**[Tesseract-OCR 字符识别---样本训练 [转]](https://www.cnblogs.com/samlin/p/Tesseract-OCR.html)**

Tesseract是一个开源的OCR（Optical Character Recognition，光学字符识别）引擎，可以识别多种格式的图像文件并将其转换成文本，目前已支持60多种语言（包括中文）。 Tesseract最初由HP公司开发，后来由Google维护，目前发布在Googel Project上。地址为<http://code.google.com/p/tesseract-ocr/>。

**使用默认的语言库识别**

1.安装Tesseract

        从<http://code.google.com/p/tesseract-ocr/downloads/list>下载Tesseract，目前版本为Tesseract3.02。因为只是测试使用，这里直接下载winodws下的安装文件tesseract-ocr-setup-3.02.02.exe。安装成功后会在相应磁盘上生成一个Tesseract-OCR目录。通过目录下的tesseract.exe程序就可以对图像字符进行识别了。

2.准备一副待识别的图像，这里用画图工具随便写了一串数字，保存为number.jpg，如下图所示：

http://img.my.csdn.net/uploads/201212/25/1356422284_1112.jpg

3.  打开命令行，定位到Tesseract-OCR目录，输入命令：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/feihu521a/article/details/8433077)

1. tesseract.exe number.jpg result -l eng

tesseract.exe number.jpg result -l eng

     其中result表示输出结果文件txt名称，eng表示用以识别的语言文件为英文。

3.  打开Tesseract-OCR目录下的result.txt文件，看到识别的结果为7542315857，有3个字符识别错误，识别率还不是很高，那有没有什么方法来提供识别率呢？Tesseract提供了一套训练样本的方法，用以生成自己所需的识别语言库。下面介绍一下具体训练样本的方法。



**训练样本**

关于如何训练样本，Tesseract-OCR官网有详细的介绍<http://code.google.com/p/tesseract-ocr/wiki/TrainingTesseract3>。这里通过一个简单的例子来介绍一下如何进行样本训练。

**1.下载工具jTessBoxEditor.** <http://sourceforge.net/projects/vietocr/files/jTessBoxEditor/>，这个工具是用来训练样本用的，由于该工具是用JAVA开发的，需要安装JAVA虚拟机才能运行。

**2. 获取样本图像。**用画图工具绘制了5张0-9的文样本图像（当然样本越多越好），如下图所示：

http://img.my.csdn.net/uploads/201212/25/1356423737_1361.jpg

http://img.my.csdn.net/uploads/201212/25/1356423748_2259.jpg

http://img.my.csdn.net/uploads/201212/25/1356423753_4094.jpg

http://img.my.csdn.net/uploads/201212/25/1356423756_8982.jpg

http://img.my.csdn.net/uploads/201212/25/1356423760_4557.jpg

**3.合并样本图像**。运行jTessBoxEditor工具，在点击菜单栏中Tools--->Merge TIFF。在弹出的对话框中选择样本图像（按Shift选择多张），合并成num.font.exp0.tif文件。4.生成Box File文件。打开命令行，执行命令：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/feihu521a/article/details/8433077)

1. tesseract.exe num.font.exp0.tif num.font.exp0 batch.nochop makebox

tesseract.exe num.font.exp0.tif num.font.exp0 batch.nochop makebox

  生成的BOX文件为num.font.exp0.box，BOX文件为Tessercat识别出的文字和其坐标。

注：Make Box File的命令格式为：

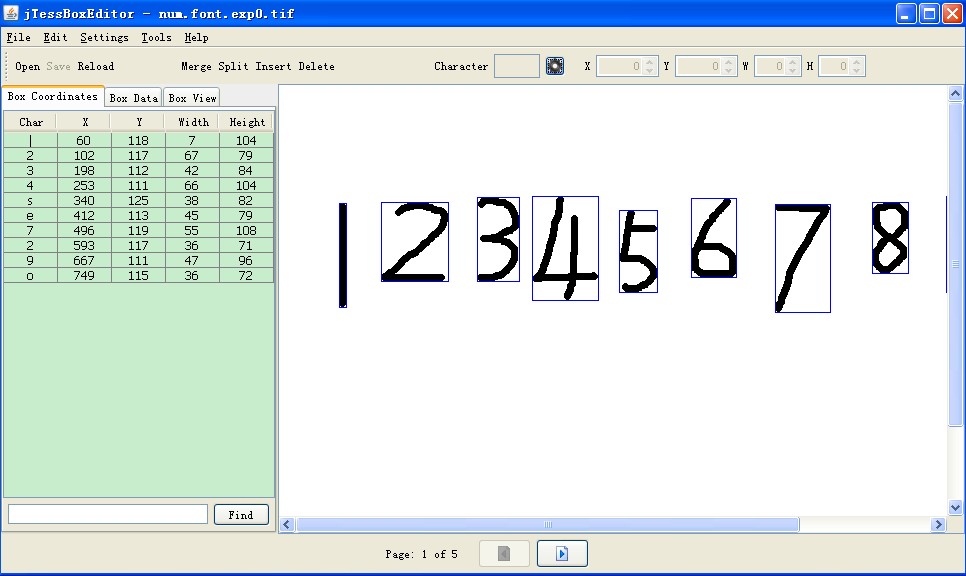
**[plain]** [view plaincopyprint?](http://blog.csdn.net/feihu521a/article/details/8433077)

