**目录**

1. OCR光学识别相关技术 1

1.1 OCR光学识别技术概述 1

1.2图像预处理技术 1

1.2.1灰度化 2

1.2.2二值化 2

1.3传统OCR光学识别技术 4

1.4深度学习的OCR光学识别技术 5

1.4.1 CTPN网络文本区域检测概述 5

1.4.2 CRNN网络文本信息识别概述 5

1. 基于Android移动端的OCR光学识别系统 6

2.1移动端OCR光学识别系统流程图以及架构图 6

2.2 光学识别系统的客户端 7

2.2.1 图像输入模块 8

2.2.2 图像预处理模块 8

2.2.3 图像发送至服务器端模块 9

2.2.4 移动端离线光学识别模块 9

2.3 光学识别系统的服务器端 10

2.3.1 文本区域检测模块 11

2.3.2 文本识别模块 11

1. OCR光学识别系统效果展示 12

3.1基于Tesseract离线COR光学识别 12

3.2基于CTPN区域检测、CRNN文本识别的OCR光学识别 13

1. 开发过程中的问题及解决方案 14

4.1 问题一 14

4.2 问题二 15

第五章 总结 15

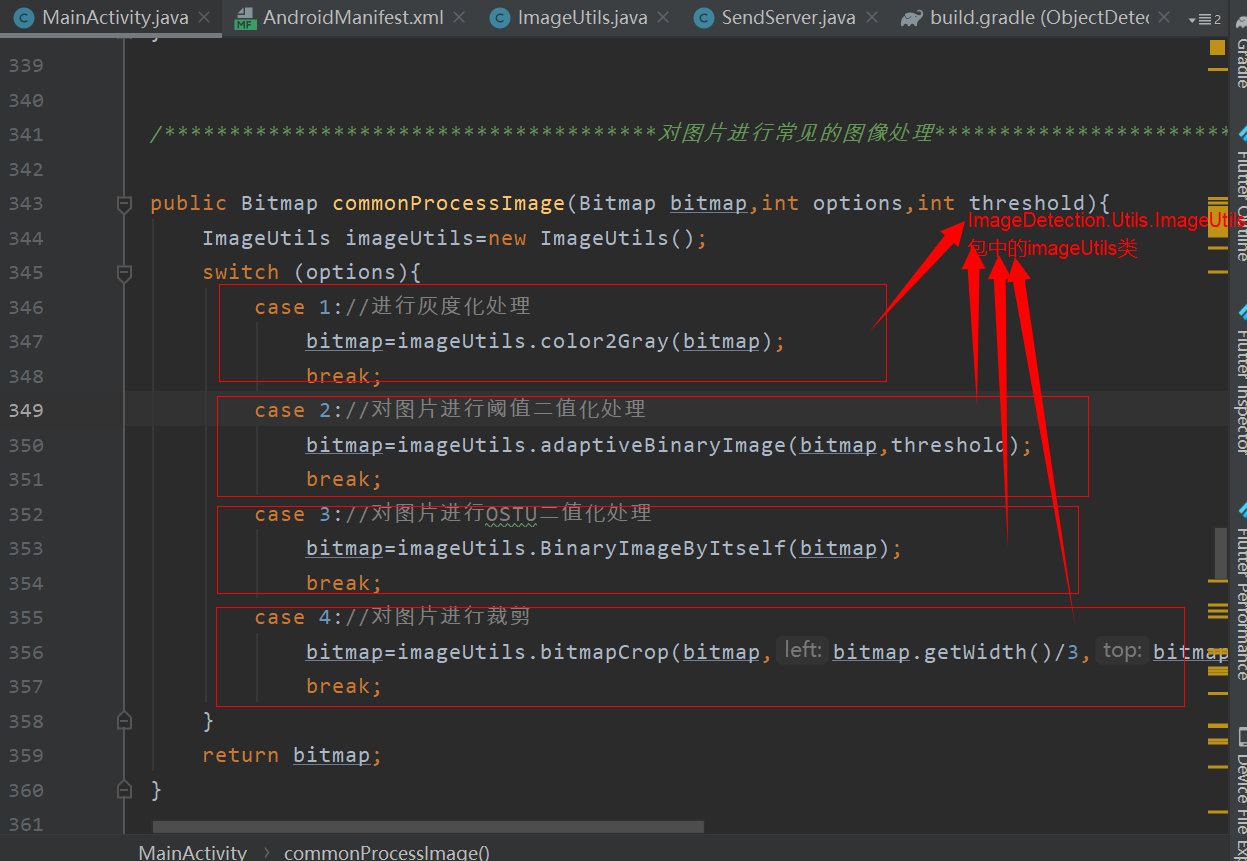
