目　　录

[第一章 绪论 1](#_Toc452476009)

[1.1 课题背景与意义 1](#_Toc452476010)

[1.1.1 课题背景 1](#_Toc452476011)

[1.1.2 课题意义 1](#_Toc452476012)

[1.2 课题任务及目标 2](#_Toc452476013)

[1.2.1 课题任务 2](#_Toc452476014)

[1.2.2 课题目标 2](#_Toc452476015)

[第二章 技术调研 3](#_Toc452476016)

[2.1 计算机视觉 3](#_Toc452476017)

[2.1.1 计算机视觉简介 3](#_Toc452476018)

[2.1.2 图像特征分析 3](#_Toc452476019)

[2.1.3 HSV 4](#_Toc452476020)

[2.1.4 LBP 4](#_Toc452476021)

[2.1.5 Dark Channel 4](#_Toc452476022)

[2.2 机器学习 4](#_Toc452476023)

[2.2.1 SVM 5](#_Toc452476024)

[2.2.2 LogisticRegression 5](#_Toc452476025)

[2.2.3 kNN 6](#_Toc452476026)

[2.2.4 Early Fusion 7](#_Toc452476027)

[2.3 图像去雾 7](#_Toc452476028)

[2.3.1 暗通道先验假设理论 7](#_Toc452476029)

[2.3.2 特征利用 7](#_Toc452476030)

[第三章 算法设计 8](#_Toc452476031)

[3.1 图像特征提取算法 8](#_Toc452476032)

[3.1.1 特征存储 8](#_Toc452476033)

[3.1.2 HSV 8](#_Toc452476034)

[3.1.3 LBP 8](#_Toc452476035)

[3.1.4 Dark Channel 8](#_Toc452476036)

[3.2 机器学习算法 8](#_Toc452476037)

[3.2.1 LogisticRegression 8](#_Toc452476038)

[3.2.2 kNN 9](#_Toc452476039)

[3.2.3 SVM 10](#_Toc452476040)

[3.2.4 Libsvm库 11](#_Toc452476041)

[第四章 系统实现 12](#_Toc452476042)

[4.1 模块设计 12](#_Toc452476043)

[4.1.1 图像特征提取 13](#_Toc452476044)

[4.1.2 原始数据处理 15](#_Toc452476045)

[4.1.3 模型训练 15](#_Toc452476046)

[4.1.4 交叉验证 15](#_Toc452476047)

[4.1.5 模型融合 16](#_Toc452476048)

[4.2 图像特征提取 16](#_Toc452476049)

[4.2.1 HSV 17](#_Toc452476050)

[4.2.2 LBP 17](#_Toc452476051)

[4.2.3 Dark Channel 18](#_Toc452476052)

[4.3 原始数据处理 18](#_Toc452476053)

[4.3.1 格式统一 18](#_Toc452476054)

[4.3.2 归一化处理 18](#_Toc452476055)

[4.3.3 数据分块 19](#_Toc452476056)

[4.4 模型训练 20](#_Toc452476057)

[4.4.1 针对特征训练 20](#_Toc452476058)

[4.4.2 模型融合 21](#_Toc452476059)

[4.5 交叉验证 21](#_Toc452476060)

[4.5.1 分块 21](#_Toc452476061)

[4.5.2 测试 22](#_Toc452476062)

[4.6 模型融合 22](#_Toc452476063)

[4.6.1 融合方法设计 22](#_Toc452476064)

[第五章 实验数据分析 23](#_Toc452476065)

[5.1 数据集分析 23](#_Toc452476066)

[5.1.1 训练集分析 23](#_Toc452476067)

[5.1.2 测试集分析 23](#_Toc452476068)

[5.1.3 LBP 23](#_Toc452476069)

[5.2 训练方法分析 23](#_Toc452476070)

[5.2.1 核函数选择 23](#_Toc452476071)

[5.2.2 参数调整 23](#_Toc452476072)

[5.3 融合方法分析 23](#_Toc452476073)

[第六章 总结 24](#_Toc452476074)

[6.1 成果小结 24](#_Toc452476075)

[6.2 课题展望 24](#_Toc452476076)

[参考文献 26](#_Toc452476077)

[致　　谢 27](#_Toc452476078)

1. 绪论
   1. 课题背景与意义
      1. 课题背景

目前，空气污染问题逐渐成为大家广泛关注的问题，空气质量的检测非常重要。我国部分城市的空气状况不容乐观，以北京为例，根据北京环保局公布的最新环境状况公报，全市空气中细颗粒物PM2.5年平均浓度值为85.9微克/立方米，超过国家标准1.45倍；二氧化氮年平均浓度为56.7微克/立方米， 超过国家标准42%；可吸入颗粒物PM10年平均浓度值超过国家标准65%。

空气质量与人们的健康密切相关，所以对于空气质量的检测与研究是非常必要的， 目前主流的空气质量检测手段主要依赖一些专业的设备，但缺陷是成本极高，而且难以普及。 本课题目的就是将是计算机视觉与机器学习算法相结合，设计一个专家系统，通过照片来对空气状况进行估计，解决专业测量设备成本高，难以普及的缺陷，使得每个人都能够随时地掌握自己周围的空气质量状况， 保护自己的健康。

计算机视觉是一个研究如何让计算机能够像人类一样“看”的学科，具体地来讲，就是让计算机能够识别图像，视频中的视觉特征，并从中提取我们所需要的信息。计算机视觉发展到今天，已经不单单是一个有关图像特征的学科，它常常与模式识别、机器学习、图像理解等人工智能方面的学科综合研究。传统的计算机视觉通常是对图像矩阵的运算，对目标进行检测，跟踪等工作，随着人工智能技术的蓬勃发展，如今计算机视觉不仅仅能够提取图像特征，跟踪检测目标，还能够辨别图像中的物体，甚至于“理解”图片的意义。

机器学习是近些年来迅猛发展的一个学科，可以说目前已经广泛地应用于我们的生活中，我们每天浏览的个性化新闻，购物网站上的商品推荐，短信、邮件的垃圾过滤，手机输入法里的手写识别，都是机器学习技术的产物。深度学习也是机器学习中一个重要的研究方向，目前已经取得了非常令人惊叹的进步。2016年3月，Google利用基于机器学习的围棋AI - AlphaGo，成功击败了世界冠军李世乭，打破了围棋“人类智慧最后骄傲”的神话，而这一标志性的成果，正是依靠深度学习算法来实现的。可以预见，深度学习算法将在未来极大地改变人们的生活

* + 1. 课题意义

首先，课题的目标是设计一套基于计算机视觉的空气质量检测系统，传统的空气质量检测方法是通过专业的仪器采集空气，对其质量进行一个准确的判断，优点是准确率高，数据来源于采集样本的空气成分分析，结果最为真实可信。但是缺点也有很多，一是成本太高，二是移动不方便，难以随身携带， 三是操作需要专业知识，对于普通人而言难以普及。在日常生活中，就普通人的真实需求来看，人们需要的是这样一个空气质量检测系统：

1. 实时实地：通常官方发布的数据只有某一时段，某个地点的空气质量数据，这个数据并不能代表整个城市每个地点的空气质量。因此人们无法获取当时所处位置相对准确的空气质量信息
2. 便携性：专业的空气质量检测设备体积巨大，而本课题所需要的仅仅是一张照片，而拍摄照片的可以是身边的各种移动设备。
3. 成本：大部分非专业人员很难在空气质量的检测上投入很多的资金，并且也没有必要为了获取控制质量就购买一台专业的设备。而基于计算机视觉的检测技术只需要拍一张照片，几乎是零成本
4. 准确性：一方面，生活中对于空气质量监测的准确性要求是不高的，人们只需要知道今天的空气质量好或者坏，好到了何种地步或者坏到了何种地步，是否适宜运动，是否需要戴口罩等等， 所以最终监测的结果不一定是具体的数值，可以是分类。而另一方面，基于机器学习构建的专家系统，事实证明在诸多领域都有着很高的的准确度，所以一个设计良好的系统是完全可以满足需要的。

其次，图像去雾，机器学习等概念目前都很热门，本课题结合这两种技术，对使用机器学习识别雾霾天气的技术进行了积极的探索。

* 1. 课题任务及目标
     1. 课题任务

1. 研究基于视觉特征的PM2.5空气污染自动检测算法。

2. 图像视觉特征提取

3. 运用机器学习中的有监督学习算法，对不同区域的PM2.5进行分类(优，良，轻度污染等)

4. 设计一个基于视觉特征的PM2.5空气污染自动检测原型系统

1. 技术调研
   1. 计算机视觉
      1. 计算机视觉简介

计算机视觉是一个提取，处理，分析，理解图像的学科，它综合了图像处理，模式识别，机器学习，甚至于物理学，生物学，目标就是要使计算机具有想人类一样“看”的能力。一个完整的计算机视觉系统通常有以下组成部分：

