**基于CNN的英语默写批改**

曹汇杰 王菲

华东师范大学 上海

**摘要**：本文对中小学生的默写本进行校正，降噪，轮廓分割提取等一系列图像预处理后，喂入CNN网络进行识别，以期得到相应的字符串与老师输入的正确答案进行比较以实现英语默写的自动批改，来达到为老师减负的目的。

关键词：Opencv CNN 识别 emnist VGG

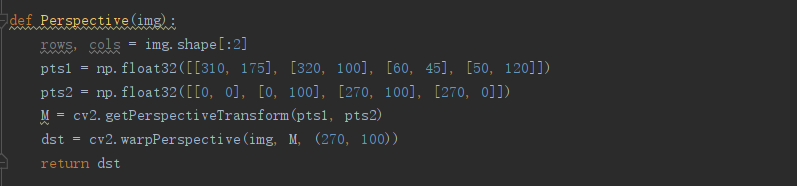
一、课题缘起

现今默写已经成为小学、初中、高中英语老师检查学生英语词汇和语法掌握程度的最快捷的方法，然而英语老师也常常为每天批改上百本英语默写而烦恼。由此我们希望开发出一款能自动批改手写英语默写的应用来减轻老师的负担。

二、图像预处理

2.1 图像校正

很多老师在拍摄照片时，未必会正好拍成水平位置，这样会对我们后期的处理带来麻烦，所以我们首先做矫正，将图像转化至水平位置。我们使用getPerspectiveTransform来将旧的点映射至新的点，具体代码如下：



2.2 降噪

手写的字体难免会有很多噪声，我们这里使用高斯模糊和形态学运算来降噪。

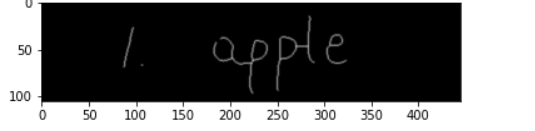
2.2.1 缩小原图分辨率

缩小原图的分辨率来减少计算量，具体代码显示如下：

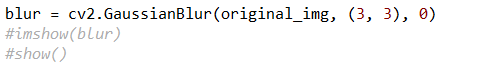


2.2.2 高斯模糊

高斯模糊滤波器是在像素与周围像素的亮度差值小于一个特定值时，平滑该像素的亮度。一开始为了更好地实现降噪，选择了更大的kernel size，但会使字母之间相连，结果如下：



因为目前还不具备识别连体的能力，所以选择了更小的kernel size。具体实现如下：

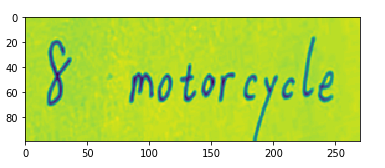


2.2.3 将图像转换为灰度图

之后将使用otsu阈值法，所以将图像转化为灰度图，具体实现如下：



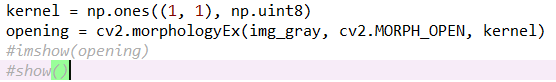
结果如下：



2.2.4 形态学运算

2.2.4.1 开运算

开运算是对图像先腐蚀再膨胀，分离物体，消除小区域，使图片中的字符边界更圆滑，没有皱褶。具体实现如下：



2.2.4.2 膨胀

腐蚀的效果是把图片"变瘦"，其原理是在原图的小区域内取局部最小值，以达到“瘦身”的效果。具体实现如下：



2.3 轮廓分割提取

将图进行降噪之后，我们要将一个单词分成一个个字母，来为之后的识别做好准备。

2.3.1 转化为二值图

我们这里采用otsu阈值法来将原图转化为二值图，因为之后轮廓识别的输入应为二值图。Otsu算法假设这副图片由前景色和背景色组成，通过统计学方法（最大类间方差）选取一个阈值，将前景和背景尽可能分开，具体代码实现如下：



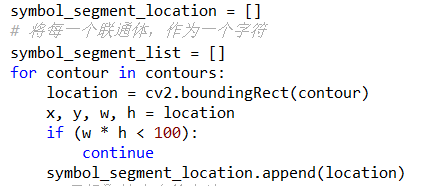
2.3.2 轮廓识别

我们这里采用findContours来进行轮廓识别。寻找每一个字符的轮廓，使用cv2.RETR\_EXTERNAL模式，表示只需要每一个字符最外面的轮廓，至于像p和b这种有内部空档的字母在之后用位运算来还原，具体实现如下：



