基于 Keras 的手写省份识别

一、 问题描述

如今我国物流快递行业飞速发展，为了提高快递分拣的效率，如何精确识别快递单中的通讯地址逐渐成为一个备受关注的研究课题。由于每种快递单上的地址位置相对固定，且分割出地址属于识别地址的边缘算法，故本文省去此部分。另外，由于时间因素，本文只针对全国 34 个省份（直辖市、自治区）的手写图片进行识别，只要构建了成熟的算法，识别城市只是时间问题。综上所述，本文旨在利用图像处理和神经网络，针对每个手写的省份图片给出其相对应的省份。

二、 模型的假设

（1）默认地址处图片已从快递单中提取

（2）针对手写省份图片进行识别

（3）手写图片无大块涂抹

（4）连笔字的字体结构清晰，不存在省略过多文字关键信息的情况

三、 模型的建立

1 手写省份图片的预处理（代码见 Image manipulate2.0.py）

经过讨论，我们发现手写文字的分割具有很大的风险和不确定性。如果将连通域或垂直投影间断作为分割工具，很难处理连笔字和文字间粘连的问题，而且部分汉字本身（如川）不是一个连通域，给文字分割以及后续的人工分类及训练样本的收集带来很大难度。故我们采用将文字统一处理、统一训练的方法。既能保证稳定性，又省去了添加文字间联系的算法（如湖的下一个字一定是南或北）。下面是具体处理过程，处理后的数据集见 dataset40（由于空间有限，只上传部分数据集）。

1.1 二值化

由于原始图片有背景亮度、字体颜色等与省份识别无关的因素，故我们经过灰度化与二值化将图片统一转换成黑色背景，白色文字的格式。由于图片背景和字体颜色的灰度不同，普通的固定阈值二值化会出现笔画粗细不等、背景出现大量噪声等问题。故在此处我们采用自适应阈值二值化函数 cv2.adaptiveThreshold，对灰度化后的图片进行二值化，使算法可以对不同亮度背景和文字都有很好的处理效果。但是由于其函数性质，在对笔画过粗的图片处理时，会将笔画中心部分去掉（图 1-1-1），因此我们读取固定阈值二值化中的图片信息，并将中心部分补全（图 1-1-2）。

图 1-1-1

图 1-1-2

二值化前后的处理图像如下：

图 1-1-3 原图

图 1-1-4 二值化后

1.2 开运算与闭运算

在二值化后图像中仍会有一些噪点（图 1-1-4），这时需要引入开运算来去除黑背景中的白噪点，引入闭运算来去除白背景中的黑噪点（图 1-2-1、图 1-2-2）

图 1-2-1 开运算后

图 1-2-2 闭运算后

1.3 最大边界矩形与去除字间隙

原始图片很难做到使其中的文字紧靠图片边缘，我们利用 opencv 中的最大边界矩形函数来保留矩形中的内容（图 1-3-1、图 1-3-2）。取到边界矩形后，字间隙和文字本身的比例不同，为了将与文字无关的变量降到最低，我们缩小所有垂直、水平间隙（定义为白像素点小于 8 的行或列）至 5 个像素。通过这样的操作，我们也可以降低无法去除的噪点（如在图片中误写的点）对最终尺寸归一化的影响（图 1-3-3）。

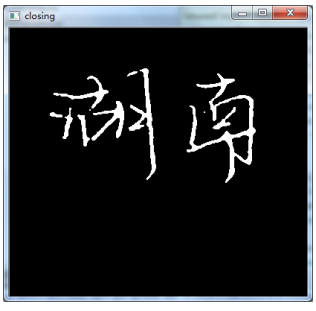


图 1-3-1 取最大边界前

图 1-3-2 取最大边界矩形后

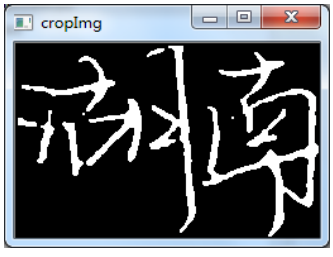


图 1-3-3 去除字间隙

1.4 尺寸归一化

为了满足神经网络输入的要求，我们统一将处理后的图片归一为 200*400 的大小，并将由于线性插值非纯黑白的点二值化为黑白点（图 1-4-1）。

图 1-4-1 图像归一化

2　图像的进一步处理

详见文件”图像的进一步处理.pdf”