# 1.**OCR**光学识别相关技术

## 1.1.OCR光学识别技术概述

# OCR光学识别是指电子设备(例如扫描仪或数码相机)检查纸上打印的字符，通过检测暗、亮的模式确定其形状，然后用字符识别方法将形状翻译成计算机文字的过程。随着智能手机的逐渐普及，人手一台智能手机已经是常态，因此基于此社会现状，我开发了一套C/S移动端OCR光学识别系统，使得人们可以随时随地拍下图片将其转化为字符，大大减少文本输入时间。

## 1.2图像预处理技术

拍照或者相册选中的图片不能直接作为识别对象,必须经过图像预处理后，才能作为系统的输入,图像预处理方法主要有灰度处理和二值化处理。在该移动端OCR光学识别技术中实现了灰度化，阈值二值化，OSTU自适应二值化，以及裁剪等预处理方法。

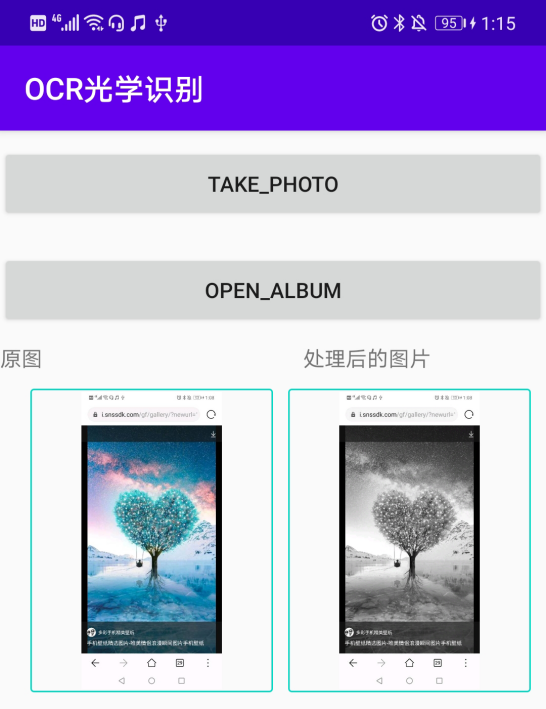


### 1.2.1灰度化

现在大部分的彩色图像都是采用RGB颜色模型，每个维度上的像素值的范围为[0,255],其中白色为255，黑色为0。对彩色图像进行灰度化处理，通俗的讲就是使图像没有了色彩即RGB色彩分量全部相等R=G=B，灰度化的主要方法有分量法、最大值法、平均值法、加权平均法。在本系统中采用的是加权平均法，其公式如下:



加权平均法实现灰度化效果图:



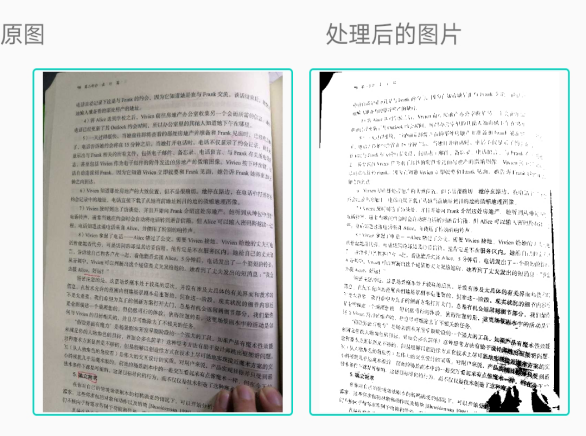
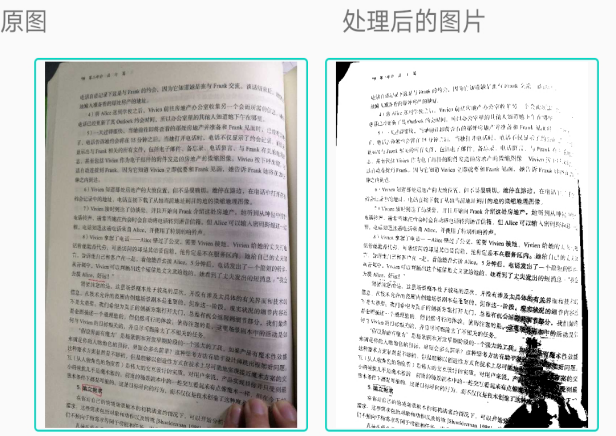
### 1.2.2二值化

