**存档日期： 存档编号：**





NanChang Hangkong University

**本科毕业设计**

UNDERGRADUATE DESIGN

**设 计 题 目：** 基于机器学习的互联网医疗诊APP

的设计与实现

**姓 名：**  占文勋

**专 业：** 软件工程（东软班-嵌入式方向）

**班 级 、 学 号：**  162041、16204126

**指 导 教 师：**  陈德圣

**学士学位设计原创性声明**

本人声明，所呈交的设计是本人在导师的指导下独立完成的研究成果。除了文中特别加以标注引用的内容外，本设计不包含法律意义上已属于他人的任何形式的研究成果，也不包含本人已用于其他学位申请的论文或成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式表明。本人完全意识到本声明的法律后果由本人承担。

作者签名： 日期： 年 月 日

**学位设计版权使用授权书**

本学位论文作者完全了解学校有关保留、使用学位设计的规定，同意学校保留并向国家有关部门或机构送交设计的复印件和电子版，允许设计被查阅和借阅。本人授权南昌航空大学可以将本设计的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位设计。

作者签名： 日期： 年 月 日

导师签名： 日期： 年 月 日

**摘要**

如今随着科技、经济水平的不断发展，人们对健康的需求指数也越来越高。但目前，病患主要的问诊形式依旧是以前往目标医院进行咨询的传统模式为主。这种形式将极大的浪费患者的时间，同时，也正是这种医疗模式使得很多人小病不医致使最终酿成大病。目前普通医疗体系不能永远保持精确又快速的诊断,通过机器学习辅助医疗不仅能大幅削减成本, 其诊断结果几乎能实时获取。越来越多的情况下，通过机器学习进行的诊断能够比老练的医生提供更准确的诊断。开发一款基于机器学习的医疗APP，初衷在于解决看病难，看病浪费时间大的问题。无论在何地，通过APP就能进行医疗诊断，为患者提供相应的治疗方案、注意事项与健康生活习惯计划，为用户的小病进行定制化治疗，帮助用户早日恢复健康。若诊断判定病症较严重，将提醒患者尽早到医院就医，以保证安全性。

本文采用基于TensorFlow的机器学习模型实现在线诊断，并且向用户提供完整的医疗服务链，包括预约、诊断、咨询、购药等功能。系统可以分为App端和Bmob管理后台，App端使用Android原生的Bottom Navigation Activity模板进行编写，后台采用Bmob云数据库对系统各表数据和用户权限进行管理。

本系统实现了用户登录注册、管理预约就诊信息、购买药品及查看订单、医患交流、医院预约挂号、医院导航、浏览健康资讯、查询疾病、医院、药品等信息以及在线诊断等功能，大大提高了患者就诊效率，达到了系统设计的预期。

**关键词：**互联网医疗 机器学习 自主诊断 服务链

**Abstract**

Nowadays, with the continuous development of science and technology and economic level, people's demand index for health is getting higher and higher. But at present, the main form of patient consultation is still based on the traditional mode of consulting at the target hospital. This form will greatly waste the patient's time. At the same time, it is this medical model that makes many people suffer from minor illnesses and eventually become serious illnesses. At present, the general medical system cannot always maintain accurate and rapid diagnosis. The aid of medical learning through machine learning can not only significantly reduce costs, but also obtain the diagnosis results in real time.In more and more cases, diagnosis through machine learning can provide a more accurate diagnosis than experienced doctors. Develop a medical APP based on machine learning, the original intention is to solve the problem of difficulty in seeing a doctor and wasting time in seeing a doctor. No matter where you are, you can make medical diagnosis through APP, provide patients with corresponding treatment plans, precautions and healthy lifestyle plans, and customize treatment for users' minor illnesses to help users recover health as soon as possible. If the diagnosis determines that the condition is serious, the patient will be reminded to go to the hospital as soon as possible to ensure safety.

This article uses a TensorFlow-based machine learning model to achieve online diagnosis and provide users with a complete medical service chain, including appointment, diagnosis, consultation, drug purchase and other functions. The system can be divided into App side and Bmob management background. App side uses Android's native Bottom Navigation Activity template to write. The background uses Bmob cloud database to manage the system table data and user permissions.

This system realizes functions such as user login and registration, management of appointment information, purchase of medicines and order viewing, doctor-patient communication, hospital appointment registration, hospital navigation, browsing health information, querying diseases, hospitals, drugs and other information, and online diagnosis This improves the efficiency of patient visits and meets the expectations of system design.