3　神经网络的选择与训练（代码见 train.py, retrain.py, predict.py）

3.1 神经网络 ver1.0

基于先前大量的预处理，我们的神经网络其实要做的东西已经减少了很多，虽然这样减弱了神经网络的利用率，但另一方面我们也能简单地分析出我们需要的特征。所以我们选择了只把神经网络当做一个分类器，然后直接把处理后的图片拉伸成向量投入神经网络。以下是我们训练后的得到的结果。

```
max possible hunan
Yes!!
jilin tianjin hunan      possiblity:
[  1.48827303e-06   1.33328707e-04   9.99864817e-01]


max possible hunan
Yes!!
hubei neimenggu hunan      possiblity:
[  2.60465227e-09   2.16412587e-07   9.99999762e-01]


max possible hunan
Yes!!
tianjin hubei hunan      possiblity:
[  2.10190092e-06   7.45422076e-05   9.99922395e-01]


max possible hunan
Yes!!
hubei neimenggu hunan      possiblity:
[  3.88972882e-11   3.32638734e-08   1.00000000e+00]


Warning:I am confused,but I still give an answer which may be wrong!!
max possible hunan
Yes!!
sichuan shanghai hunan      possiblity:
[ 0.00544824  0.0063965   0.9878726 ]


Warning:I am confused,but I still give an answer which may be wrong!!
max possible hunan
Yes!!
sichuan xinjiang hunan      possiblity:
[ 0.041932    0.12845552  0.79558951]


max possible hunan
Yes!!
neimenggu hubei hunan      possiblity:
[  2.59761764e-05   2.08570535e-04   9.99765217e-01]


max possible hunan
Yes!!
taiwan hubei hunan      possiblity:
[  6.94640221e-08   4.34376489e-06   9.99995470e-01]


max possible hunan
Yes!!
tianjin neimenggu hunan      possiblity:
[  4.59256029e-08   6.16669655e-04   9.99383211e-01]


Warning:I am confused,but I still give an answer which may be wrong!!
max possible hunan
Yes!!
xizang hainan hunan      possiblity:
[  1.41444572e-04   8.93978123e-03   9.90874290e-01]
```