1. 过程控制
2. 事件检测
3. 信息组织
4. 物体与环境建模
5. 交感互动

目前计算机视觉经常与模式识别，机器学习相联系，与人工智能相关的研究进行结合，用于理解图片内容，构建专家系统等。现在这些技术都有了实际的应用，例如下面这些场景：医学中利用计算机视觉对肿瘤的x光片进行识别；让计算机根据图片造句；利用照片搜索商品。

计算机视觉需要解决的经典问题如下：、

* 识别
  + 鉴别
  + 监测
  + 图像内容提取
  + 姿态评估
  + 光学字符识别
* 运动
  + 自体运动
  + 图像跟踪
* 场景重建
* 图像恢复
  + 1. 图像特征分析

常见的彩色图像格式有jpg，bmp等格式，图片可以解析为一个三维矩阵，两个维度表示像素位置，第三个维度有三个分量，分别是R,G,B，表示红，绿，蓝三个值，红绿蓝三原色的叠加即可显示缤纷的彩色。从图像的原始矩阵中，我们可以提取出图像的相关特征，包括颜色特征，纹理特征等等。

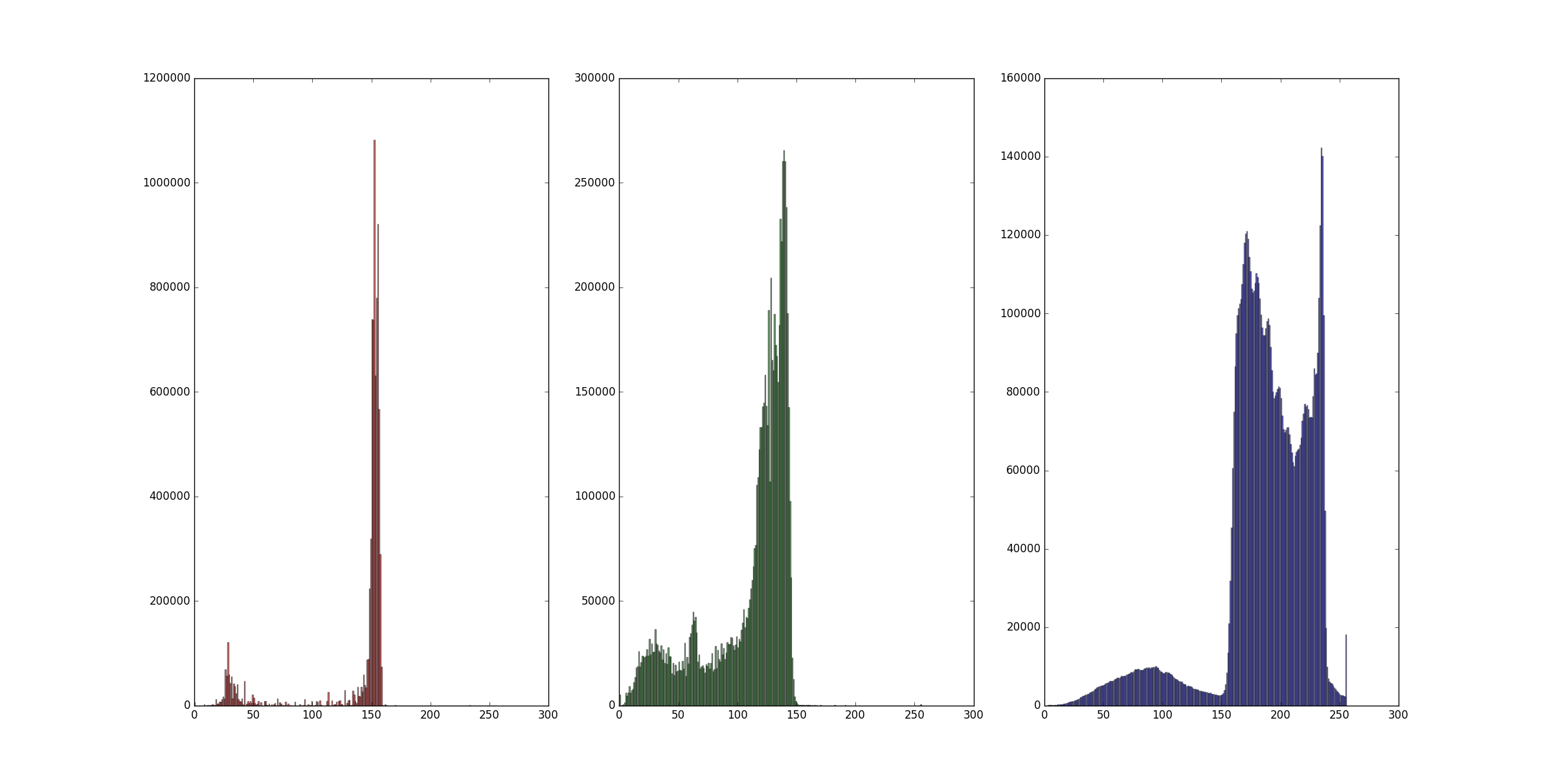
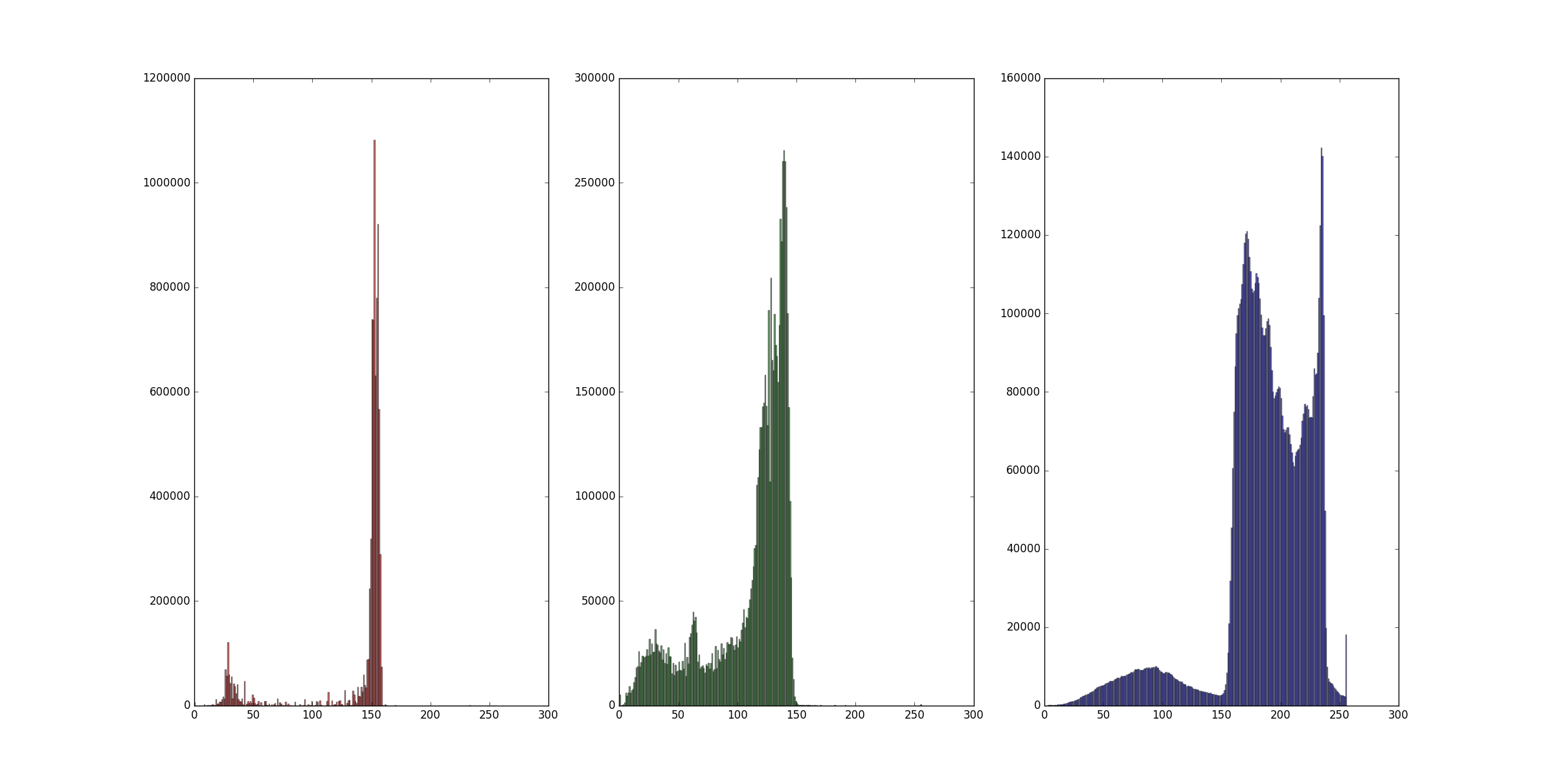
* + 1. HSV