图像二值化就是将图像上的[像素](https://baike.baidu.com/item/%E5%83%8F%E7%B4%A0/95084" \t "https://baike.baidu.com/item/%E5%9B%BE%E5%83%8F%E4%BA%8C%E5%80%BC%E5%8C%96/_blank)点[灰度值](https://baike.baidu.com/item/%E7%81%B0%E5%BA%A6%E5%80%BC/10259111" \t "https://baike.baidu.com/item/%E5%9B%BE%E5%83%8F%E4%BA%8C%E5%80%BC%E5%8C%96/_blank)设置为0或255，也就是将整个图像呈现出明显的黑白效果的过程。因此对于彩色图片必须先经过灰度化处理后才能进行二值化处理。

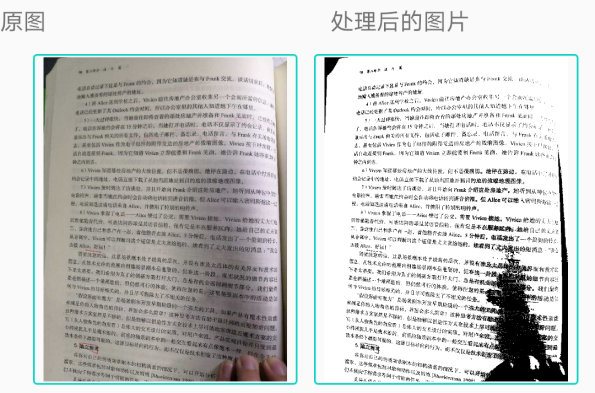
在开发过程中，我尝试了多种二值化方法，主要有人为阈值二值化，OTSU二值化方法等

对于人为阈值二值化的方法，需要人为的设置阈值，然后如果图像中的灰度值大于则将其设为255，如果灰度值小于则将其设为0。在开发过程中多次修改阈值，二值化效果变化如下:

阈值80: 阈值95：

阈值110：

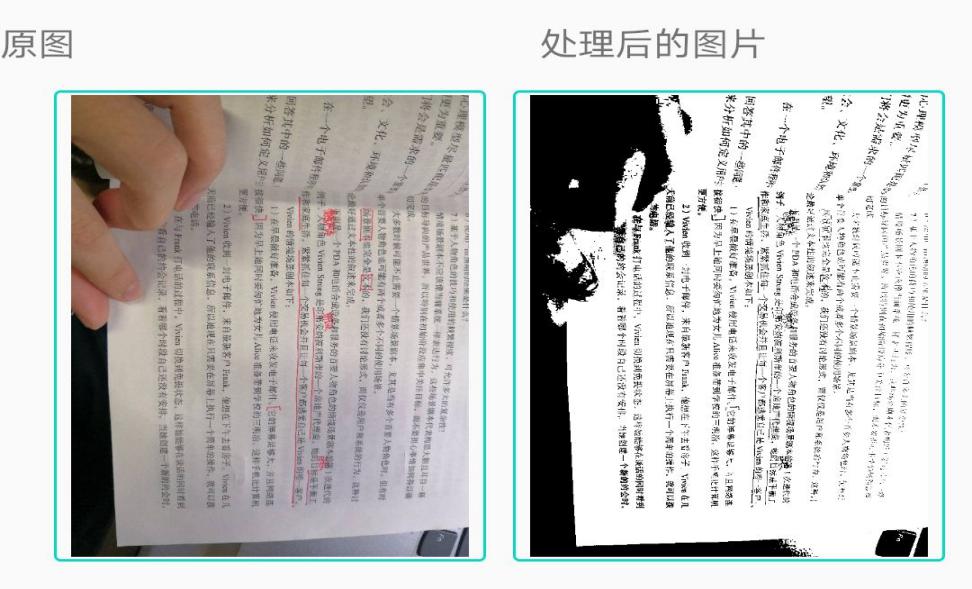


根据实验结果可知，不同的阈值对图片处理的影响巨大，如果阈值设置不当，则会出现大面积的黑色。OCR光学识别需要对图片的像素点进行精确识别，需要合适的预处理后的图片才能准确识别文本信息。不仅如此，OCR光学识别需要适应不同光照情况下的图片，因此人为设置阈值不可取，我最终选择了OTSU自适应二值化处理方法即根据输入图片的全局灰度值情况自动设置阈值，最终实现二值化的效果。