1. tesseract [lang].[fontname].exp[num].tif [lang].[fontname].exp[num] batch.nochop makebox

tesseract [lang].[fontname].exp[num].tif [lang].[fontname].exp[num] batch.nochop makebox

其中lang为语言名称，fontname为字体名称，num为序号，可以随便定义。

**5.文字校正**。运行jTessBoxEditor工具，打开num.font.exp0.tif文件（必须将上一步生成的.box和.tif样本文件放在同一目录），如下图所示。可以看出有些字符识别的不正确，可以通过该工具手动对每张图片中识别错误的字符进行校正。校正完成后保存即可。



**6.定义字体特征文件。**Tesseract-OCR3.01以上的版本在训练之前需要创建一个名称为font\_properties的字体特征文件。

font\_properties不含有BOM头，文件内容格式如下：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/feihu521a/article/details/8433077)

1. <fontname> <italic> <bold> <fixed> <serif> <fraktur>

<fontname> <italic> <bold> <fixed> <serif> <fraktur>

其中fontname为字体名称，必须与[lang].[fontname].exp[num].box中的名称保持一致。<italic> 、<bold> 、<fixed> 、<serif>、 <fraktur>的取值为1或0，表示字体是否具有这些属性。

这里在样本图片所在目录下创建一个名称为font\_properties的文件，用记事本打开，输入以下下内容：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/feihu521a/article/details/8433077)

1. font 0 0 0 0 0

font 0 0 0 0 0

这里全取值为0，表示字体不是粗体、斜体等等。 **7.生成语言文件。**在样本图片所在目录下创建一个批处理文件，输入如下内容。

**[plain]** [view plaincopyprint?](http://blog.csdn.net/feihu521a/article/details/8433077)

1. rem 执行改批处理前先要目录下创建font\_properties文件
3. echo Run Tesseract for Training..
4. tesseract.exe num.font.exp0.tif num.font.exp0 nobatch box.train
6. echo Compute the Character Set..
7. unicharset\_extractor.exe num.font.exp0.box
8. mftraining -F font\_properties -U unicharset -O num.unicharset num.font.exp0.tr
10. echo Clustering..
11. cntraining.exe num.font.exp0.tr
13. echo Rename Files..
14. rename normproto num.normproto
15. rename inttemp num.inttemp
16. rename pffmtable num.pffmtable
17. rename shapetable num.shapetable
19. echo Create Tessdata..
20. combine\_tessdata.exe num.

rem 执行改批处理前先要目录下创建font\_properties文件

echo Run Tesseract for Training..

tesseract.exe num.font.exp0.tif num.font.exp0 nobatch box.train

echo Compute the Character Set..

unicharset\_extractor.exe num.font.exp0.box

mftraining -F font\_properties -U unicharset -O num.unicharset num.font.exp0.tr

echo Clustering..

cntraining.exe num.font.exp0.tr

echo Rename Files..

rename normproto num.normproto

rename inttemp num.inttemp

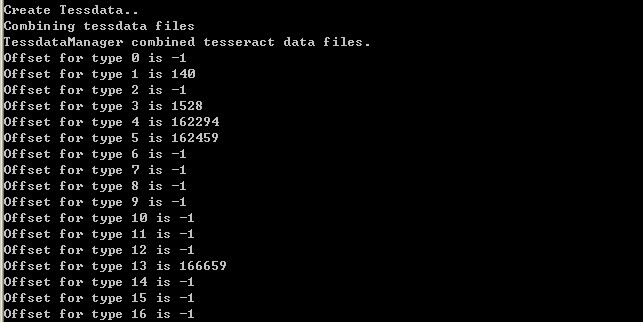
rename pffmtable num.pffmtable

rename shapetable num.shapetable

echo Create Tessdata..

combine\_tessdata.exe num.