H, S, V 即色亮，饱和度， 明度 (Hue, Saturation, Value), 与之相似的还有HSL，即色相，饱和度， 亮度(Hue, Saturation, Value), 也称为HSB， B为Brightness。HSV与HSL其实本质上都是将RGB转换到圆柱坐标系中表示的一种方法，相比于RGB，HSB的几何结构更加直观一些，表示的意义也比较明确。

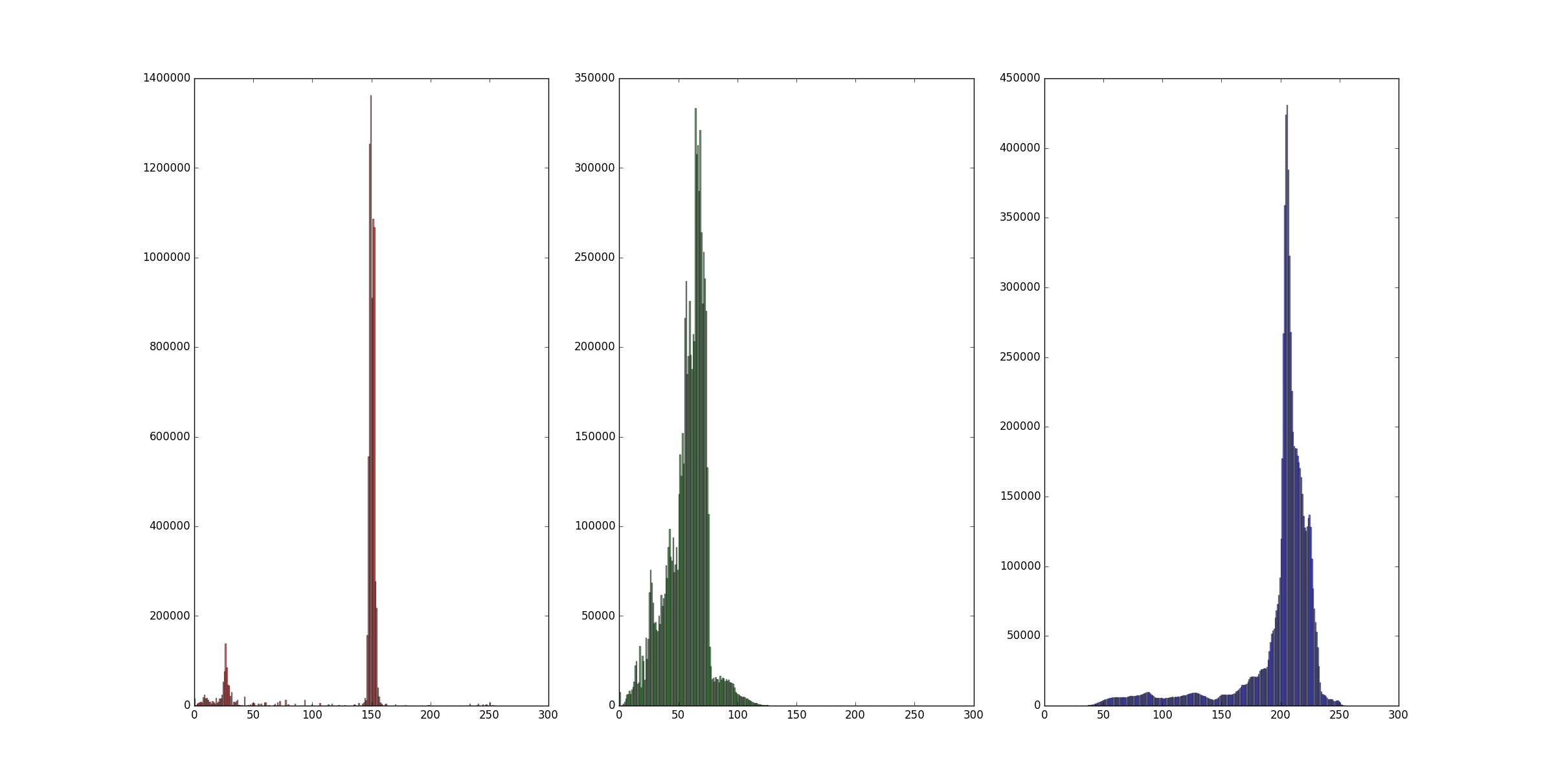
就本课题而言，HSV三个变量的分布在不同天气中的是有一定规律的。例如，天气晴朗时， 天空会更加蓝，物体颜色的饱和度会更高，亮度也会有所提高，这些推断来源于生活经验，不能直接作为课题的前提条件，所以我们需要对数据做出一定的处理， 用数据来验证猜想的正确性

我在采集到的数据集中进行了采样，统计得到了hsv的图表，以下是一些示例

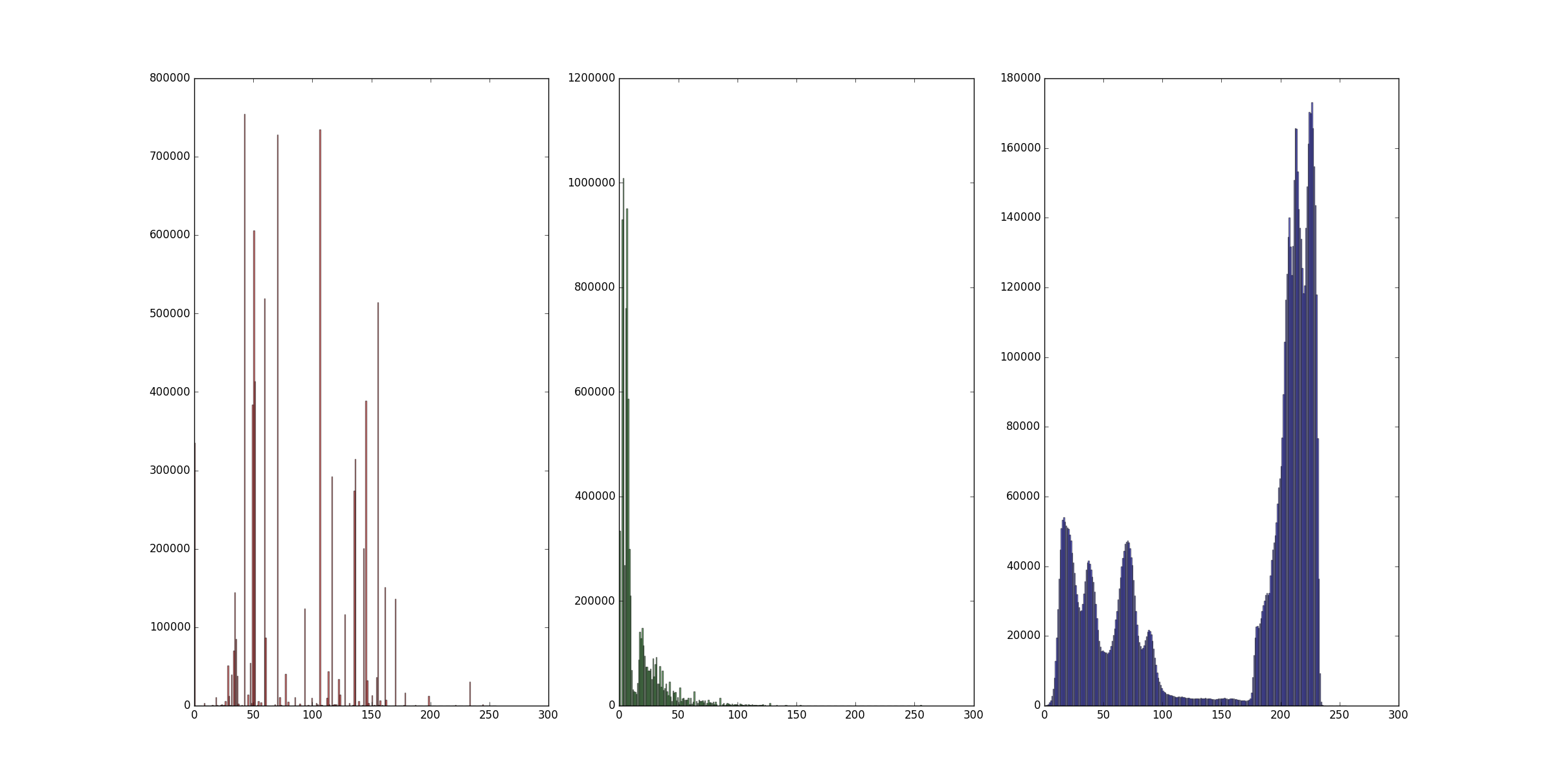
空气质量较好：



空气质量一般：



空气质量较差：



从上面的分析可以看出HSV与天气存在一定的关系，我们可以将HSV作为一个特征来处理

* + 1. LBP

LBP（Local Binary Pattern，局部二值模式）是一种用来描述图像局部纹理特征的算子，它反映了每个像素点与周围像素点的关系；是首先由T. Ojala, M.Pietikäinen,和 D. Harwood 在1994年提出的，后来经过不断地改进和优化，又提出了LBP旋转不变模式，LBP均匀模式等等