OTSU算法是一种图像灰度自适应的阈值分割算法。主要思想是按照图像上灰度值的分布,将图像分成背景和前景两部分看待,前景就是我们要按照阈值分割出来的部分。背景和前景的分界值就是我们通过算法根据全局像素值求出的阈值。

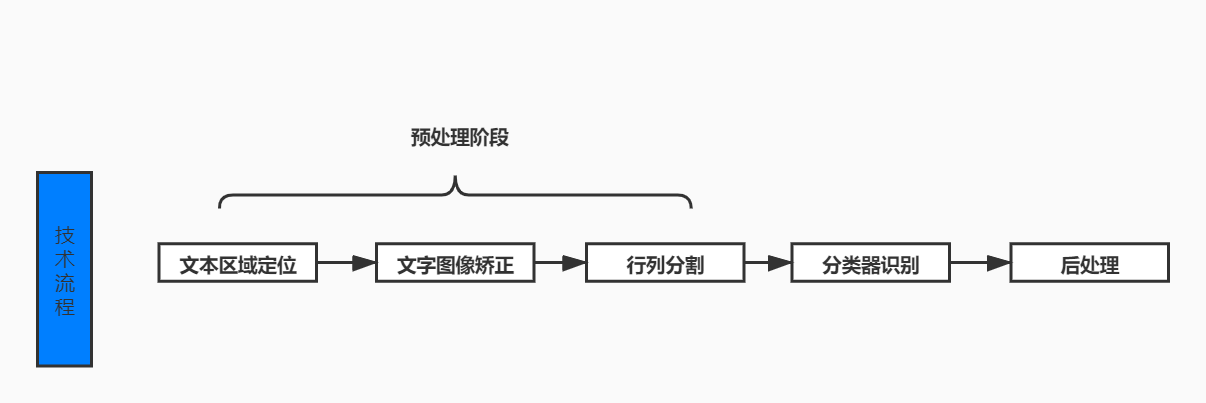
OTSU算法中先遍历不同的阈值，计算不同阈值下对应的背景和前景之间的类内方差,当类内方差取得极大值时，此时对应的阈值就是我们要设置的最终阈值。

OTSU二值化效果图:



**1.3传统OCR光学识别技术**

传统的OCR光学识别技术可以分为5个阶段如下图：



OCR光学识别技术的核心在于文本区域定位以及文本识别两个步骤。其中文字图像纠正就是在将图片输入检测之前首先会对原图进行一次图片纠正操作,确保图片文字是水平方向，提高文本检测的准确性。图像纠正的方法一般有两种:水平纠正、透视纠正。

行列分割必须是在文字图像纠正的基础上才能进行有效的分割。传统的行列分割的方法为：先对图片进行水平投影，找到每一行的上界限和下界限，然后再进行行切割---对切割出来的每一行，进行垂直投影，找到每一个字符的左右边界，从而实现单个字符的分割。

**1.4深度学习的OCR光学识别技术**

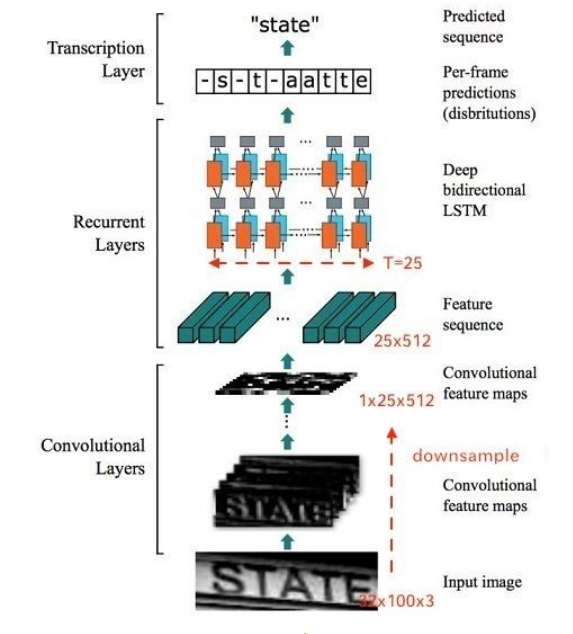
### 1.4.1 CTPN网络文本区域检测概述

OCR光学识别的前提必须要先定位到文本区域，即文本检测。CTPN网络是一个结合了CNN卷积神经网络和LSTM长短期记忆网络的文本检测算法，能有效的检测出复杂场景的横向分布的文字。