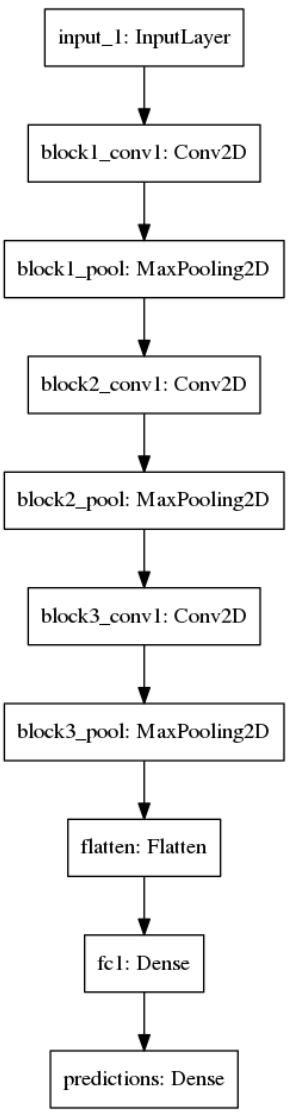
图 3-1-1

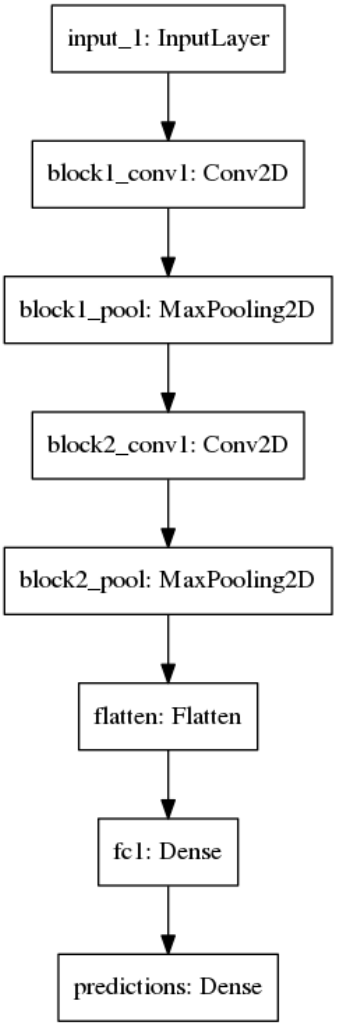
3.2 神经网络 ver1.1

由上图可以看出训练过程中给出的权重有时不接近 1.00，即训练集没有被很好地分类，因此我们就想到了在同一训练集上用不同的神经元结构训练出多个神经网络，由于不同的网络可能对不同的特征有更好或者更差的识别能力，所以将这些神经网络（分类器）联合起来，就可以给出一个更好的预测。（详见 4 多种神经网络构建的联级分类器）。

3.3 神经网络卷积版（CNN）ver1.0

我们经过查找资料对比分析得出结论，卷积神经网络明显可以比单纯的神经元更好的处理图像特征，在尝试 Resnet 失败之后，我们选择自己搭建神经网络，参考 vgg16 的网络构造，并且结合汉字的特殊性以及我们之前所做的一系列预处理，我们化简网络结构，构造了如下三个模型，并且一一进行训练，根据他们在测试集上的表现，最终选择了最后一个模型。





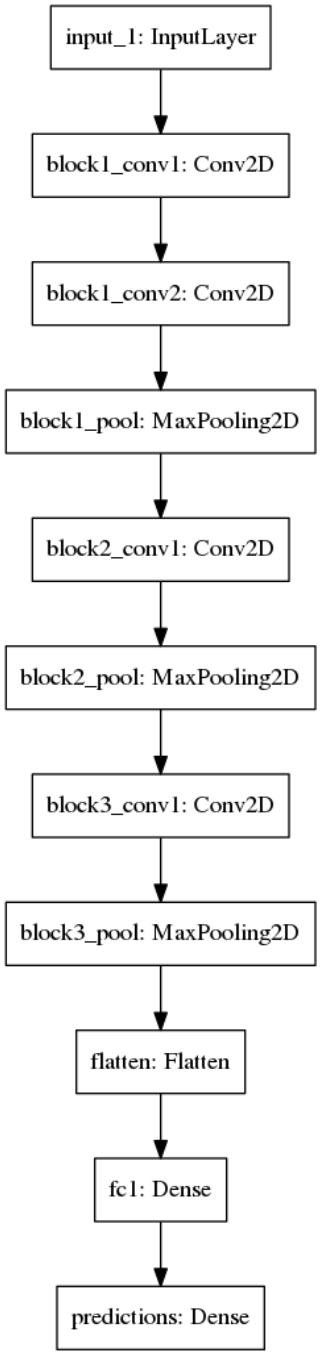
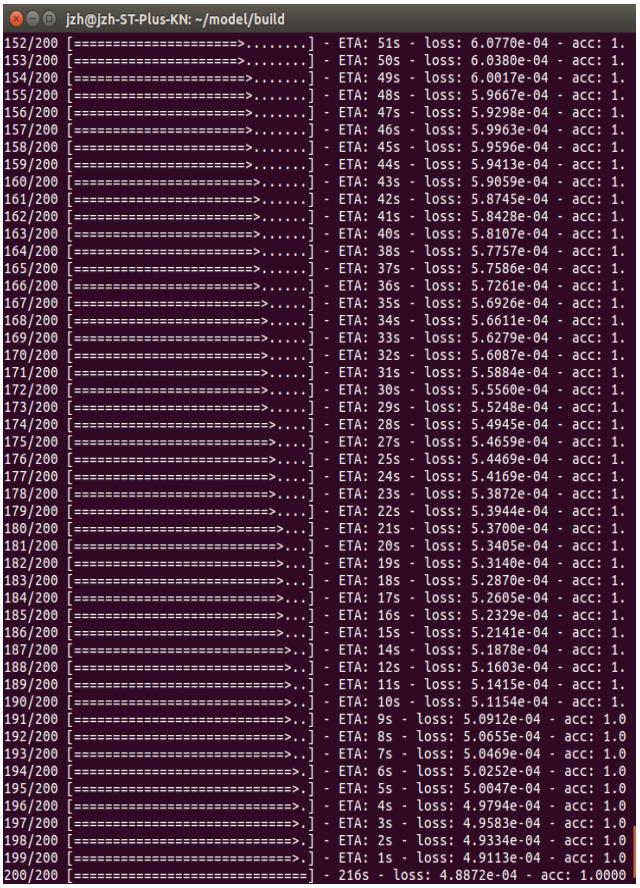