LBP具有以下优点：

* 旋转不变性
* 灰度不变性

本项目实现了原始LBP， 均匀LBP和旋转不变LBP，它们各有优缺点，本项目最终采取的方案为性能和效果俱佳的均匀LBP。以下是对不同模式的简介，具体的原理详细参考本文算法设计部分的论述

原始LBP：信息量最大，较为完整地反映了图片的纹理特征，但缺点是维数过高，不利于后续的研究，而且由于具有旋转不变性，所以信息有冗余。

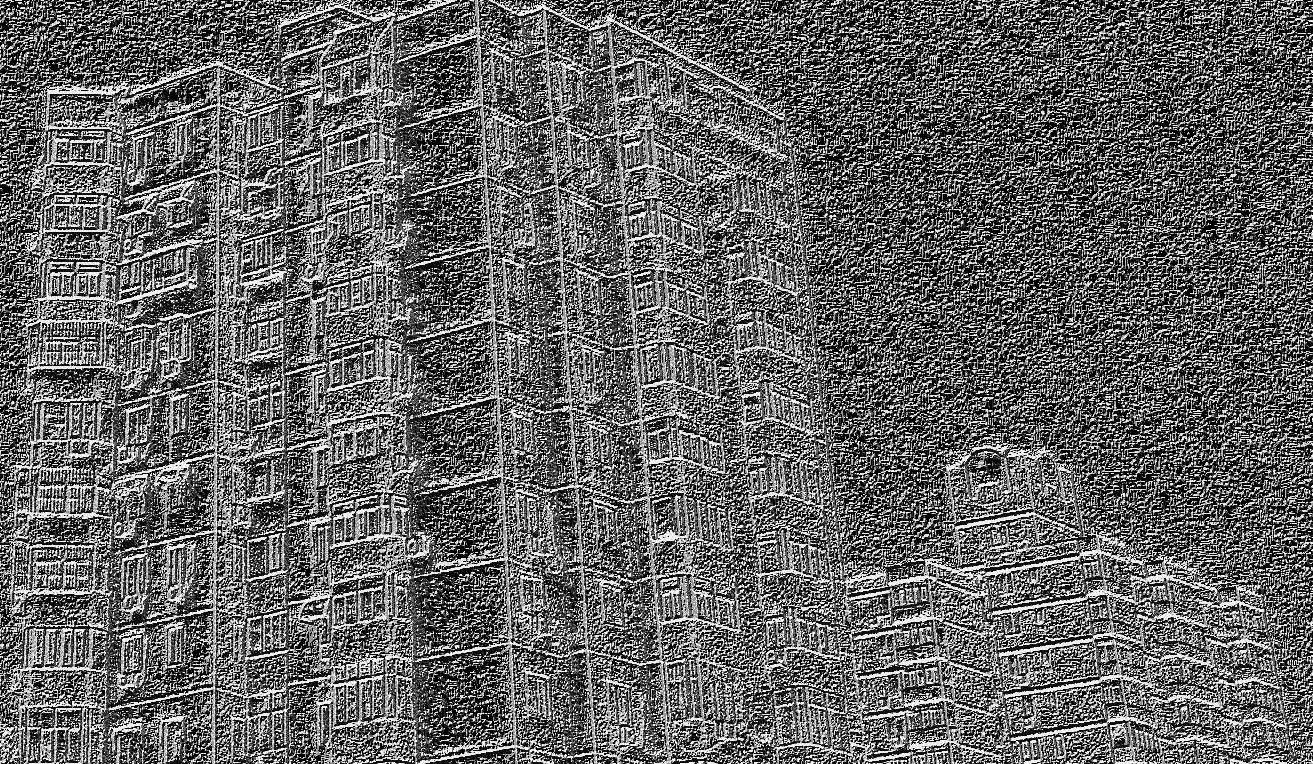
旋转不变LBP：在原始LBP的基础上，利用旋转不变特性，将256种LBP模式缩减为36种，优点是大大减少了维数，但缺点是效果一般。

均匀LBP模式： 为了综合解决以上两种LBP模式的问题， Ojala提出了采用一种“等价模式”（Uniform Pattern）来对LBP算子的模式种类进行降维。经Ojala的大量实验验证，在实际图像中，绝大多数LBP模式最多只包含两次从1到0或从0到1的跳变。“等价模式”定义为：当某个LBP所对应的循环二进制数从0到1或从1到0最多有两次跳变时，该LBP所对应的二进制就称为一个等价模式类。如10001111（先由1跳到0，再由0跳到1，共两次跳变），00000111（只含一次从0到1的跳变），00000000（0次跳变），都是等价模式类。除等价模式类外，其他的模式都属于另一类，称为混合模式类，例如10010111（共四次跳变）

均匀LBP将特征256维降至59维（59=(2+0+56)+1），一方面解决了原始LBP维度过高的问题，另一方面由于该理论基于大量的实验验证，包含了实际图片中的绝大部分模式，解决了旋转LBP效果不佳的问题

以下是提取出的原始LBP特征，可以看到LBP清晰地反映了图像的纹理特征：



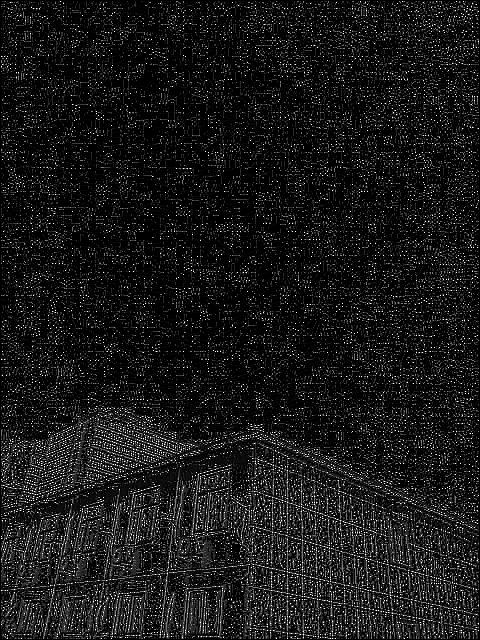
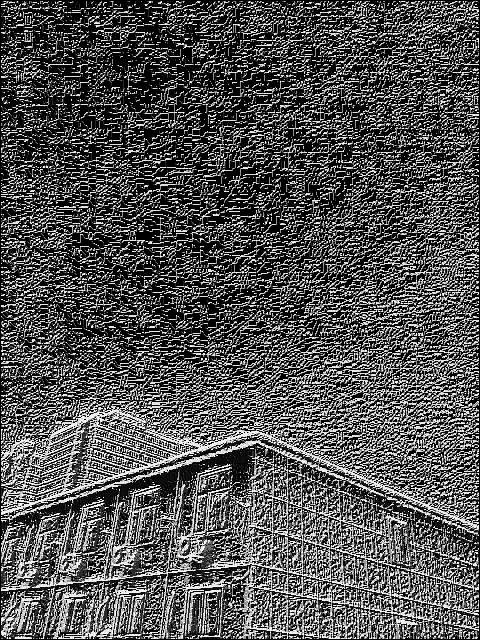
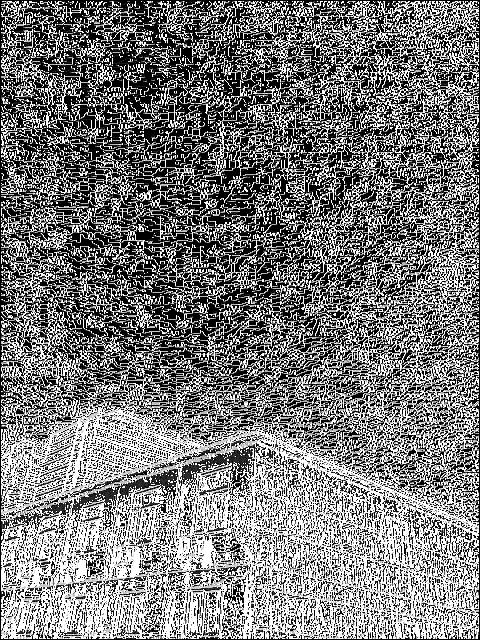


