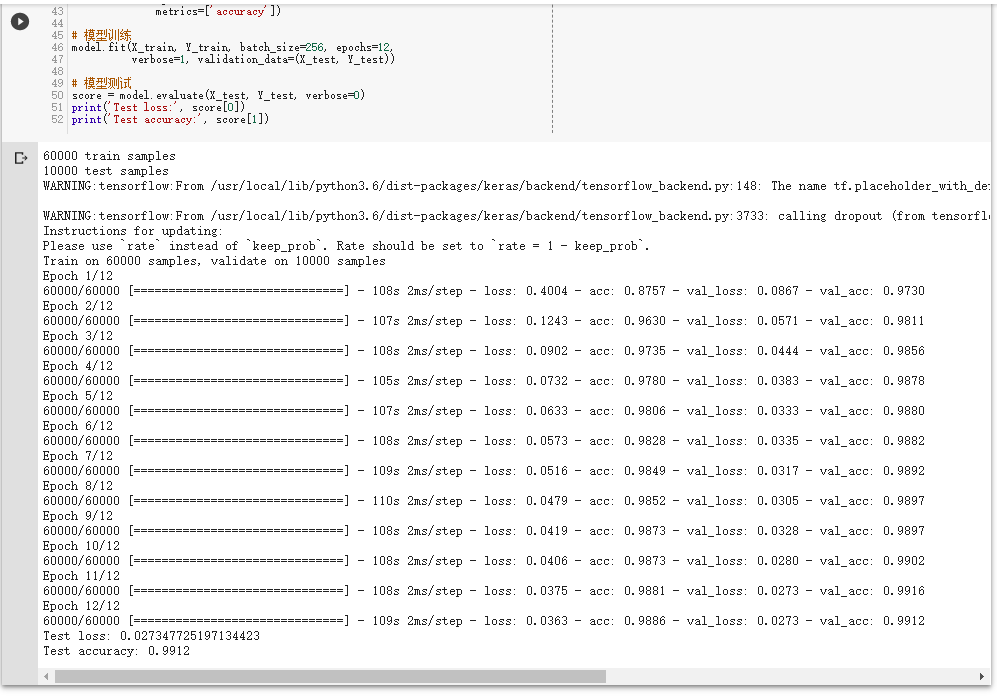
提示：1.增加网络深度

2.增加卷积核的个数

3.增加全连接层神经元的个数

4.增加dropout层（可以自己查阅相关资料）

* 1. 网络结构：



# 建立CNN模型

model = Sequential()

model.add(Conv2D(32, 3, input\_shape=[28,28,1]))

model.add(Activation('relu'))

model.add(Conv2D(32, 3))

model.add(Activation('relu'))

model.add(MaxPool2D([2,2]))

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(128))

model.add(Activation('relu'))

model.add(Dropout(0.5))

model.add(Dense(10))

model.add(Activation('softmax'))

Input：1@28\*28

layer1：32@3\*3 featuremap1：8@26\*26

layer2：32@3\*3 featuremap2：12@24\*24

layer3：MaxPool2D([2,2]) featuremap3：14@22\*22

layer4：Dropout(0.25) featuremap4：18@20\*20

layer5：Flatten() featuremap5：18@10\*10

layer6：Dense(128) featuremap6：10@8\*8

layer7：Dropout(0.25) featuremap7：12@6\*6

layer8：Dense(10) featuremap8：14@4\*4

* 1. 网络训练

训练集样本数量：60000 测试集样本数量：10000

损失函数：crossentropy

优化算法：Adadelta

迭代次数：10

dropout概率：0.25, 0.5

* 1. 找出10幅判错的图像



问题：

1. demo1中网络各层的各有多少个参数（卷积核+偏置个数）？

layer1：100

layer2：0

layer3：33820 (20+13\*13\*10\*20)

layer4：210

1. 对于全连接层，前一层有a个节点，后一层有b个节点，那么a与b之间有多少个参数？

a\*b+b

1. 卷积操作和相关操作的区别是什么？

卷积:图像绕自己的核心元素顺时针旋转180°,然后对应项相乘再相加;

相关:对应项相乘再相加

1. 卷积神经网络与全连接神经网络的区别是什么？

卷积神经网络:相邻2层之间只有部分结点连接

全连接神经网络:相邻2层之间的结点都有连接