图 3-3-1

事实证明，深度神经网络确实可以一定程度上提高拟合的效果。



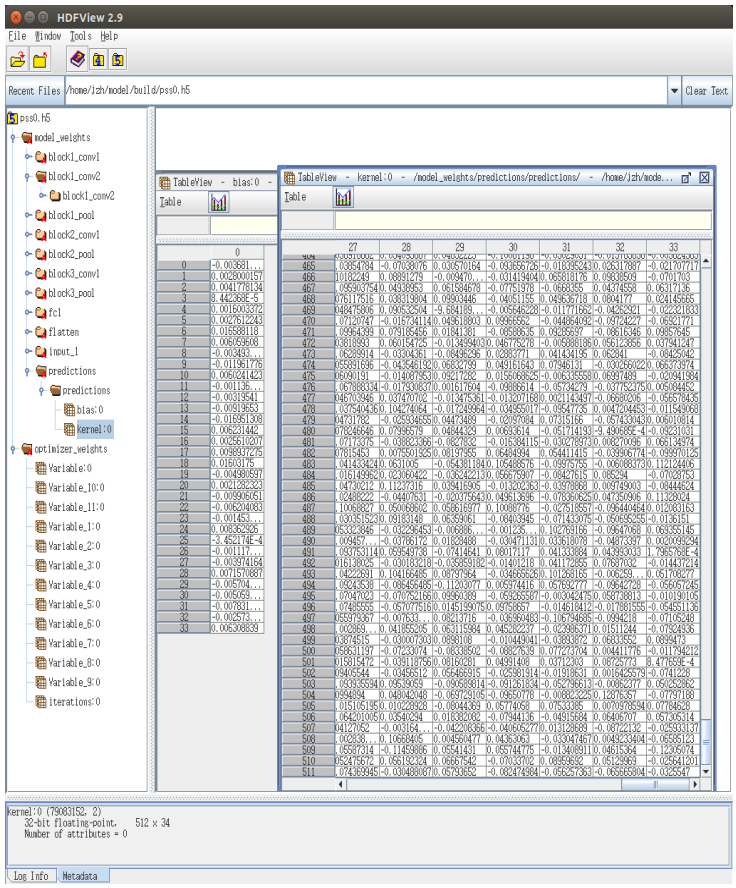


图 3-3-2　　图 3-3-3

以上是在训练集上训练后的结果和权重的示意图（图 3-3-2、图 3-3-3），可以看到经过充分训练之后，lossfunction 已达到较低的数值，并且在训练集上的准确率已经达到了 100%，已经是比较好的结果，为了防止过拟合，我们调整了代码，当整个训练集上的准确率达到 100%两个 epochs 之后，自动停止训练。

权重（图 3-3-3）反映了我们的最后一层，选用的是 512 个神经元的 dense 层，激活函数为 softmax，即分为 34 个类，此处的 512 也是经过调整后选取的值。

