模式识别作业报告——辛普森一家角色识别

组员：王爱民

经过一个学期的学习，我对模式识别有了一定的认知，对其中主流的方法原理有了一定的了解，并能证明其中一些基本算法，这使得我对模式识别产生了极大的兴趣。，我通过对kaggle上的一个问题进行解决以加强自己对模式识别的理解。

我对辛普森一家动画中的角色进行了识别，方法使用的是卷积神经网络，通过修改数据预处理的方式，加深神经网络层数，修改损失函数参数等一系列方法，以提高测试集和训练集的分类精度，在其中也遇到了很多问题，得到了同学和师兄们的指导帮助下完成了人物，下面是我的解决过程。

一、数据集简介

我这次看到了kaggle上的一个比赛，任务是使用快速神经网络识别辛普森一家的角色，我虽然没有接触过人脸识别，但是对这些很感兴趣，因此我选择了学习使用卷积神经网络来实现人脸识别。

数据集是一位辛普森一家爱好者提供的图片数据，数据是直接从电视节目剧集中获取并标记的，一共有大约二十个人物，每个人物的图片从400到2000不等，我从中选取了7个人物，每个人物有1000张左右的图片。



**图1数据集**

二、整体解决方案

2.1 数据处理

这次采用的数据是图片数据，彩色图片使用python自带的函数读取后，每一张图片是三个矩阵，每个矩阵的维度对应图片的大小，而三个矩阵分别是图片的rgb（red，green，blue）值，也就是三原色值，通过程序将rgb值读出后，每个数据都是在0~255之间的一个数值。

由于有些图片之间维度不同，有些维度是288×416，有些维度是576×416，所以为了统一维度，把所有的图片数据都修改为256×256大小，并将图片存储为numpy数组格式，用以特征提取。然后取前100张图片的数据作为测试数据，其余图片作为训练数据，测试训练比约为1比9，为了节省数据内存，先将样本转换为float32类型，并除以255进行归一化。

该部分 Python 程序如下所示：

def read\_img(location):

x\_train = []

y\_train = []

x\_test = []

y\_test = []

label\_name = []

dirs = os.listdir(location)

label = 0

count = 0

for i in dirs: #loop all directory

print(i)

n = 0

label\_name.append(i) #save folder name in var label\_name

x\_s = 256

y\_s = 256

for pic in glob.glob(location+'\\'+i+'\\*.jpg'):

im = Image.open(pic) #open data

im = im.resize((x\_s, y\_s), Image.ANTIALIAS)

im = np.array(im) #store im as numpy array

if(im.shape[0]==256 and im.shape[1]==256):

r = im[:,:,0]

g = im[:,:,1]

b = im[:,:,2]

if(n<100):

x\_test.append([r,g,b]) #save in x\_test

y\_test.append([label]) #save in y\_test

else : #remaining data set as training data

x\_train.append([r,g,b]) #save in x\_train

y\_train.append([label]) #save in y\_train

n = n + 1

count = count + 1

label = label + 1 #increment label

print(label\_name)

print(dirs)

return np.array(x\_train),np.array(y\_train),np.array(x\_test),np.array(y\_test)

x\_train = x\_train.reshape(x\_train.shape[0], img\_rows, img\_cols, 3)

x\_test = x\_test.reshape(x\_test.shape[0], img\_rows, img\_cols, 3)

input\_shape = (img\_rows, img\_cols, 3)

x\_train = x\_train.astype('float32')

x\_test = x\_test.astype('float32')

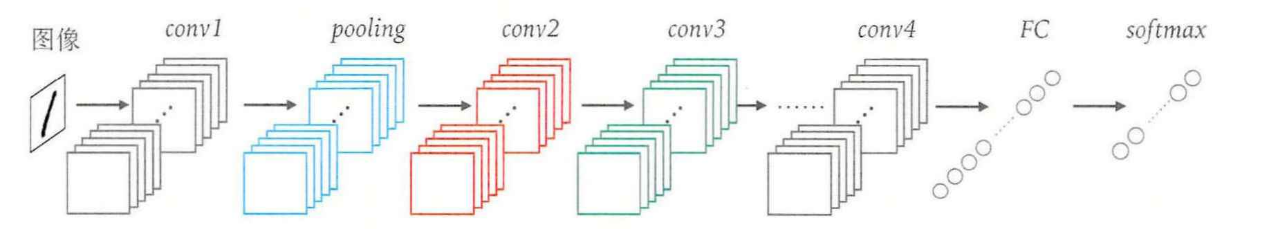
x\_train /= 255

x\_test /= 255

2.2 算法原理及程序实现

卷积神经网络（convolutional neural network，CNN），是一种专门用来处理具有类似网格结构的神经网络。例如时间序列数据（可以认为是在时间轴上有规律地采样形成的一维网络）和图像数据（可以看作二维的像素网格）。卷积网络在很多领域上都有着优异的表现。卷积网络是指那些至少在网络的一层中使用卷积运算来替代一般的矩阵乘法预算的神经网络。

随着机器学习理论的完善以及硬件GPU运算速度的加快，卷积网络神经网络在近几年有了结构上的改进和较大的发展，但是其主要的功能单元并没有发生变化。卷积神经网络在结构上主要由输入层、卷积层、池化层（pooling）、全连接层和输出层组成，如图2所示。其中全连接层（Fully-connected layer，FC）可以视为特殊的卷积层。