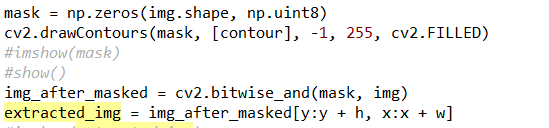
2.3.3 矩形分割

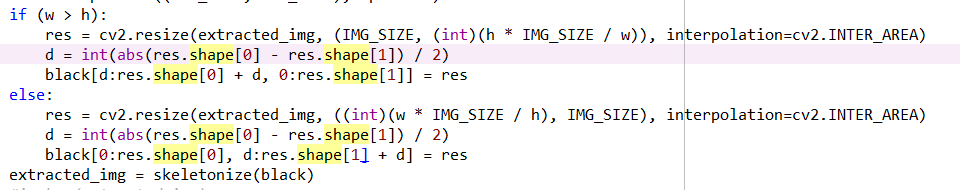
我们这里采用boundingRect来实现矩阵分割，该函数用一个最小的矩形，把找到的形状包起来，并返回x，y，w，h，其中x，y是矩阵左上点的坐标，w，h是矩阵的宽和高，代码实现如下：



2.3.4 统一大小并还原字母

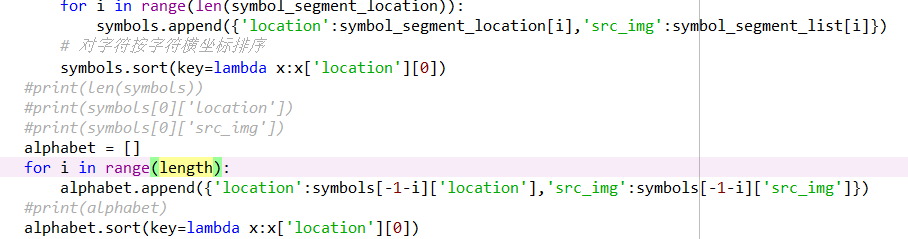
我们将轮廓经过位运算，将原来含有内轮廓的字母进行还原，并将还原后的字母大小统一，并用skeletonize描绘轮廓使后面可以更好地投入训练，具体代码实现如下：





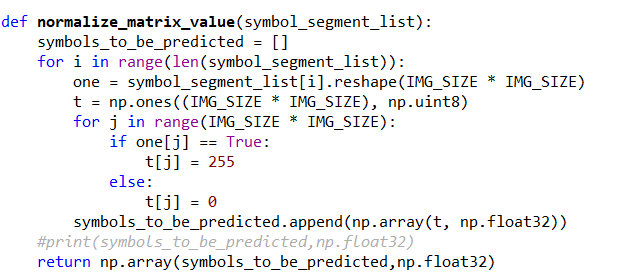
2.3.5 删除数字

默写不可避免的要写序号，然而这样的数字在识别中毫无意义，甚至带来负面作用，所以我们通过把图像转至水平位置之后，根据用户输入的答案长度，提取排序后轮廓的后几位进行识别，起到删除数字的作用，具体实现如下：