**Key Words：**Internet MedicalMachine Learning Self-diagnosis Service Chain

**目 录**

[摘要 I](#_Toc29177)

[Abstract II](#_Toc27472)

[目 录 III](#_Toc22356)

[图清单 V](#_Toc15771)

[表清单 VI](#_Toc13438)

[1 绪论 1](#_Toc6299)

[1.1 课题意义和目标 1](#_Toc35)

[1.2 国内外研究现状 1](#_Toc863)

[1.3 论文的主要工作 2](#_Toc11232)

[1.4 论文的组织结构 2](#_Toc15539)

[1.5 本章小结 3](#_Toc18536)

[2 系统分析 4](#_Toc4908)

[2.1 可行性分析 4](#_Toc12264)

[2.2 需求分析 5](#_Toc20103)

[2.3 方案比选 10](#_Toc4199)

[2.3.1 方案一 10](#_Toc15618)

[2.3.2 方案二 11](#_Toc15917)

[2.4 本章小结 11](#_Toc19621)

[3 系统的设计 12](#_Toc13955)

[3.1 软件体系结构 12](#_Toc6025)

[3.2 功能设计 12](#_Toc15683)

[3.3 持久化设计 14](#_Toc17314)

[3.4 社会健康、文化、法律相关设计 17](#_Toc5135)

[3.5 本章小结 18](#_Toc22938)

[4 系统的实现 19](#_Toc48)

[4.1 医疗诊断 19](#_Toc5052)

[4.2 用户操作管理 22](#_Toc15458)

[4.3 医院服务 24](#_Toc27905)

[4.4 医疗信息显示和查询 26](#_Toc7711)

[4.5 本章小结 28](#_Toc14782)

[5 系统运行与效果分析 29](#_Toc12841)

[5.1 界面设计概要 29](#_Toc7810)

[5.2 用户管理 29](#_Toc29273)

[5.3 诊断购药 30](#_Toc7972)

[5.4 医院医生服务 33](#_Toc1794)

[5.5 医疗信息管理 35](#_Toc3866)

[5.6 Bmob管理后台 37](#_Toc22640)

[5.7 本章小结 37](#_Toc13696)

[6 系统测试 37](#_Toc6470)

[6.1 测试方法 38](#_Toc19078)

[6.2 测试方案及计划 38](#_Toc30909)

[6.3 测试过程及结果分析 39](#_Toc24860)

[6.4 本章小结 39](#_Toc25637)

[7 总结与展望 41](#_Toc9994)

[7.1 总结 41](#_Toc14605)

[7.2 展望 41](#_Toc30433)

[参考文献 42](#_Toc5625)

[致谢 43](#_Toc21099)

**图清单**

|  |  |  |
| --- | --- | --- |
| 图序号 | 图名称 | 页码 |
| 图2-1 | 患者用户用例图 | 6 |
| 图2-2 | 医生用户用例图 | 7 |
| 图2-3 | TensorFlow系统架构图 | 10 |
| 图2-4 | Core ML层级结构图 | 11 |
| 图3-1 | 系统功能结构图 | 12 |
| 图3-2 | 系统类图 | 13 |
| 图3-3 | 自诊功能时序图 | 13 |
| 图3-4 | 自诊功能活动图 | 14 |
| 图3-5 | E-R图 | 15 |
| 图3-6 | 数据库表图 | 15 |
| 图4-1 | 损失值曲线 | 21 |
| 图4-2 | 准确率曲线 | 21 |
| 图5-1 | 用户登录界面 | 29 |
| 图5-2 | 用户注册界面 | 30 |
| 图5-3 | 个人信息界面 | 30 |
| 图5-4 | 密码管理界面 | 31 |
| 图5-5 | 疾病自诊界面 | 31 |
| 图5-6 | 诊断结果界面 | 32 |
| 图5-7 | 药品购买界面 | 32 |
| 图5-8 | 医院服务界面 | 33 |
| 图5-9 | 医院预约界面 | 34 |
| 图5-10 | 医生服务界面 | 34 |
| 图5-11 | 医患记录界面 | 35 |
| 图5-12 | 疾病资讯界面 | 36 |
| 图5-13 | 预约就诊信息界面 | 36 |
| 图5-14 | Bmob管理后台界面 | 37 |
| 图6-1 | 预约缺陷界面 | 40 |
| 图6-2 | 模糊搜索缺陷界面 | 40 |

**表清单**

|  |  |  |
| --- | --- | --- |
| 表序号 | 表名称 | 页码 |
| 表2-1 | 填写症状表格用例描述 | 7 |
| 表2-2 | 登录用例描述 | 8 |
| 表2-3 | 预约挂号用例描述 | 8 |
| 表2-4 | 购买药品用例描述 | 8 |
| 表2-5 | 辅助医疗诊断用例描述 | 9 |
| 表3-1 | 患者信息表 | 16 |
| 表3-2 | 医生信息表 | 16 |
| 表3-3 | 医院信息表 | 16 |
| 表3-4 | 疾病信息表 | 17 |
| 表3-5 | 药品信息表 | 17 |
| 表4-1 | 数据集样例 | 19 |
| 表6-1 | 测试功能范围 | 38 |
| 表6-2 | 功能测试用例 | 39 |