将批处理通过命令行执行。执行后的结果如下：



需确认打印结果中的Offset 1、3、4、5、13这些项不是-1。这样，一个新的语言文件就生成了。

num.traineddata便是最终生成的语言文件，将生成的num.traineddata拷贝到Tesseract-OCR-->tessdata目录下。可以用它来进行字符识别了。

**使用训练后的语言库识别**

用训练后的语言库识别number.jpg文件， 打开命令行，定位到Tesseract-OCR目录，输入命令：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/feihu521a/article/details/8433077)

1. tesseract.exe number.jpg result -l eng

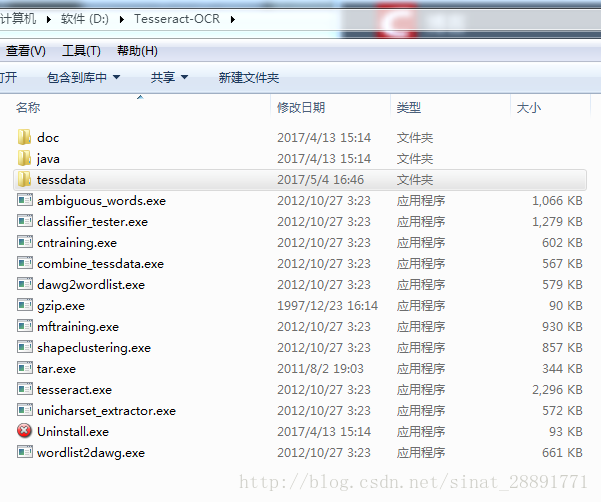
tesseract.exe number.jpg result -l eng

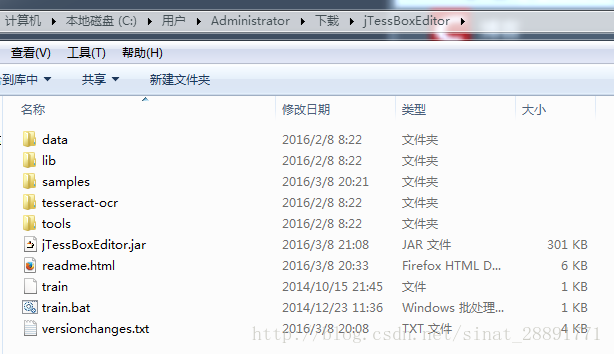
识别结果如如图所示，可以看到识别率提高了不少。通过自定义训练样本，可以进行图形验证码、车牌号码识别等。感兴趣的朋友可以研究研究。

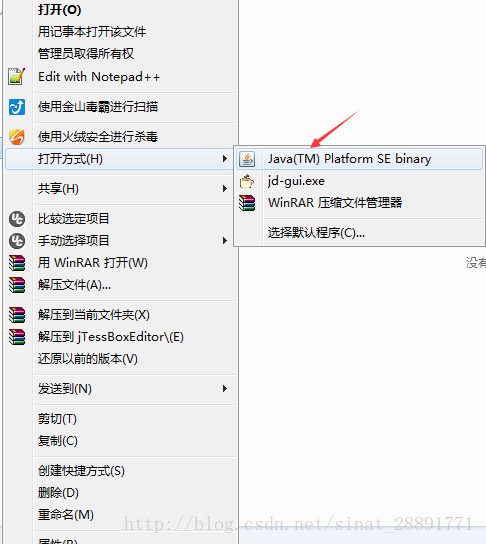


转载请注明出处［<http://samlin.cnblogs.com/>］

## ##################################################################利用jTessBoxEditor工具进行Tesseract3.02.02样本训练，提高识别率

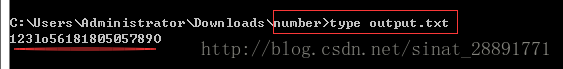
**1 . 下载Tesseract-OCR(相关版本自行选择)**   
得到目录结构如下:   
   
tessdata目录为相关的语言包文件目录

**2 .下载jTessBoxEditor(运行环境为java虚拟机)**   
得到目录结构如下:   


打开方式如下:   