2.3.6 将图像转化回【0，255】

因为之后的训练集是【0，255】的，所以将图重新转化为【0，255】，具体实现如下：



三、基于CNN的识别

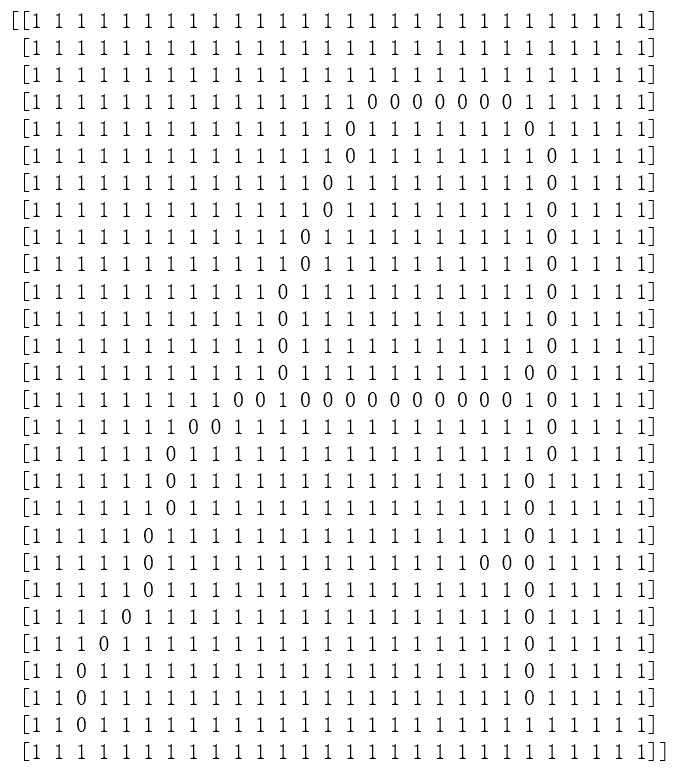
3.1 数据集介绍

本次实验中我们使用的是emnist数据集，emnist数据集是对mnist数据集的延申，我们使用其中的letter类，它是大小写字母组合在一起的，每一类训练集 4800 张，测试集 800 张，共145600张图片。其中图片为28\*28，每一个像素点为0-255。图片如下：

3.2 数据预处理

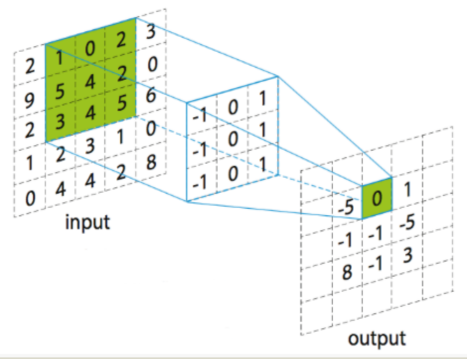
由于要将数据集贴近输入图像的格式，在导入数据的时候，对原有的数据集进行一定的变换。首先将图片转换为灰度图，然后采用阈值法将图片转换为二值图，最后用OpenCV中的轮廓识别api识别出图片最外层轮廓。得到形式如下的矩阵，然后再将1转化为255：



3.3 模型建立

3.3.1 基本思想

采用了VGGNet的基本思想。首先，连续的两个使用３ｘ３的卷积核的卷积层，可以扩大视野域，并且减少卷基层的参数；两个使用３ｘ３卷积核的卷积层可以等价于一个使用5ｘ5卷积核的卷积层，但是参数减少28%。此外，因为经过两次卷积核，进行了两次非线性变换，提高了模型的准确率。



3.3.2 计算图结构

第一层为输入层，将一维向量转换成为4D的tensor，格式为[-1,28,28,1]，是一通道，28\*28的图像；

第二层与第三层为卷积层，卷积核大小为(3, 3)，因为图像大小不能整除，所以要开启padding。通过两层卷积核为3\*3的卷积层相当于通过一层卷积核为5\*5的卷积层，而且参数少，速度快，准确率高；