以下是三种LBP的对比，可以看到原始LBP效果最佳，均匀LBP效果与原始LBP相似，但旋转不变LBP效果就比较不明显

原图：



从左至右：原始LBP，旋转LBP， 均匀LBP



* 1. 机器学习

近些年来，人工智能领域蓬勃发展，机器学习算法已经广泛应用到各个领域，甚至可以说我们的日常生活都有机器学习算法的参与，想象一下这样的场景：清晨起床打开手机，播放器根据你的喜好推荐给你一首歌，给你带来一天的好心情；吃过早饭你开始查收邮件，邮件过滤系统帮助你自动识别垃圾邮件，让你远离广告的烦恼。这时候手机来了一条消息，你用手写输入法回复对方。到公司开始一天的工作，公司服务器上的数据已经自动分析完成，这些海量用户数据分析帮助你做出了正确的决策。当你回到家里，打开购物网站，系统自动根据你的浏览记录购买记录进行推荐。

以上的例子在生活中非常常见，而无论是音乐喜好，邮件过滤，数据分析，购物推荐这些看起来很智能的应用，都是通过机器学习算法来实现的。

机器学习分为监督学习和无监督学习，监督学习是在已知目标变量分类信息的基础上，进行最优的拟合。机器学习中的分类和回归都属于监督学习。

与监督学习相对应的是无监督学习，无监督学习中没有类别信息，也没有给定的目标值， 数据集合通过类似程度组合成多个类，这个过程被称作聚类，在这个过程中寻找数据统计值的过程称为密度估计。

本课题中所用到的机器学习算法属于监督学习，即在已知图片拍摄时空气质量的情况下进行训练，得到的模型用于预测空气质量。具体用到的算法如下。

* + 1. SVM
    2. LogisticRegression
    3. kNN
    4. Early Fusion
  1. 图像去雾
     1. 暗通道先验假设理论

在图像去雾有一篇非常著名的文章，何凯明博士的Single Image Haze Removal Using Dark Channel Prior，这篇文章阐述了利用暗通道去雾的原理，获得了2009年的CVPR最佳论文。本项目就采用暗通道作为识别雾霾天气的视觉特征之一

暗通道是图像的三个通道中值最低的那一个，即R,G,B三个分量中取最小值。暗通道先验理论是指，在非天空的局部区域里，某一些像素总会有至少一个颜色通道具有很低的值。也就是说该区域的光强度的最小值是个很小的数。

这个假设看起来复杂，但其实很好理解，在自然界中，没有雾霾的情况下，可以说自然界的景物要么是鲜艳的彩色，例如彩色物体的表面，要么是阴影或者比较暗的颜色，例如阴影，深色的物体。色彩鲜艳的物体通常rgb值相差通常很大，某个值很高，另一个值很低，而深色的物体整体颜色比较暗，所以rgb值也会比较低，它们共同的特点就是暗原色的通道值很低，

* + 1. 特征利用

概括来说NDK主要有以下三种用途：

* 保护代码，在apk中java层代码被反编译是很容易的，然后而反编译C/C++库难度较大。
* 因为C/C++为大部分的开源库的编写语言，在NDK中可以顺利调用第三方C/C++库。
* 用C/C++写的库便于移植，可以方便在其他的嵌入式平台上多次使用。

1. 算法设计
   1. 图像特征提取算法
      1. 特征存储

* 功能简单实用，适合移动端设备。
* 采用高效的Twofish加密算法，提升用户体验。
* 使用范围广法，支持各类Android设备。
  + 1. HSV

本课题所研究的移动智能终端的个人信息保密系统旨在保护Android设备上的数据与隐私安全。由于现在大部分Android设备都配备了SD卡，并将SD卡作为存储与传递信息的媒介。因此，本课题所采取的信息保护手段为在Android设备上建立虚拟磁盘，以此在SD上生成新的磁盘文件夹，并在其中存放信息与数据。

通过该信息保密系统，用户可以直接在Android设备上管理文件、图片等各类信息与数据，最大限度的保证信息与数据的安全性和完整性。同时，加密过程也是对用户保持透明的，并不会因为加解密而降低用户体验。

* + 1. LBP

本信息保密系统只由一个本地客户端构成，不需要与服务器通信，且密钥存储与设备中。文件管理器功能通过调用Android自身的SQLite数据库实现，UI界面由各类图片与控件自行编写。

* + 1. Dark Channel

本信息保密系统可以在大部分Android 2.3的移动设备上，但该设备必须获得ROOT权限以进行磁盘操作。进行磁盘操作的时候需该设备连接SD卡方能生效。从理论上分析，所有装有本软件的设备都可以读取SD卡上的加密文件。

* 1. 机器学习算法
     1. LogisticRegression

通过利用内核的密码应用编程接口，dm –crypt完成密码操作。根据一般的情况，内核通常以模块的形式加载将各种加密程序。而对于256-bit 的Twofish来说，已具备很高的安全强度，足够用来保护绝密级的数据。因此，本课题中我们使用了256-bit Twofish密码，利用下列命令，检查保证内核已经将Twofish密码模块进行加载，：

$ cat /proc/crypto

出现下列语句，说明已将Twofish模块加载：

name : twofish

module : twofish

type : cipher

blocksize : 16

min keysize : 16

max keysize : 32

若非如此，我们手动加载Twofish模块，modprobe命令如下所示：

$ sudo modprobe aes

接下来进行dmsetup软件包的安装，配置device-mapper所需的工具包含在其中：

$ sudo apt-get install dmsetup cryptsetup该语句检查dmsetup软件包是否已经建立了设备映象程序，接下来输入下列命令：

$ ls -l /dev/mapper/control

接下来为加载dm-crypt的内核模块：

$ sudo modprobe dm-cryptdm-crypt

加载完成后，通过调用evice-mapper自动注册。如果再次进行检验，device-mapper已能识别dm-crypt，并且自动把crypt 添加为可用的对象：

$ sudo dmsetup targets

当所有步骤完成，crypt会输出下列语句：

crypt v1.1.0

striped v1.0.2

linear v1.0.1

error v1.0.1

这种情况，说明系统已经做好了装载加密设备准备。

此过程在Linux（如ubuntu）下编译运行cryptsetup时有必要进行，而在Android系统安装时就准备好需要的包，并在电脑端通过使用NDK直接交叉编译，从而生成cryptsetup可执行文件。下面建立一个加密设备[8]。

* + 1. kNN

有两种选择来创建作为加密设备装载的文件系统：一是使用物理设备；二是建立一个磁盘映像，然后作为回送设备加载。除了建立和捆绑回送设备外，其它操作过程无论那种情况都是相似的。

1. 建立回送磁盘映象

作为用来加密的物理设备的替换，可以通过使用命令dd来建立一个空磁盘映象，并将该映象作为回送设备来装载。下面以实例来加以介绍：

$ dd if=/dev/zero of=~/secret.img bs=1M count=80

这里我们新建了一个大小为80MB、映象名字为secret.img的磁盘映象。可以通过改变count的值来改变大小，。

接下来，将该映象和一个回送设备利用losetup命令联系起来：

$ sudo losetup /dev/loop0 ~/secret.img

现在得到了一个虚拟的块设备，其位于/dev/loop0。

2. 设置块设备

准备好了物理块设备，抑或是虚拟块设备（如前文所述，利用device-mapper将其作为加密的逻辑卷加载，并建立了回送映象），就可以进行块设备的配置。

使用cryptsetup建立逻辑卷，并将块设备与其捆绑：

$ sudo cryptsetup -y create myEncryptedFilesystem /dev/DEVICENAME

其中，myEncryptedFilesystem 为新建的逻辑卷的名称。并、且最后一个参数必须是将用作加密卷的块设备。所以，如果虚拟块设备要使用前面建立的回送映象的话，应当运行以下命令：

$ sudo cryptsetup -y create myEncryptedFilesystem /dev/loop0

无论是使用虚拟块设备还是物理块设备，要求输入逻辑卷的口令是程序必需的，-y的作用在于要求输入两次口令以确保无误。

使用下列命令进行检查以确认逻辑卷是否已经建立：

$ sudo dmsetup ls只要该命令列出了逻辑卷，就说明逻辑卷已经成功建立。设备号根据机器可能有所不同：

myEncryptedFilesystem (221, 0)device-mapper会将它的虚拟设备装载到/dev/mapper下面，所以，虚拟块设备应该为/dev/mapper/myEncryptedFilesystem ，它却经过透明加密的，尽管用起来它和其它块设备效果类似。