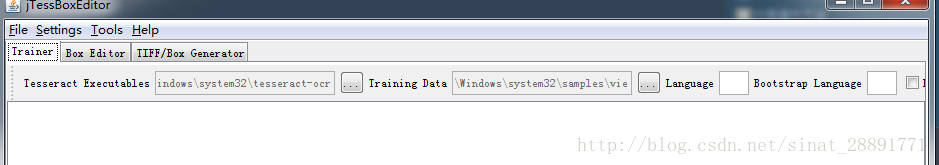
工具都安装完成了,我们接下来测试下tesseract   
我的测试图片如下:   
这里写图片描述

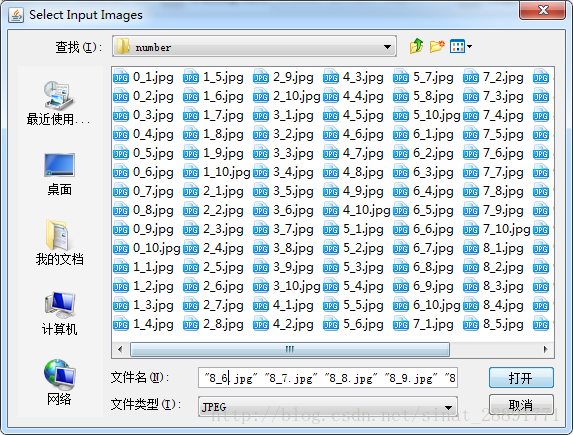
**在命令行输入 : tesseract b.jpg output -l eng -psm 7**   
这里写图片描述   
其中eng为你tesseract目录tessdata里的已有的eng语言包