第四层为最大池化层，核的大小为(2, 2)，步长为(2, 2)。相当于将图像中的信息凝练1/2，加快训练速度；

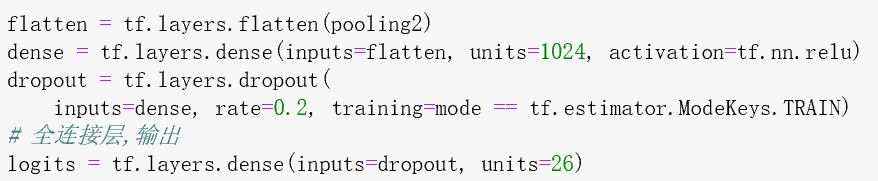
第五、六、七层与第二、三、四层结构相似，增加了非线性变换，增加准确率；

第八层为展开层，将pooling2得到的矩阵展开成一维向量；

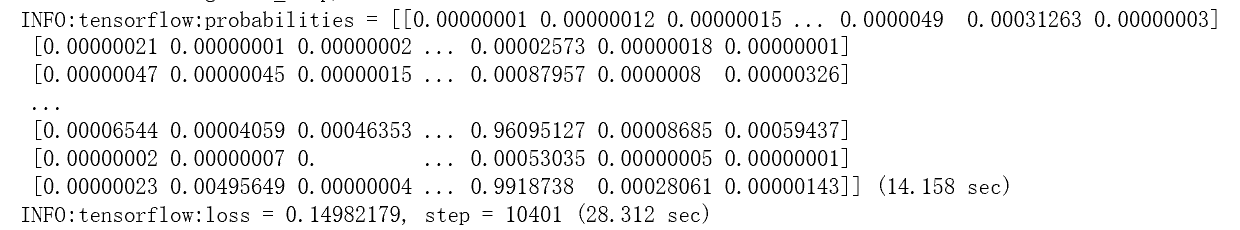
第九层为预测层，将一维向量映射到1024个结果上，计算每个结果的可能性；

第十层为dropout层，设置超参数损失率为0.2 (经前人验证得知，0.2的损失率得到的模型综合泛化能力与预测能力之后，表现最好)；

第十一层为预测层，将数据映射到26个结果上，分别表示26个英语字母。



3.3.3 训练过程

 训练次数为11000次，每100次输出一次模型信息，输出信息如下图：

图中显示的是训练过程中100张照片所对应的26个英文字母的概率，总体的损失率为loss，总次数为10401。

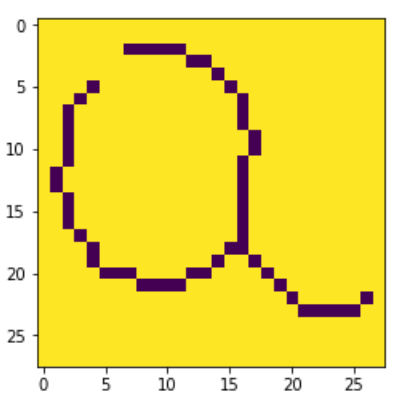
3.3.4 日志记录

因为在训练完模型之后，需要另外载入测试集进行预测，所以需要记录下模型的各个参数，以便之后再次运行Session进行预测；而且，这样还可以不需要一次训练完模型，可以分次，分批对模型进行训练，使得训练过程更加灵活多变。

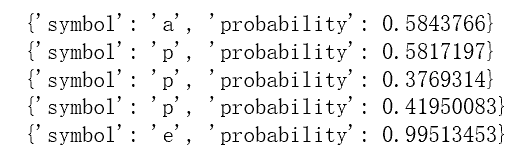


四、结果展示

用的是apple的测试集，输入格式：



测试结果：



可见，对于字母e模型预测的准确率很高，字母a和p的准确率一般，而字母l并未被识别出来。

五、未来展望

现在的作品还存在两个缺陷；

（1）对连笔字的识别效果不是最好，甚至无法识别。

针对这个问题，我们希望用隐马尔可夫模型（HMM）或者制作属于自己的连笔字数据集来解决。

（2）输入的图片和数据集中的照片不同，这一点很有可能对正确率带来极大的影响，对于英语默写批改这种高精度要求的事情不是最合适。

针对这个问题，我们可能在后期尝试制作属于自己的数据集，或者在原始数据集中加入噪声。

另外试着为该程序完善前端，方便用户使用。

六、参考文献

[1] 深思海数\_willschang. Opencv-Python学习笔记二——cv2.GaussianBlur，cv2.Canny[EB/OL].[2019-6-9]. <https://www.jianshu.com/p/bfd5dd2566bb?tdsourcetag=s_pctim_aiomsg>

[2] ex2tron. Python+OpenCV教程12：腐蚀与膨胀[EB/OL].[2019-6-9]. <https://www.baidu.com/baidu?wd=ex2tron&tn=monline_4_dg&ie=utf-8>

[3] ex2tron. Python+OpenCV教程番外篇4：Otsu阈值法[EB/OL].[2019-6-9]. <https://www.jianshu.com/p/c6b0d22aa2d7?tdsourcetag=s_pctim_aiomsg>

[4] dz4543. Python OpenCV findContours()函数与drawContours()函数用法[EB/OL].[2019-6-9]. <https://blog.csdn.net/dz4543/article/details/80655067?tdsourcetag=s_pctim_aiomsg>

[5] hjxu2016. opencv-python中 boundingRect（cnt）以及cv2.rectangle用法[EB/OL].[2019-6-9]. <https://blog.csdn.net/hjxu2016/article/details/77833984?tdsourcetag=s_pctim_aiomsg>

[6] 我叫小随.3\_2018-11-11\_VGG16和VGG19介绍[EB/OL].[2019-6-9]. https://www.jianshu.com/p/e0845ecaf7f7