3.4 神经网络卷积版（CNN）ver1.1

我们在训练最后一个模型的同时，也附带训练了前两个比不上最后一个模型的卷积神经网络，但是有时候他们对一些比较特别的省份识别要更出色一点。比如说区分湖北和湖南，后者做的就比较好，但是区分湖北和浙江，前者就会更好一点，并且由于层数较小，前者的训练时间和难度也会降低很多，在数据集充分的情况下，前者也是可以达到很好的效果的。具体测试结果见第四章数值测试。

以下是他们在测试集上的表现，为了看的更具体，我们逐个对测试集的样本进行了判断，accu 是正确的个数，total 是总样本数。可以看出在别的组给的 3 个测试集上，分类器的有很好的表现。准确率分别达到了 95.6%，97.1%，92.1%。另一个图为测试集，已经过我们的预处理。

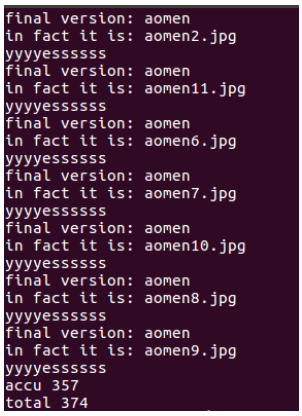


图 3-4-1

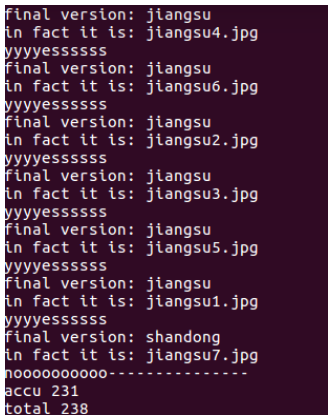


图 3-4-2

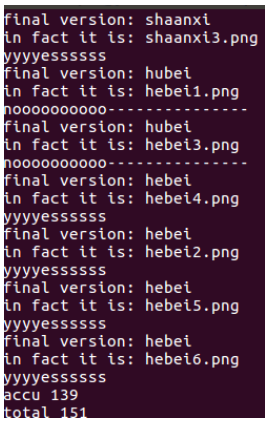


图 3-4-3

图 3-4-4

图 3-4-5

4 多种神经网络构建的联级分类器

在神经网络的训练中，选择不同参数及不同的优化器，产生了许多种分类器。在检测结果时，不同的分类器可能有不同的表现。如果一些分类器对某些类分辨效果好（记为 A 类），对其他类分辨效果差（记为 B 类），而且其他分类器对 A 类分辨效果差，对 B 类效果好。那么我们有必要将这两个分类器进行联合。在此本文提出一种对 AdaBoost 改进的算法（称为类 AdaBoost 算法），来构建多种神经网络的联级分类器。但在测试中，vgg16 的训练结果明显优于其他神经网络，故此处算法未编程实现，仅给出详细的算法流程，以便将来需要的时候使用。

给定一个省份图片的数据集 T

$$T=\{(x_1,y_1),(x_2,y_1),\cdots,(x_{s_1},y_1),(x_2,y_2),\cdots,(x_{s_2},y_2),\cdots(x_{s_{34}},y_{34})\}$$

其中 $x_j\in\chi$ 为每个省份的手写图片， y_i 用来标记图片对应的省份。每个神经网络分类器的输出是对于每个图片给出属于 34 个类的概率向量。对于第 m 个分类器和数据 $\{(x_1,y_i),(x_2,y_i),\cdots,(x_{s_i},y_i)\}$，我们记 $\{x_1,x_2,\cdots,x_{s_i}\}$ 属于 y_i 的平均概率为 $p_{m,i}$。

类 Adaboost 算法的流程如下：

步骤 1.首先，初始化训练数据的权值分布。每一个训练样本最开始时都被赋予相同的权值：1/N，其中 N=34。

$$D_1 = (w_{11}, w_{12}, \cdots, w_{1N}), w_{1i} = \frac{1}{N}, i = 1, 2, \cdots, N$$

步骤 2.进行多轮迭代，用 m = 1,2, ..., M 表示迭代的第多少轮

a. 使用具有权值分布 D_m 的训练数据集学习，得到基本分类器（选取整体错误率最低的分类器作为基本分类器）：$G_m(x): \chi \to R^N, x \mapsto (q_1, q_2, \cdots, q_N)$