我们可以在虚拟设备上如同物理设备一样创建文件系统：

$ sudo mkfs.ext3 /dev/mapper/myEncryptedFilesystem

为新的虚拟块设备建立一个装载点并进行装载。命令如下所示：

$ sudo mkdir /mnt/myEncryptedFilesystem

$ sudo mount /dev/mapper/myEncryptedFilesystem /mnt/myEncryptedFilesystem

利用如下的命令查看其装载后的情况：

$ df -h /mnt/myEncryptedFilesystem Filesystem Size Used Avail Use% Mounted on /dev/mapper/myEncryptedFilesystem 97M 2.1M 90M 2% /mnt/myEncryptedFilesystem

尽管我们看到装载的文件系统看起来与其它文件系统无异，但实际上写到/mnt/myEncryptedFilesystem/下的所有数据，都是经过透明的加密处理后才写入磁盘的，因此，所有数据都是已密文格式读出。

* + 1. SVM

和平常的方法一样即可卸载该设备：

$ sudo umount /mnt/myEncryptedFilesystem

在dm-crypt中仍然视为一个虚拟设备，即便已卸载了块设备。命令 sudo dmsetup ls可以用以验证，该设备依然会被看到列出。正因为dm-crypt缓存了口令，机器上的其他用户能重新装载该设备即便不知道口令。

必须在卸载设备后从dm-crypt中显式的删除该设备以避免这种情况的发生。命令具体如下：

$ sudo cryptsetup remove myEncryptedFilesystem

此后，快设备将完全清除，必须再次输入口令以再次装载。

* + 1. Libsvm库

在卸载加密设备后，非常可能还需作为普通用户来装载他们，可以通过完成dm-crypt设备的创建和卷的装载工作，方法是将/dev /DEVICENAME用实际设备的名称或文件路径来替换：

cryptsetup create myEncryptedFilesystem /dev/DEVICENAME

mount /dev/mapper/myEncryptedFilesystem /mnt/myEncryptedFilesystem

如果使用的是回送设备的话，还能利用脚本来捆绑设备：

losetup /dev/loop0 ~/secret.img

cryptsetup create myEncryptedFilesystem /dev/loop0

mount /dev/mapper/myEncryptedFilesystem /mnt/myEncryptedFilesystem

如果收到消息 “ioctl: LOOP\_SET\_FD: Device or resource busy”，说明回送设备仍然装载在系统上。可以通过输入sudo losetup -d /dev/loop0将其删除。

1. 系统实现

本课题设计并实现的个人信息保密系统主要是通过实现磁盘创建、挂载、卸载、删除和文件管理功能来保护信息的安全。本章阐述了个人信息保密系统主要功能的设计和实现过程。

代码目录（图4-1）：

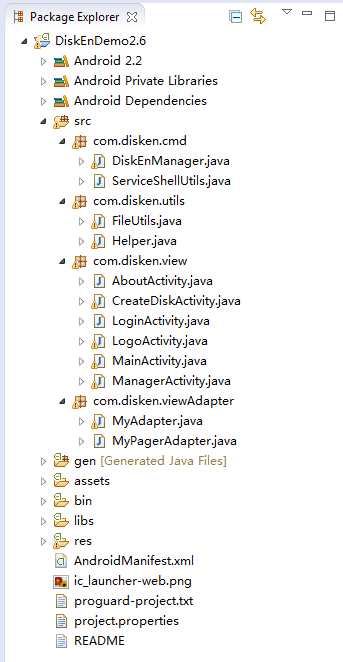


图4-1 代码目录

* 1. 模块设计

个人信息保密系统的主要功能共分为五个方面：磁盘创建、挂载、卸载、删除和文件管理。

软件流程图如图4-2：

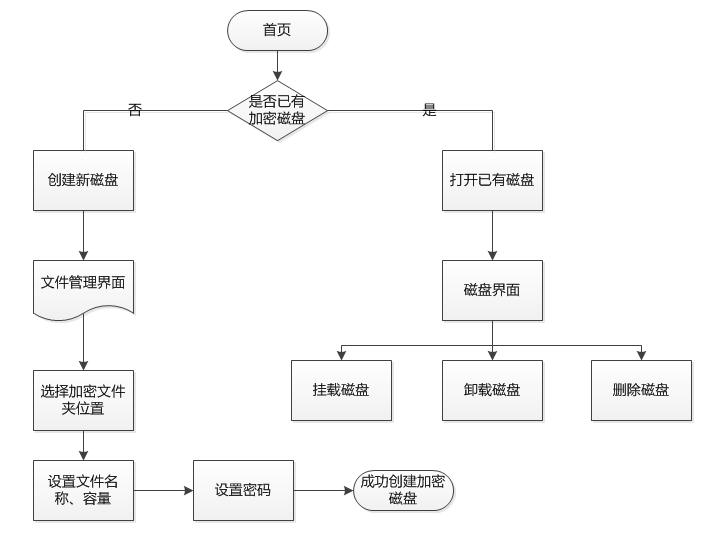


图4-2 软件流程图

* + 1. 图像特征提取

在卸载加密设备后，非常可能还需作为普通用户来装载他们，可以完成dm-crypt设备的创建和卷的装载工作，将/dev /DEVICENAME用实际设备的名称或文件路径来替换：

* 首先需要选择创建加密磁盘的位置，再选择磁盘名称、磁盘容量、磁盘密码，然后击创建保险箱，待磁盘格式化完成后，自动生成加密磁盘文件夹。
* 加密过程采用Twofish算法。
* 创建回送文件（镜像文件）
* 创建环路设备
* 创建逻辑卷
* 创建设备映射程序
* 创建文件系统，文件系统格式为ext3。
* 文件系统加载权限。
* 文件系统格式化权限。

**实现方法：**

1. **交叉编译cryptsetup**

我们可以在在Android下调用cryptsetup进行磁盘操作。首先，需要使用Android NDK交叉编译出可执行的二进制文件cryptsetup。方法如下：

* 1. 在电脑上建立NDK
  2. 安装需要的软件包，运行命令：

sudo apt-get install autoconf automake libtool autopoint git make patch

* 1. 获取源代码：

 git clone git://github.com/guardianproject/LUKS.git

* 1. cd LUKS/external
  2. make NDK\_BASE=/usr/local/android-ndk-r5（路径是NDK所在位置）

该步骤是用NDK工具进行交叉编译，成功后会在文件夹sbin中产生cryptsetup.static文件。

* 1. adb push sbin/cryptsetup.static /data/local/cryptsetup

将cryptsetup.static文件导入手机系统文件/data/local/中，并更名为cryptsetup，该步骤需在获得管理员权限的前提下进行

1. **创建磁盘加密文件**
2. 创建一个回送文件（镜像文件）

mknod /dev/loop0 b 7 0

dd dev/zero of=~/mydisk.img bs=1M count=250

该处~/mydisk.img 是回送文件的路径,可以根据自己的需要进行创建,若文件不存在，则会自动创建,若文件已存在，则将覆盖。bs 为扇区的大小,这里设置成 1M，count 为扇区的个数,这里设置为 250个，因此总容量为， 1M\*250=250M

1. 下面利用上面建立的回送文件建立一个环路设备

losetup /dev/loop0 /mnt/sdcard/mydisk.img

这里建立的环路设备是/dev/loop0

（losetup -d /dev/loop0命令为删除名称为/dev/loop0的环路设备）

1. 利用环路设备创建一个逻辑卷

echo "Password" | cryptsetup -q --key-file=- luksFormat -c aes-plain /dev/loop0

在提示符下输入大写的YES，然后输入密码，此时逻辑卷就创建了。

1. 为环路设置创建一个设备映射程序

echo "Password" | cryptsetup luksOpen --key-file=- /dev/loop0 myEnFs

此时也需要输入密码

1. 创建ext3文件系统

mke2fs -O uninit\_bg,resize\_inode,extent,dir\_index –L myEnFs /dev/mapper/myEnFs

1. **权限问题**

在SD卡上进行操作将会涉及到以下三个权限，皆需在AndroidManifest.xml里标明：

SD卡读写权限：

android.permission.WRITE\_EXTERNAL\_STORAGE

加载卸载文件系统权限：

android.permission.MOUNT\_UNMOUNT\_FILESYSTEMS

加载并格式化文件系统的权限:

android.permission.MOUNT\_FORMAT\_FILESYSTEMS

* + 1. 原始数据处理
* 挂载虚拟磁盘。
* 获得外部存储设备读写权限。
* 在主界面点击加载磁盘，输入密码，则会在SD上生成该磁盘文件夹，相当于虚拟磁盘的功能。