### 1.4.2CRNN网络文本信息识别概述

文本识别即对定位好的文字区域进行识别，主要解决的问题是每个文字是什么，将图像中的文字区域进转化为字符信息。

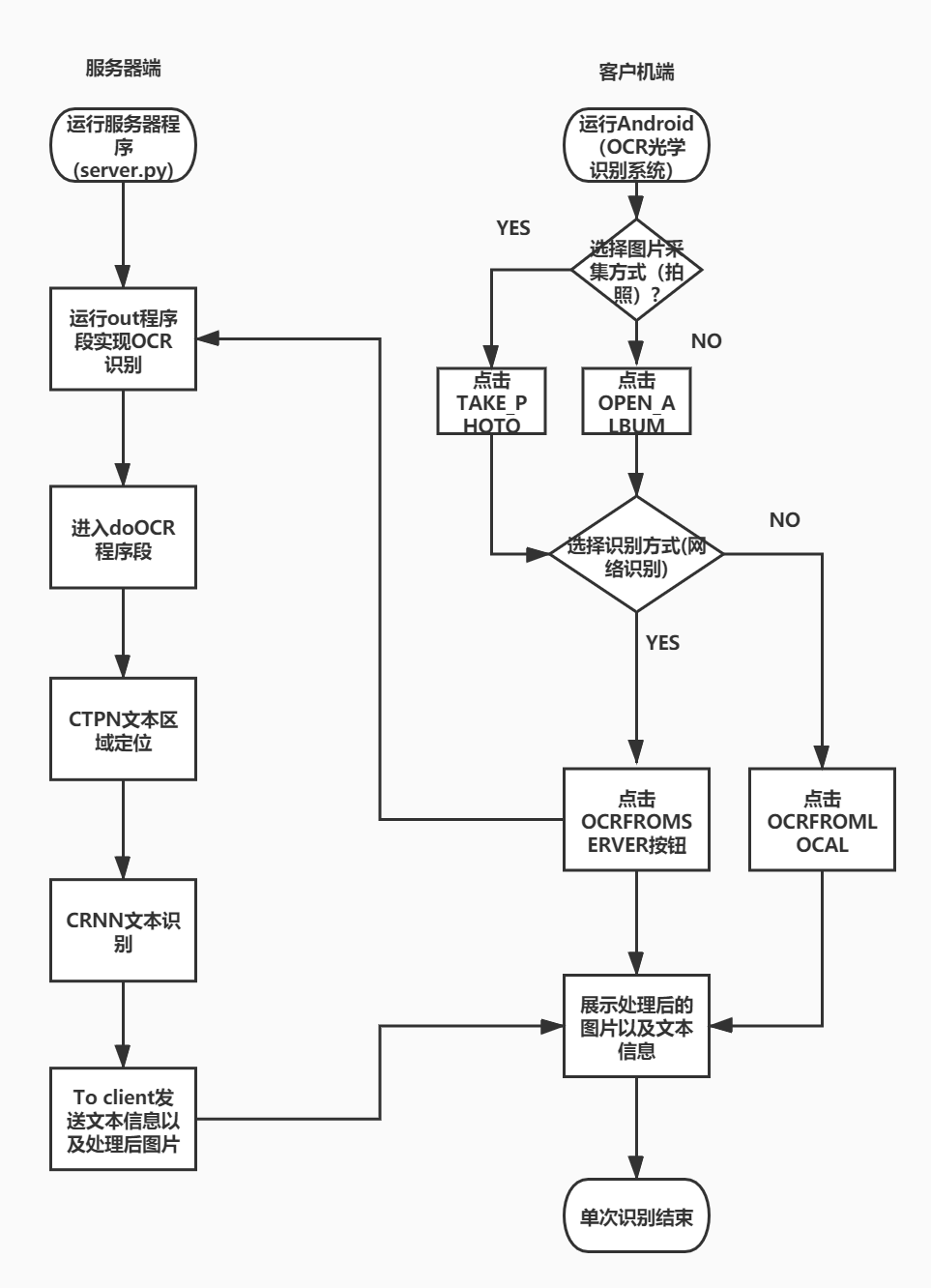
该CRNN网络是一种CNN卷积神经网络和深层双向LSTM网络相结合的文本识别网络,主要有三个部分组成的：卷积层、循环层、转录层。这三个部分的作用为：1、卷积层，对图像进行卷积操作提取出图像的深层特征；2、循环层，产生特征序列每一帧的预测值；3、转录层，将循环层生成的帧预测序列转换为标签序列。如下图。