**接下来把识别的结果用文本打印出来,输入:type output.txt**   


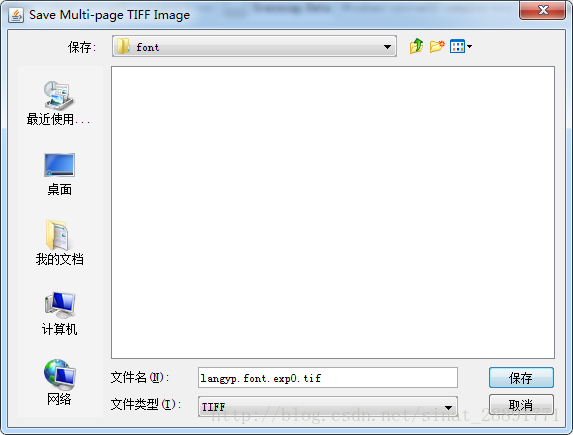
可以看到eng的语言包识别有错误的结果,4识别成了1o

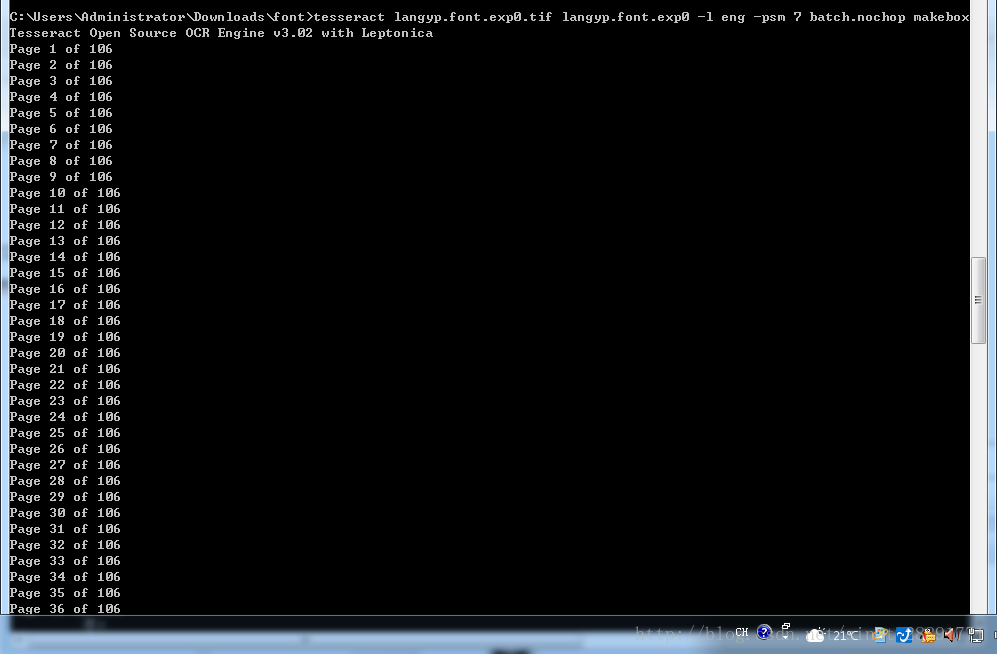
所以在我们开发中针对我们要识别的字体,进行我们自己训练的语言是很有必要的

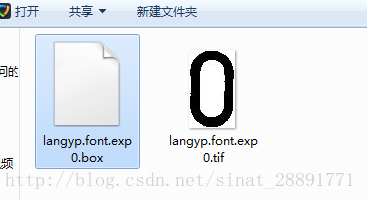
**3 . 训练自己的语言包文件**   
**a. 用jTessBoxEditor把要训练样本图片文件合并成tif文件(样本图片一定要为有效的格式图片)**   
运行jTessBoxEditor程序,界面如下:   


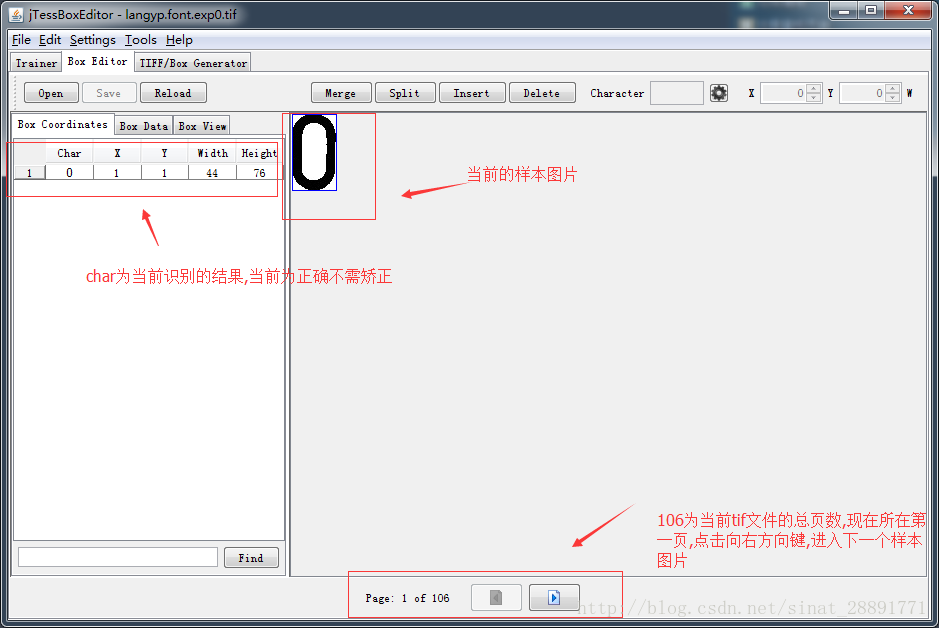
点击顶栏的Tools选项,选择Merge TIFF.. 进入你要训练的样本图片所在的目录,点击Ctrl+Alt+A,选择所有图片点击打开:   


然后保存文件名到指定目录,我这里保存的文件名为: **langyp.font.exp0.tif**

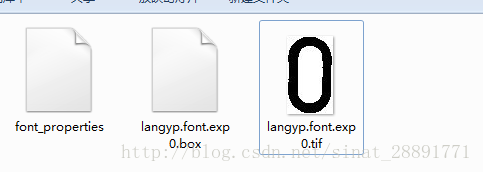


**b.生成Box文件**   
打开cmd,到你langyp.font.exp0.tif文件所在目录,执行**tesseract langyp.font.exp0.tif langyp.font.exp0 -l eng -psm 7 batch.nochop makebox** 命令   


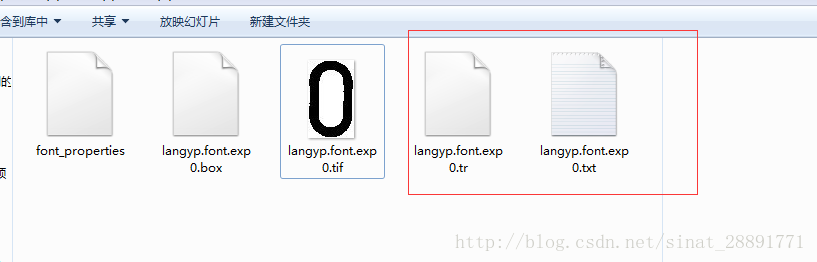
结果生成了**langyp.font.exp0.box**文件   


**C. 对样本图片用jTessBoxEditor工具进行矫正**   
点击jTessBoxEditor工具的Box Editor选项,点击下方的open选项,打开刚刚生成的langyp.font.exp0.tif文件,结果如下:   


右侧为对应的Box文件数据,如果char的字符和当前的样本图片一致时就进行矫正,修改char里的字符,然后进行save,这样就矫正了,进入下张样本图片时,同样,矫正后点击save,当所有样本图片都矫正了,这一步也就完成了