其中 q_i 为图片属于 y_i 类的概率，满足 $\sum_i q_i = 1$

b. 计算 $G_m(x)$ 在训练数据集上的分类误差

$$e_m = P(\| G_m(x_i) \|_\infty \neq y_i) = \sum_{i=1}^{N} w_{mi}(1 - p_{m,i})$$

由上述式子可知，$G_m(x)$ 在训练数据集上的误差率 e_m 就是被 $G_m(x)$ 误分类样本的权值之和。

c. 计算 $G_m(x)$ 的系数，α_m 表示 $G_m(x)$ 在最终分类器中的重要程度（目的：得到基本分类器在最终分类器中所占的权重）：$\alpha_m = \frac{1}{2}\log\frac{1-e_m}{e_m}$

由上述式子可知，、$e_m \le \frac{1}{2}$ 时，$\alpha_m \ge 0$，且 α_m 随着 e_m 的减小而增大，意味着分类误差率越小的基本分类器在最终分类器中的作用越大。

d. 更新训练数据集的权值分布（目的：得到样本的新的权值分布）用于下一轮迭代（这里我们将 $p_{m,i} > 0.5$ 定义为判断正确）

$$D_{m+1} = (w_{m+1,1}, w_{m+1,2}, \cdots, w_{m+1,N})$$

$$w_{m+1,i} = \frac{w_{mi}}{Z_m}\exp(-\alpha_m \operatorname{sgn}(p_{m,i} - 0.5)), i = 1, 2, \cdots, N$$

其中，Z_m 是规范化因子，使得 D_{m+1} 成为一个概率分布：

$$Z_m = \sum_{i=1}^{N} w_{mi}\exp(-\alpha_m \operatorname{sgn}(p_{m,i} - 0.5))$$

使得被基本分类器 $G_m(x)$ 误分类样本的权值增大，而被正确分类样本的权值减小。就这样，通过这样的方式，类 AdaBoost 方法能“重点关注”或“聚焦于”那些