# 基于Android移动端的OCR光学识别系统

## 2.1移动端OCR光学识别系统流程图以及架构图

该OCR光学识别系统采用了C/S客户机-服务器模型，客户机主要功能为：1、图片信息的采集;2、展示处理后的图片以及识别文本信息;3、本地OCR光学识别。服务器的主要功能：1、在线的基于深度学习网络的OCR光学识别;2、传递处理后的图片以及识别的文本信息其系统的流程图如下：



## 2.2光学识别系统的客户端

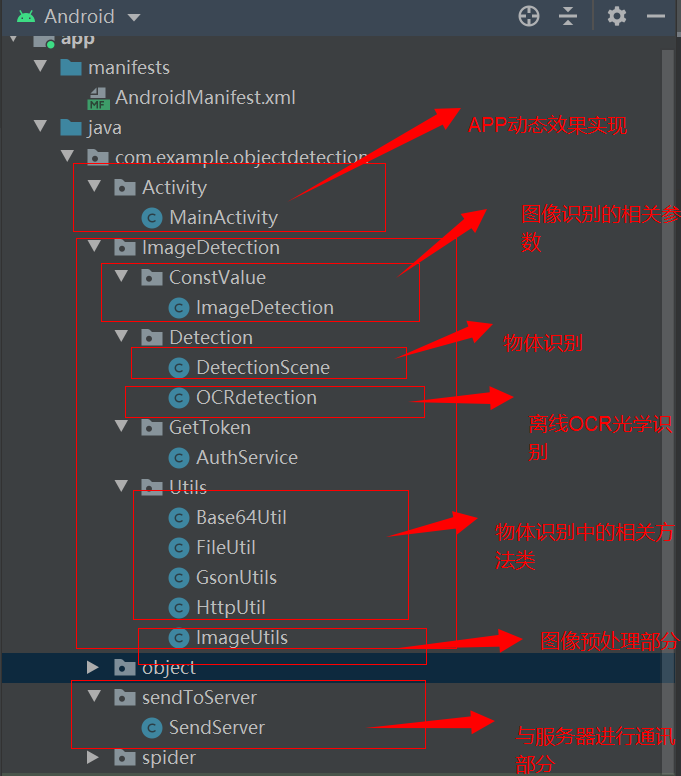
对于该系统的客户机的程序文件结构，主要有Activity包、ImageDetection包、sendToServer包。

Activity包:主要实现的是APP页面动态效果的实现，比如：拍照、相册选择图片、展示识别的文本信息以及处理后的图片等。

ImageDetection包:主要实现的是本地端图片预处理以及离线的OCR识别,以及可以对图片中的主要物体进行识别。

sendToServer包：主要实现的是客户机与服务器之间的信息通讯，比如图像交换，文本信息交换等。

程序文件结构图:



### 2.2.1图像输入模块

在移动端的图像输入主要有两种方式：一种是通过相册选择图片，另一种是拍照。两种方式，可以增加系统的实用性和适用性以及在一定程度上提高系统的选择性以及灵活性。

该模块的实现主要在Activity.MainActivity文件下:



2.2.2图像预处理模块

在移动端的图像预处理模块主要是为离线OCR光学识别服务的，主要有OTSU阈值二值化，灰度化，对图片的裁剪降低分辨率以及旋转等。

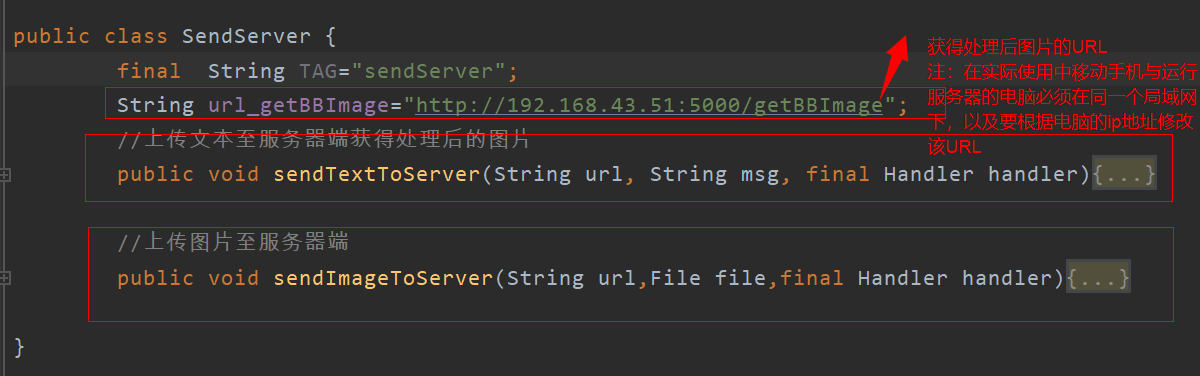
该模块的实现主要在ImageDetection.Utils.ImageUtils下