**d.生成font\_properties文件(该文件没有后缀名)**   
在命令行执行:**echo font 0 0 0 0 0 >font\_properties**   
结果生成了font\_properties文件   
   
内容为字体名font，后面带5个0，分别代表字体的粗体、斜体等属性，这里全部是0

**e.生成.tr训练文件**   
在命令行执行: **tesseract langyp.font.exp0.tif langyp.font.exp0 -l eng -psm 7 nobatch box.train**

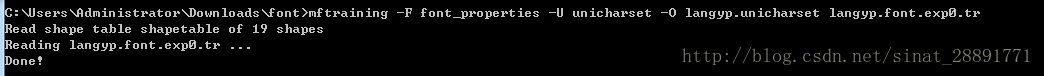


会生成框内两个文件

**f.生成字符集文件**   
在命令行执行 : **unicharset\_extractor langyp.font.exp0.box**   


结果生成了unicharset文件

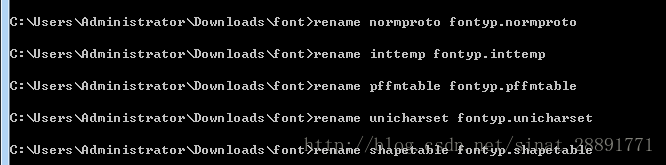
**g.生成shape文件**   
在命令行执行 : **shapeclustering -F font\_properties -U unicharset -O langyp.unicharset langyp.font.exp0.tr**   
  
结果生成了shapetable文件和langyp.unicharset文件

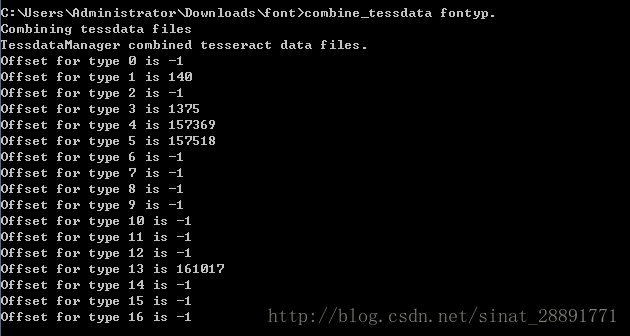
**h.生成聚集字符特征文件**   
在命令行执行: **mftraining -F font\_properties -U unicharset -O langyp.unicharset langyp.font.exp0.tr**   


结果生成了pffmtable,inttemp,unicharset文件

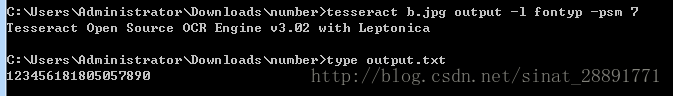
**i.生成字符正常化特征文件**   
在命令行执行: **cntraining langyp.font.exp0.tr**   


结果生成了normproto文件

**j.把h,i步骤生成的文件用rename命令进行更名**   
在命令行执行:   
\*\*rename normproto fontyp.normproto   
rename inttemp fontyp.inttemp   
rename pffmtable fontyp.pffmtable   
rename unicharset fontyp.unicharset   
rename shapetable fontyp.shapetable\*\*   


**k.合并训练文件**   
在命令行执行: **combine\_tessdata fontyp**.   


**l.将fontyp.traineddata文件拷贝至Tesseract-OCR文件夹里的tessdata语言包文件夹里**

**测试 :**   
cmd操作回到样本图片的目录,如果没有你当前目录没有该样本图片,执行命令时会提示”Cannot open input file: b.jpg”,还有执行指定的新的语言包,Tesseract-OCR目录的tessdata的文件夹必须要有新的语言包文件   
执行命令: **tesseract b.jpg output -l fontyp -psm 7**(fontyp为训练的语言包文件)   
最后执行 : **type output.txt**   


可以看见没有一个错误,最后用其它的样本进行识别,都能完全识别,用默认的eng语言包识别错误的”4”,用新的语言包识别后都没有错

本文档训练的语言包文件可用作身份证识别的语言包文件和车牌识别的语言包文件!