# Tesseract训练文档

## 环境配置

* tesseract-ocr版本：3.0以上（3.0以下不支持中文识别）
* chi\_sim.traindata字库：下载并放在tesseract-ocr安装目录的tessdata文件夹里
* jTessBoxEditor训练工具：用于修改box文件，java -jar jTessBoxEditor.jar即可打开jar包

## 识别步骤

* 打开dos命令行，进入要识别的图片所在的路径下
* 输入命令

tesseract 图片名称 生成的结果文件的名称 字库

例如 识别 test.jpg 图片，生成output.txt 文件，用正文字库识别

tesseract test.jpg output.txt -l chi\_sim

注意：-l 必需有且不是数字1

提高识别可以在最后加上-psm 默认为3

-psm

0 =仅定向和脚本检测（OSD）。

1 =自动页面分割与OSD。

2 =自动页面分割，但没有OSD或OCR。

3 =全自动页面分割，但没有OSD。（默认）

4 =假设可变大小的单列文本。

5 =假设垂直排列文本的单个统一块。

6 =假设单个统一的文本块。

7 =将图像视为单个文本行。

8 =将图像视为单个字。

9 =将图像视为一个单一的单词。

10 =将图像视为单个字符。

有时候识别不准确，训练字库

## 训练步骤

### 准备工作

* 下载安装java 环境，若不想这样，也可以下载带jre的jTessBoxEditor
* 下载jTessBoxEditor(依赖于java)
* 安装tesseract 3.0 以上，最好是最新版本，因为最新版本对中文支持更好

### 步骤

* 准备你的训练文本
* 将文本转为image+box文件.(如果你已经有image文件的话，只需要手动生成box文件)
* 生成unicharset文件
* 有选择性的生成字典数据
* 运行tesseract来处理之前的image+box文件生成一个训练数据集合
* 在训练数据集合的基础上进行训练
* 合并数据文件

#### 实例

* 如果是其他图片格式，将其转为tif格式，windows自带画图就可以，画图打开，另存为，选择格式。
* 如果是多张图片，使用之前安装jTessBoxEditor工具将多张图片合并为一张（菜单栏 Tools → Merge TIFF）。并按照格式 [lang].[fontname].exp[num] 重命名合并后的文件，例如命名为 chi.fangzheng.exp0.tif。
* 生成box文件