**实现方法：**

1. 在sdcard根目录新建一个文件夹myEnFs，相当于虚拟磁盘。

mkdir /mnt/sdcard/ myEnFs

1. 挂载虚拟设备。

ount -o rw -t ext3 /dev/mapper/myEnFs /mnt/sdcard/myEnFs

* + 1. 模型训练
* 反挂载虚拟设备。
* 反映射虚拟设备。
* 在磁盘内文件处理完毕后，选择关闭并卸载磁盘则可以将磁盘关闭，此时在SD卡上看不到该磁盘文件夹，保证了文件的安全性。

**实现方法：**

1. 反挂载虚拟设备

umount /mnt/sdcard/myEnFs

1. 反映射虚拟设备

cryptsetup luksClose myEnFs

* + 1. 交叉验证
* 卸载虚拟设备。
* 反映射虚拟设备。
* 删除虚拟设备。

**实现方法：**

1. 卸载虚拟设备

umount /mnt/sdcard/myEnFs

1. 反映射虚拟设备

cryptsetup luksClose myEnFs

1. 删除加密文件

rm –r /mnt/sdcard/mydisk.img

1. 删除磁盘路径

rm /mnt/sdcard/myEnFs

1. 删除环路设备

losetup -d /dev/loop0

* + 1. 模型融合
* 该软件中文件管理器功能调用了Android自带的SQLite数据库，部分文件格式会有相应的图标。
* 在加载磁盘后，可以通过文件管理器在磁盘内对文件及文件夹进行新建、修改、删除等操作。
* 通过调用Android系统的API，选择合适程序打开指定文件。

**具体过程：**

1. 调用SQLite数据库

单独用一个MyAdb.java文件建立数据库。在Android系统中，每个应用只能调用其各自建立的数据库。

首先导入数据库包：

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteOpenHelper;

包括如下四个方法：

* onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)函数用于在数据库改变时，删除原表，建立新表
* getFileSet()函数用于查询所有记录
* insertFileSet(int isZoom, int isOpen)函数用于新增记录
* updateFileSet(int id, int isZoom, int isOpen)函数用于根据ID修改记录

1. 文件夹操作

文件管理器可以识别文件夹与某些文件的类型、大小，并用相应的图标显示，更简单直观。通过MimeType的方法getMIMEType(File f, boolean isOpen)，可以识别常见的媒体格式并选择打开方式，调用外部程序打开对应文件。

此外，长按文件可以选择重命名、复制、剪切、粘贴、删除等操作功能，也可新建文件（夹），系统会自动判断是否重名和是否符合命名规范。

* 1. 图像特征提取

我们通过一系列命令的组合实现对磁盘的各种操作，因此采用了脚本语言的方式来确保系统可以依照正确的顺序自动执行命令。我们将执行脚本的方法doShellCommand()定义在ServiceShellUtils.java文件中，每一次执行命令，都可以通过调用doShellCommand()方法，同时传入对应的参数。

* + 1. HSV

public static int doShellCommand(String[] cmds, StringBuilder log,boolean runAsRoot, boolean waitFor) throws Exception

{

…………………………

Process proc = null;

// “su”是用管理员权限执行命令进程，“sh”与之相反。

if (runAsRoot) proc = Runtime.getRuntime().exec("su");

else proc = Runtime.getRuntime().exec("sh");

//用一个for循环执行命令，每执行完一条语句，回车执行下一条，在所有语句执行完后exit回车退出。

OutputStreamWriter out = new OutputStreamWriter(proc.getOutputStream());

for (String cmd : cmds)

out.write(cmd + "\n");

out.write("exit\n");

out.flush();

//判断是否在等待输出。

if (waitFor == false)

return -1;

………………

int exitCode = proc.waitFor();

log.append("Process exit code: ");

log.append(exitCode);

log.append("\n");

//打印返回值，查看运行状况。

logNotice("command process exit value: " + exitCode);

return exitCode;

}

* + 1. LBP

以创建磁盘为例，在DiskEnManager.java文件中的createSecretFile(参数)方法。

首先要定义完整的命令语句，即一个较长的字符串：

String[] cmds =

{

"mknod " + loopback + " b 7 0",

"dd if=/dev/zero of=" + storePath + " bs=1M count=" + size,

LOSETUP\_BIN + " " + loopback + " " + storePath,

"echo \"" + password + "\" | " + CRYPTSETUP\_BIN

+ " -q --key-file=- luksFormat -c aes-plain "+ loopback

};

在定义字符串的时候，特别要注意空格和引号的使用。echo 指令是用来输入参数的，反斜杠用来输入脚本中的引号。

代码：

StringBuilder log = new StringBuilder();

int exitCode = ServiceShellUtils.doShellCommand(cmds, log, true, true);

这里调用ServiceShellUtils(参数)方法，参数中包括指令语句cmds等。

* + 1. Dark Channel
  1. 原始数据处理
     1. 格式统一

编译cryptsetup除了cryptsetup源码包之外还需要LVM、libgcrypt、popt、util-linux-ng等包，如果没有在PC安装这些包，在编译过程中会报错并提示缺少某些文件，此时再按要求下载所需要的包即可。

编译完成后，在sbin目录下生成cryptsetup.static文件，就是我们所需要的可执行二进制文件，将其导入手机中就可以进行相应的操作

* + 1. 归一化处理

private static File installCryptSetup(Context context)

{

try

{

AssetManager assetmanager = context.getAssets();

File applicationFilesDirectory = context.getFilesDir();

//applicationFilesDirectory = com.encrypt.test/files，在手机的系统data文件夹下

File binDir = new File(applicationFilesDirectory, "bin");

//binDir = com.encrypt.test/files/bin

if (!binDir.exists())

binDir.mkdirs();

File binAppFile = new File(binDir, "cryptsetup");

//创建cryptsetup空文件binAppFile = com.encrypt.test/files/bin/cryptsetup

InputStream in = assetmanager.open("cryptsetup");

OutputStream out = new FileOutputStream(binAppFile);

byte[] buffer = new byte[1024];

int read;

while ((read = in.read(buffer)) != -1)

{

out.write(buffer, 0, read);

}

//导入cryptsetup的过程就是不断读取cryptsetup内容然后写入的过程。

in.close();

in = null;

out.flush();

out.close();

out = null;

Runtime.getRuntime().exec("chmod 755 " + binAppFile.getAbsolutePath());

return binAppFile;

} catch (Exception e) {

Log.e(TAG, "Error copying app file" + e.getMessage());

}

return null;

}

在开始的时候，使用ZipInputStream方式压缩读取写入，即：

ZipInputStream in = new ZipInputStream(context.getAssets().open("cryptsetup"));

in.getNextEntry();

结果因为某些错误导致导入到手机中的cryptsetup文件是个空文件。使用这种方法的原因是我当初认为Android2.3不支持直接拷贝大于1M的文件，于是进行了压缩处理。后来得知在Android2.3以后支持了大文件的拷贝，于是直接使用InputStream方式。

导入到手机中成功后，却在运行测试的过程中出现另一个问题，该文件无法使用，后来我意识到必须给导入到系统中的文件赋予一定的权限，于是加入了Runtime.getRuntime().exec("chmod 755 " + binAppFile.getAbsolutePath())命令，给cryptsetup文件赋予了755权限，问题随之解决。

* + 1. 数据分块

private static File installCryptSetup(Context context)

{

* 1. 模型训练
     1. 针对特征训练

界面设计工作在飞速发展的互联网潮流中，越来越多的被重视起来。其实软件界面产品的重要卖点，就像传统工业中的商品的工业设计。一个拥有美观的界面的软件会给人带来更加舒适的视觉享受，并进一步拉近人与商品的距离。检验一个软件产品界面设计的好坏，既不是该产品组领导的意见，也不是项目成员投票的结果，而是最后实际的感受，这是建立在科学性之上的艺术设计[9]。

由于本应用以Android移动设备作为基础平台，且将主要应用场景定位于随身使用，故将一切使用从简，从最简单的操作出发。分析实用者的需求与使用流程，可以归结为创建加密文件夹、管理文件夹，再次进入文件夹这三个步骤。本应用也分为三个主要界面：新建界面、管理界面、登陆界面，分别对应上述三个步骤。三个界面如下三图所示：