2.2.3图像发送至服务器模块

在移动端处与服务器交互的方法则是利用okhttps3的http库实现http交互,实现了上传图片、下载图片、文本信息交换等。

该模块的实现主要在sendToServer.SendServer下



2.2.4移动端离线光学识别模块

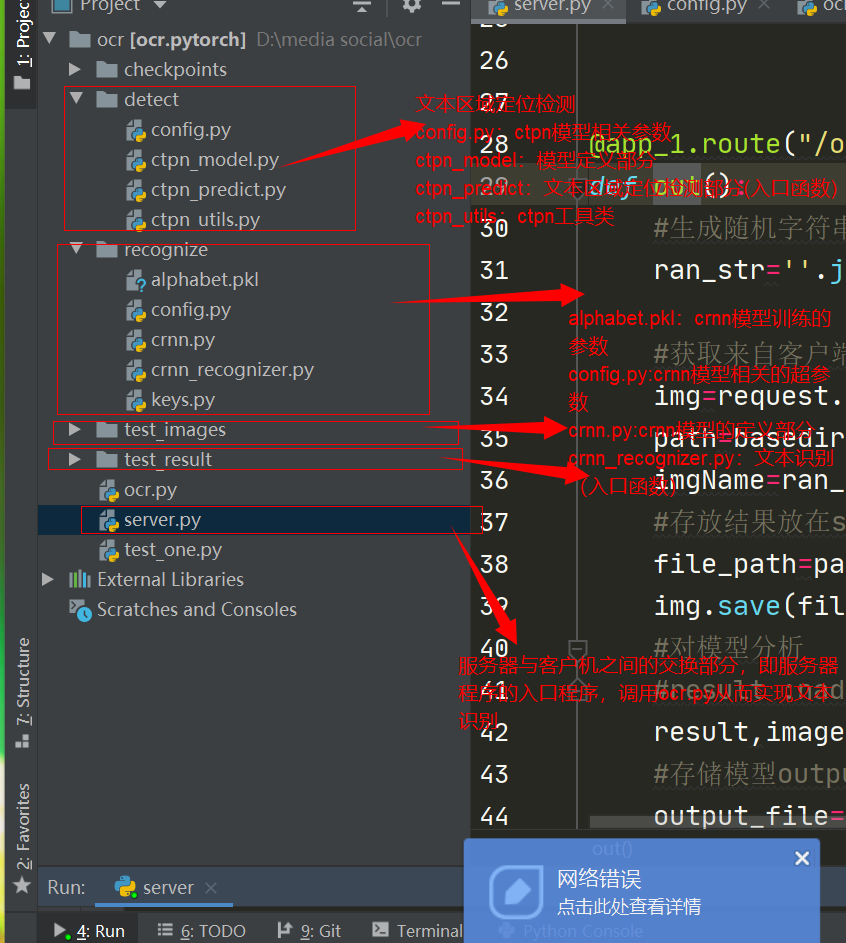
移动端的离线光学识别采用的基于Tesseract-ocr光学引擎下的识别方法。首先对图片进行裁剪（减低分辨率，弥补移动手机处理性能的缺陷），以及旋转（图像文本矫正）,利用Tesseract-ocr光学引擎的分类器进行分类。

## 2.3光学识别系统的服务器端

该系统的服务器端主要功能其一:与客户机进行信息通讯，其二：弥补移动手机处理性能的缺陷，在线进行图像处理以及文本识别等。

其文件结构图如下:

注：以下图片中test\_images文件夹存放的是客户机上传的原图，test\_result文件夹存放的是经过网络处理后的图片(标注过回归框和置信度的图片)



对于服务器端的OCR光学识别是利用CTPN网络以及CRNN网络进行识别，其网络之间的衔接部分：ocr.py文件下



注：回归框的信息为五元组(left,top,width,height,置信度)

### 2.3.1文本区域检测模块

主要利用的CTPN连接文本提议网络,实现文本区域定位的功能.