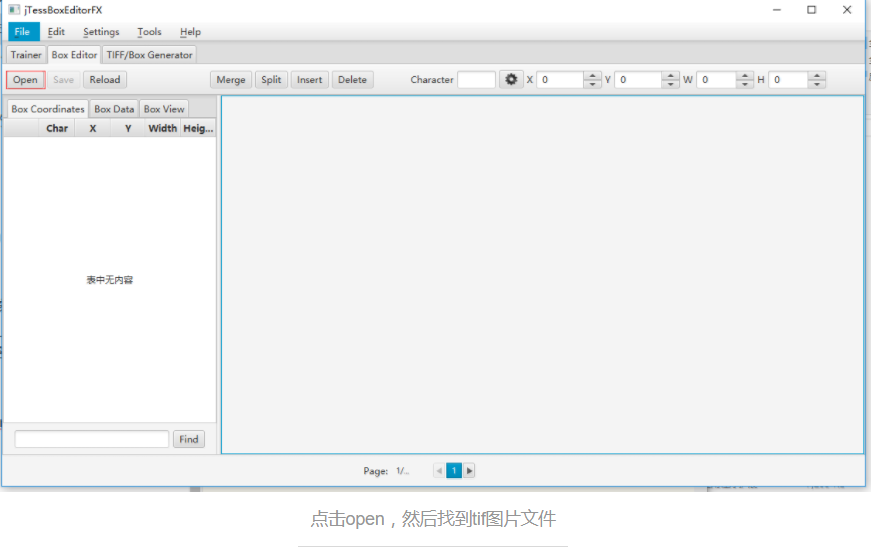


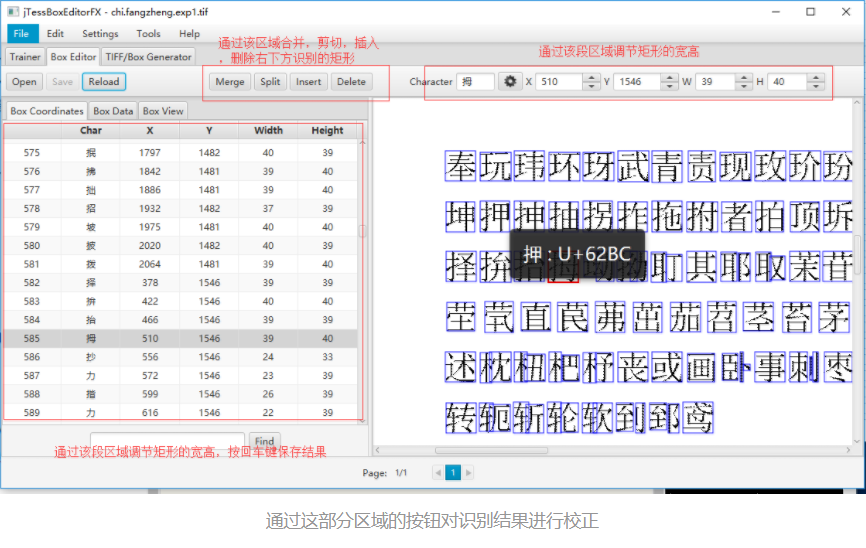
输入路径和输出路径文件名（除了后缀）应该保持一致。  
因为是要训练中文所以还需要加上-l chi\_sim（l代表language chi\_sim是放在tessdata目录下中文简体字体名的前缀），实际命令如下所示

tesseract.exe chi.fangzheng.exp0.tif chi.fangzheng.exp0 -l chi\_sim batch.nochop makebox

* 矫正box文件

打开之前安装的jTessBoxEditor





校正完之后点击保存

* 生成unicharset文件

使用刚才修改正确后的 box 文件，对 Tesseract 进行训练，生成 .tr 文件

//tesseract.exe [tif图片文件名] [生成的tr文件名] nobatch box.train

tesseract.exe chi.fangzheng.exp0.tif chi.fangzheng.exp0 nobatch box.train

生成Character集合

//unicharset\_extractor.exe [box文件名]

unicharset\_extractor.exe chi.fangzheng.exp0.box

如果有多个图片的话，则需要合并生成1个Character集合,命令如下

//unicharset\_extractor.exe [1个box文件名] [1个box文件名] .....

unicharset\_extractor.exe chi.fangzheng.exp0.box chi.fangzheng.exp1.box

* 创建字体特征文件

定义字体特征文件，Tesseract-OCR 3.01 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

<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。

//本次示例

fangzheng 0 0 0 0 0

* 生成字典数据

如果是单个依次输入下面两条命令,多个文件则输入多个tr

mftraining.exe -F font\_properties -U unicharset -O chi.unicharset chi.fangzheng.exp0.tr

//mftraining.exe -F font\_properties -U unicharset -O chi.unicharset [第一个tr] [第二个]...

cntraining.exe chi.fangzheng.exp0.tr

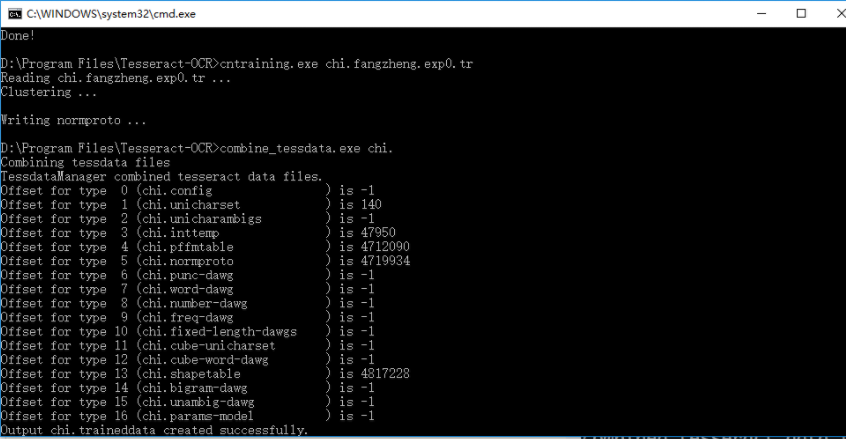
//cntraining.exe [第一个tr] [第二个]...

接下来手工修改 Clustering 过程生成的 4 个文件（inttemp、pffmtable、normproto、shapetable）的名称为 [lang].xxx。例如我这里改为 chi.inttemp、chi.pffmtable、chi.normproto、chi.shapetable。

* 合并数据文件

生成语言文件

combine\_tessdata chi.



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。

chi.traineddata 便是最终生成的语言文件，将生成的 chi.traineddata 文件拷贝到 tessdata 目录下，就可以用它来进行字符识别了。

* 测试

//tesseract [图片文件名] [需要输出的文本文档的文件名] -l [识别的语言]

tesseract chi.fangzheng.exp0.tif out -l chi

2017/08/05

谢谢使用