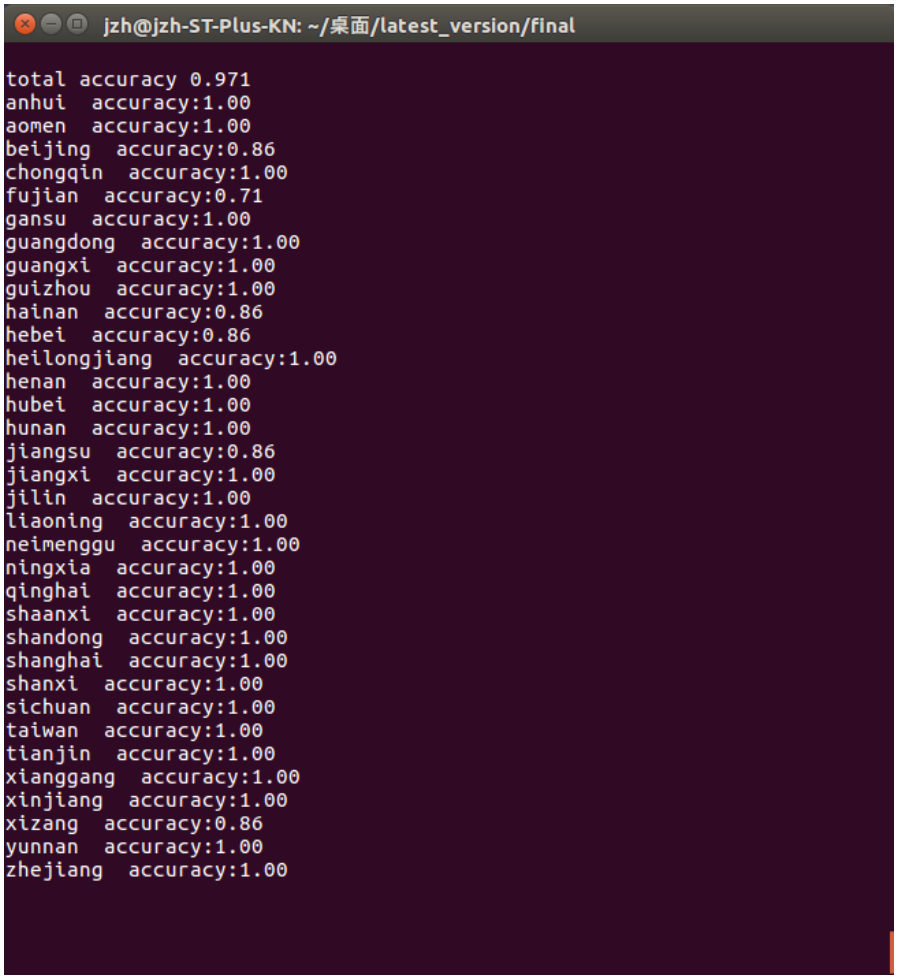
较难分的样本上。

步骤 3. 组合各个弱分类器从而得到最终分类器，如下：$G(x)=\sum_{m=1}^{M}\alpha_m G_m(x_i)$

其中最大的分量为最终输出结果。

四、 数值测试

我们将 testdata7 中的数据集（每个省份 7 张图片）进行预处理，并载入神经网络进行测试，以下是他们在测试集上的表现，total accuracy 为分类器的总正确率，其他为各省份的正确率。可以看出在此测试集上的总体，分类器的有很好的表现。准确率达到了 97.1%，但不同省份的识别准确率有所不同，部分类别如北京、江苏等错误率略高于其他类别。（对测试集每张图片的识别结果见“测试结果.pdf”）



五、 结果分析

经过数次改进，神经网络的总体识别正确律逐渐上升，在测试集上的结果也比较可观，最后参考 vgg16 结构来搭建的卷积神经网络已经较为成熟。从数据集的角度来看，模型中的训练集为 34 类*90 张共 3060 张图片，测试集为 34 类*7 张共 238 张图片。在收集数据的过程中，我们曾用每类 40 张的训练集来构造分类器，准确率在 90%左右，在将训练集拓展到每类 90 张后，准确度达到了 97%。可见，现在训练集还远没有达到理想的数量。

分类器较难区分的省份：（海南、湖南、河南），（浙江、湖北、河北），（北京、山东、江苏），（宁夏、西藏）。给出的建议是在这些省份的图片集中加大数据量，有可能可以起到更好的效果。或者是单独在训练几个区分这些省份的分类器，比如一旦主分类器将省份归类到湖南，那么就调用这个次级分类器，对省份再次识别，以达到更高的准确率。事实上，我们有一个考虑，就是对 maxpooling 池化层的改进，可能造成例如湖北，浙江这两组文字识别困难的主要原因就是 maxpooling 把他们的区别抹掉了，所以在单独构建分类器的时候可以考虑使用较少的池化层。但是对于总分类器，若只用一个池化层却又会大大降低效果。所以网络的构建还需要更深入的考虑。

六、 模型评价

本文提出了一种基于神经网络和特征提取的两种分类方法。我们针对手写汉字进行了归一化处理，减小了文字特征以外的其他特征带来的影响。并查找资料，根据汉字笔画方向（横、竖、撇、捺）的特殊性，提出了一种将汉字笔画转化为特征矩阵的方法。在日后的研究中，可以从特征矩阵入手，进行手写汉字分类。本文直接从图片入手，将手写汉字归一化处理后投入构建好的神经网络中进行训练，得到了比较满意的结果。由于训练数据有限，与投入实际应用的水准仍有差距。在进一步扩大训练集的情况下，本模型中的神经网络会有更好的表现。并且本文基于 AdaBoost 的算法原理，提出了一种将多个神经网络分类器构成联级分类器的算法，可以使各个分类器在其分类能力强的类别中获得更高的权重。

七、 参考文献

【1】 王正群. 手写体汉字识别研究[D]. 南京理工大学, 2001.

【2】 OpenCV Python Tutorial

【3】 Keras 中文手册 2017