该模块的主要实现在detect文件夹下，网络定义部分在ctpn\_model.py下：

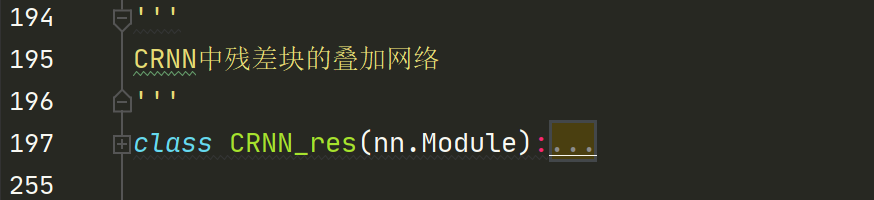


### 2.3.2文本识别模块

该模块接收上游CTPN网络的输出，进行图片文本识别。

该模块的主要实现是在recognize文件夹下的crnn.py文件:





# OCR光学识别系统效果展示

## 3.1基于Tesseract离线OCR光学识别

注解：OCRFROMSERVER按钮为网络识别，OCRFROMLOCAL按钮为离线识别。

1：包含数字、汉字、标点符号的短文本

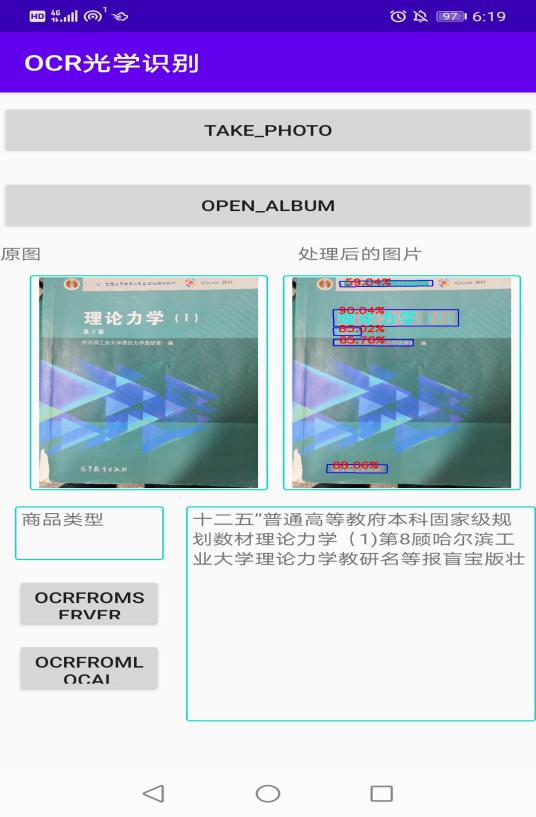


## 3.2基于CTPN区域检测、CRNN文本识别的OCR光学识别

1：数字、汉字、标点符号短文本 2.汉字、标点符号的长文本

3:书面（复杂环境下的OCR官学识别） 4:手写报告识别

# 开发过程中的问题及解决方案

## 4.1问题一

Android开发过程中调用系统相机的回调函数onActivityResult(int requestCode,int resultCode,Intent data)代码过程中得到的总是null即app上可以跳转到系统相册但是不能返回选中的图片。

原因分析：

默认情况下,intent.putExtra(MediaStore.EXTRA\_OUPUT,uri);不需要使用。照相机有自己默认的存储路径，拍摄的照片将返回一个缩略图如果想访问原始图片，可以通过data.extra能够得到原始图片位置。即，如果指定了目标uri，data就没有数据，如果没有指定uri，则data就返回有数据！

## 4.2问题二

对于离线端的OCR光学识别，识别10个汉字都需要十几秒，识别速度非常慢。

原因分析:

由于拍照或者系统相册中选中的图片默认为原图，分辨率非常高，且移动手机的性能有不足，因此离线的OCR光学识别比较慢。

解决方案:

首先对图片进行预处理，进行裁剪->灰度化->二值化.在这后两个处理过程中并不是采用按像素点处理而是类似于pooling的过程按区域进行处理，最终不仅达到二值化的效果也达到降采样的目的。

# 5.总结

本次的课程设计对我来说，受益匪浅。这一次的课程设计，我实现了C/S客户机与服务器模型的系统，前端使用了java语言编写Android app，后端则是用python搭建了一个服务器，完成了离线以及网络OCR光学识别的功能。

在这个过程中，我逐渐熟悉了Android app的开发以及python语言的编写。这次的课程设计可以说是上个学期计算机网络、这个学期机器学习、媒体计算基础等相关知识转化为实践的过程。我不仅亲手搭建了客户机与服务器之间的通讯系统，而且通过实践加深了对深度学习相关知识的理解。