**图2卷积神经网络示意图**

网络的结构直接影响卷积神经网络的泛化性能。一般来说，层次越深的网络，卷积层和池化层就越多，网络容量就越大。此时在大数据集上训练，得到的结果泛化能力就很强。但是如果使用小数据集训练，不仅泛化能力不会增强，还会导致严重的过拟合。因此，网络结构并不是越深越好，而是应该视数据集的大小而定，例如本文中所使用的数据集较小，因此网络结构就较浅。

用CNN卷积神经网络识别图片，一般需要的步骤有：卷积层初步提取特征、池化层提取主要特征、全连接层将各部分特征汇总、产生分类器并进行预测识别。

首先，我们构建了一个前馈网络，包括4个带有ReLU激活函数的卷积层和一个全连接的隐藏层（随着数据量的增大，可能会进一步加深网络）。我还在模型中加入了Dropout层来防止网络过拟合。在输出层中，使用softmax函数来输出各类的所属概率。

损失函数为分类交叉熵（Categorical Cross Entropy）。优化器optimizer使用了随机梯度下降中的RMS Prop方法，通过该权重临近窗口的梯度平均值来确定该点的学习率。

该部分 Python 程序如下所示：

model = Sequential()

model.add(Conv2D(64, kernel\_size=(3, 3),activation='relu',input\_shape=input\_shape))

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

model.add(Conv2D(32, (3, 3), activation='relu'))

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

model.add(Dropout(0.25))

model.add(Conv2D(32, (3, 3), activation='relu'))

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

model.add(Dropout(0.25))

model.add(Conv2D(32, (3, 3), activation='relu'))

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

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dropout(0.5))

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

model.compile(loss='categorical\_crossentropy',

optimizer=SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True),

metrics=['accuracy'])

2.3 训练模型

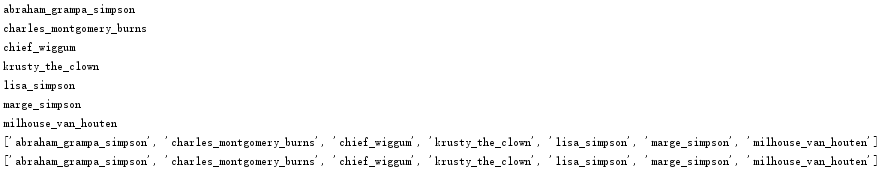
训一次模型并保存：模型一个批次处理64个样本，迭代1次，用测试集数据做验证。循环进行模型训练，每一次循环迭代一次训练，保存并读取模型，循环二十次。

该部分 Python 程序如下所示：

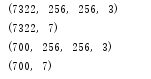
model.fit(x\_train, y\_train, batch\_size=32, epochs=1, verbose=1, validation\_data=(x\_test, y\_test))  
model.save('Simpson.h5')  
for i in range(0,20):  
 print('The '+str(i)+' th Iteration')  
 model=load\_model('Simpson.h5')  
 model.fit(x\_train, y\_train, batch\_size=32, epochs=1, verbose=1, validation\_data=(x\_test, y\_test))  
 model.save('Simpson.h5')  
 K.clear\_session()

2.4调试及预测结果

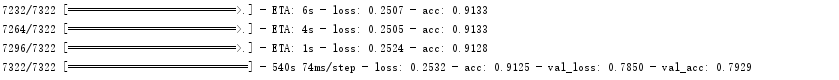
运行程序后，输出训练标签向量和测试标签向量，如图3所示；输出训练训练特征矩阵、训练标签向量、测试特征矩阵、测试标签向量的维度如图4所示；运行结果：该模型在测试集上最终达到了91.96%的准确率，在测试集上达到了79.29％的准确度，还有调试中的训练结果，如图5所示。



**图3训练标签向量和测试标签向量**



**图4训练训练特征矩阵、训练标签向量、测试特征矩阵、测试标签向量的维度**









**图5训练结果**

三、小组分工

程序设计及编写：王爱民

程序调试：王爱民

实验报告：王爱民

四、作业总结

这是第一次接触CNN，对卷积神经网络的理解不够深刻，而要处理的数据量又很大，使用的是cpu跑卷积神经网络程序，每跑一次就要数个小时，对于调制参数也很不方便，而时间和经验都不充裕，所以处理的结果也只能是差强人意，对于卷积神经网络中一些精妙的东西都难以觉察，只能粗浅的进行模仿，难以学到精髓所在。

对于这次遇到的很多问题也是不求甚解，只能猜测其中的原因。例如的模型在训练集上的分类准确率达到了90％多，而在测试集上分类准确率只有不到80％，我也改变过测试集的选取，但一直都难以提升测试集的准确率；当我减少训练模型的训练步长时，训练集的准确率一直在提升，而测试集的准确率则是一开始增加后来减少，我觉得这是因为训练步长过短，导致模型过拟合，使得模型在测试集上的性能变坏；我一开始设置了三层网络，在训练集上效果比较理想但是在测试集上不好，于是我加深了神经网络层数，测试集上的效果依然不到80％，我觉得这可能是数据集中偶尔会出现别的人物导致的。总之这次试验能做的事情很多，但是我只做了一部分，没有做的很好，还有很多的方面需要改进。这一次还得到了同学师兄的帮助，感谢他们！