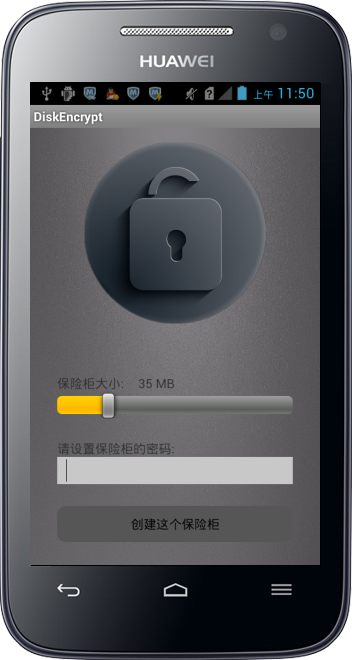
 

图4-3 新建加密文件夹界面 图4-4 文件夹管理界面



图4-5 登陆界面

* + 1. 模型融合

按照MVC原则，界面应该与程序逻辑相分离。在此基础上，程序的界面方案的调整也容易得多，故该应用中采用了在XML文件中定义界面的方式。

与创建一个普通的HTML文档相似，通过XML设计界面的特点如下：

1. 大量的自定义元素在XML中进行定义，各自代表了特定的Android显示子类。
2. 设计应用的界面可以通过保存在应用res/layout目录下的XML文件中一系列的嵌套标签来实现，就像创建HTML文档一样。
3. 每个文档描述一个android.view.View，这个元素既可以是包含了一个集合的版面设计的元素的一个子节点，也可以是一个简单的显示元素，当Android编译应用程序时，系统会将每个文件都编译进Android系统。为了加载显示资源，可以在代码Activity.onCreate()实现中通过调用setContentView(R.layout.layout\_file\_name)方法 [10]。
   1. 交叉验证

文件管理器在本应用中为用户进行主要操作界面，涉及到诸多主要功能与函数且工作量较大，故在此进行详细说明。

* + 1. 分块

public void onCheckedChanged(RadioGroup group, int checkedId)

{

if (checkedId == rb\_file.getId())

{

new\_textView.setText("新建文件:");

} else if (checkedId == rb\_dir.getId())

{

new\_textView.setText("新建文件夹:");

}

}

1. checkbox选中文件夹后，点击确定，判断是否有重名，名字是否符合规范（if (newName.indexOf("\\") == -1)），最后提示是否创建成功。
2. checkbox选中文件后，点击确定，调用java库中的createNewFile（）方法新建文件。
   * 1. 测试

如果文件夹不存在，则建立新文件夹，然后复制文件夹下的文件

new File(d\_new).mkdirs();

File[] files = f\_old.listFiles();

for (int i = 0; i < files.length; i++)

{

d\_old = oldPath + File.separator + files[i].getName();

if (files[i].isFile())

{

copyFile(d\_old, d\_new);

} else

{

copyDir(d\_old, d\_new);

}

* 1. 模型融合

文件管理器在本应用中为用户进行主要操作界面，涉及到诸多主要功能与函数且工作量较大，故在此进行详细说明。

* + 1. 融合方法设计

public void onCheckedChanged(RadioGroup group, int checkedId)

{

if (checkedId == rb\_file.getId())

{

1. 实验数据分析
   1. 数据集分析
      1. 训练集分析

* 功能简单实用，适合移动端设备。
* 采用高效的Twofish加密算法，提升用户体验。
* 使用范围广法，支持各类Android设备。
  + 1. 测试集分析

本课题所研究的移动智能终端的个人信息保密系统旨在保护Android设备上的数据与隐私安全。由于现在大部分Android设备都配备了SD卡，并将SD卡作为存储与传递信息的媒介。因此，本课题所采取的信息保护手段为在Android设备上建立虚拟磁盘，以此在SD上生成新的磁盘文件夹，并在其中存放信息与数据。

通过该信息保密系统，用户可以直接在Android设备上管理文件、图片等各类信息与数据，最大限度的保证信息与数据的安全性和完整性。同时，加密过程也是对用户保持透明的，并不会因为加解密而降低用户体验。

* + 1. LBP

本信息保密系统只由一个本地客户端构成，不需要与服务器通信，且密钥存储与设备中。文件管理器功能通过调用Android自身的SQLite数据库实现，UI界面由各类图片与控件自行编写。

* 1. 训练方法分析
     1. 核函数选择
     2. 参数调整

功能简单

* 1. 融合方法分析

功能简单

1. 总结
   1. 成果小结

随着智能移动终端的普及，以及互联网技术的高速发展，移动时代已经逐渐来临。由于各类第三方软件的出现，人们有越来越多的时间花费在移动设备上，也有越来越多的数据存储在移动设备上。如今市场占有份额最大的两款移动设备的操作系统分别是Android与IOS，其中占有率为44.62%的Android系统由于其开放性引来了很多恶意程序，首当其冲的就是个人数据的盗取与泄露。从个人隐私到国家秘密，任何一条敏感数据的传播和复制，都可能给个人或国家带来危害。移动时代给人们带来的不仅有便捷，还有潜在的危机。

同时，手机、平板等移动由于其便携的特性导致其遗失和被盗窃的几率相比PC更大，存储在这些移动设备内部的重要资料遗失或者泄露的风险也更大，一些私密信息也更容易被人侵犯。在本文中，主要对移动智能终端的个人信息保密系统做了较为详细的设计和部分的实现。

本文主要展开了一下几个方面的工作：

依托数据，分析了社会上关于Android设备的安全现状以及潜在威胁，并总结出急需弥补与完善的地方。在此之上，结合现有的技术设计了个人信息保密系统的大致架构以及所需功能。并通过分析选择了Twofish算法作为本课题的核心部分

对个人信息保密系统进行了总体分析与思考，提出了该系统的应用场景、软硬件架构和系统特色，并对本系统中使用的核心技术-磁盘加密的控制命令作了较为详细的阐述。

简单描述了磁盘创建、挂载、卸载、删除、文件管理五个主要功能，并对脚本编写、cryptsetup编译导入、UI设计、页面跳转、文件管理器设计等几个个方面进行了详细描述。

在本次毕业设计中，结合各项现有技术，进行了分析与拓展，基本实现了移动智能终端的个人信息保密系统的各项功能，基本上完成了实现Android底层磁盘加密的预期目的。

* 1. 课题展望

在上一次计算机与互联网时代到来时，技术的进步给人们带来了许多便利，也极大地促进了社会的发展。但是任何事物都有两面性。互联网非法行为给人们带来成百上千亿的损失，更重要的是窥伺并泄露了人们的隐私，同时也极大的影响到了人们的正常生活。

而移动互联网相比传统互联网，更加的贴身和方便，也存储了更多信息与数据。现在各个硬件平台趋紧于整合，手机不仅仅是以前的通讯设备，更是未来承载整个移动互联网的硬件，其代表的意义不言而喻。另一方面，Android系统作为第一大手机操作系统，其开放的特性得到了广大的认可，并建立了良好的生态环境。正因为此，才更需要注重Android系统移动设备的信息安全性。只有保障了数据与信息的安全，整个移动互联网才能朝着更加先进的方向发展。

参考文献

1. 《经济全球化对我国智能手机发展的影响》 作者: 杨国品、田中景；《现代营销(学苑版)》期刊；出版年: 2015-4；
2. 《基于Android的手机定位软件》 作者: 冯玉平、王曙光；《计算机与现代化》期刊；出版年: 2015-3；
3. 《Android的设计与实现》 作者: 杨云君；机械工业出版社；出版年: 2013-5；
4. 《Google Android开发入门指南》 作者: 王世江；人民邮电出版社；出版年: 2009-11；
5. 《疯狂Android讲义》 作者: 李刚；电子工业出版社；出版年: 2013-3；
6. 《基于Twofish的加密技术在电子政务系统中的研究与实现》 作者: 郭成；武汉理工大学；出版年: 2013-4；
7. 《密码学原理与实践》 作者: Douglas R.Stinson、冯登国；电子工业出版社；出版年: 2009-7；
8. 《TrueCrypt磁盘加密软件安全机制分析》 作者: 李晖；《微计算机信息》期刊；出版年: 2010-2；
9. 《认知与设计:理解UI设计准则》 作者: Jeff Johnson、张一宁；人民邮电出版社；出版年: 2014-8；
10. 《打造高质量Android应用:Android开发必知的50个诀窍》 作者: 塞萨 (Carlos Sessa)、杨云君；机械工业出版社；出版年: 2014-4；

致　　谢

首先诚挚地感谢我的毕业设计指导老师雷敏老师，他对学问的严谨态度值得我学习。雷老师不仅在学业上言传身教，而且以其高尚的品格给我以情操上的熏陶。本文的写作更是直接得益于他的悉心指点，从论文的选题到体系的安排，从观点推敲到字句斟酌，无不凝聚着他的心血。滴水之恩，当以涌泉相报，师恩重于山，师恩难报。我只有在今后的学习、工作中，以锲而不舍的精神，努力做出点成绩，以博恩师一笑。

同时，也要感谢大学四年中传授我知识的各位老师，是老师们的悉心教导，让我学到了各种各样的知识和技能，让我成长为一个合格的大学毕业生，可以有充足的准备迈入社会。

最后，我要感谢所有帮助过我和即将帮助我完成论文答辩的各位老师与专家，并对论文中参考文献的作者表示感谢!