# 1 绪论

## 1.1 课题意义和目标

目前市面上有许多与健康医疗相关的应用，并且机器学习技术也在医疗领域体现出极大的前景和价值。但大多数应用都只覆盖到小部分的服务范围，没有提供完善的医疗服务，需要用户同时使用多个相关应用，其疾病诊断也依旧是以传统的咨询问诊为主，极大的浪费了患者的时间。向用户提供自诊、推荐治疗方案、医院预约以及药品购买等一整套医疗服务链，同时，通过机器学习训练出成熟的诊断模型，辅助医生诊断和用户自诊。这将给看病带来极大的便利，也更好地推动了互联网在医疗领域的发展。

## 1.2 国内外研究现状

“互联网+医疗”指的是以互联网作为载体，运用大数据、机器学习等新兴技术将现有医疗资源整合在一起，为患者提供科学的医疗服务。近年来，互联网医疗正在不断地普及，其应用范围也在逐渐扩大，例如互联网医疗平台可以处理和分析大量医学相关数据，其模式也主要应用于量血压、测心率等基本临床医学领域。“互联网+医疗”正逐步向可穿戴设备、智能视频分析以及医学信息共享等方向持续发展，其应用也让用户体会到传统领域科技化带来的便利服务[1]。

### 1.2.1 国外研究现状

国外的互联网医疗领域发展较为迅速，相关的移动应用也逐步投入运用，欧美等发达国家的远程医疗应用已经逐渐成熟。其中美国远程医疗已覆盖到长期护理评估、心理治疗和健康体检访问等服务，其医疗项目也在逐年增加[3]。美国科学家曾通过远程医疗技术指导中风患者紧急自救，ZocDoc和WellDoc等公司也实现了预约医生和慢性疾病管理等医疗服务，IBM公司通过基因组测序和科学计算实现了快速诊断肿瘤类疾病。日本和德国在近年来开发了类似的医疗系统，用于救治心脏病和治疗皮肤病，法国也在医疗领域中设计了双向传感系统，同步检测病人身体状况，并对异常指标变化进行警报处理。随着互联网和医疗之间的不断合作发展，也推动了物联网技术应用创新，ZEO公司通过腕带和头贴等蓝牙设备提升用户的睡眠质量，并解决睡眠问题和提供睡眠指导[5]。在医疗大数据领域，国外已开展了Health Infoway、NHIN、PatientsLikeMe以及CardioNET等多个大数据项目，实现了临床医学数据、患者病历等医疗数据共享，使得医疗行业能够健康顺利地持续发展。

### 1.2.2 国内研究现状

虽然国内互联网医疗起步较晚，但经过了信息服务、咨询服务和诊疗服务发展阶段，在近年来正式转为智能医疗阶段，通过引入机器学习、云计算等新兴技术，互联网医疗领域服务范围基本覆盖，也不断催生着许多新的商业模式蓬勃发展。现如今，我国人民随着生活质量逐年提高，其身体健康意识也逐渐完善，从2017年起，我国的智慧医疗医院已达到上百家，预测未来十年互联网医疗将依旧保持快速增长的状态[3]。2018年2月，国内互联网医疗App用户达到了2864.9万人次，主要有预约挂号、医生问诊和药物购买配送等分类，国内互联网医疗依靠着庞大的互联网用户活跃量得到了迅速发展[7]。

## 1.3 论文的主要工作

本课题主要是实现一个基于机器学习技术，方便用户自诊和辅助医生诊断以及提供完整医疗服务链的互联网医疗App。在开发过程中，将使用Kaggle的开源数据库和API，通过机器学习训练和迭代改进医疗诊断模型。患者可以通过App进行自主诊断、管理个人健康信息、预约医院服务和购买药品，医生可以进行辅助医疗诊断、管理就诊人和个人信息等。主要研究内容如下：

1. 使用机器学习技术对临床医疗数据进行分析，训练出准确、可信的医疗诊断模型。
2. 实现完整的医疗服务链，包括诊断、预约、治疗、购药、急救以及反馈等医疗服务功能。
3. 基于Bmob云端数据库，设计适合本课题的数据库结构，从而提高数据交互效率。
4. 依据系统各模块所要实现的功能，设计出合理的类结构，使得其业务流程清晰、用户使用简便。
5. 计划出合理的测试方案，对系统业务逻辑、重要功能、安全性与兼容性进行测试和验证。

## 1.4 论文的组织结构

本论文是对开发基于机器学习的互联网医疗诊断App的过程进行描述，总共分为7个章节，各章节的主要内容如下：

1. 绪论。本部分简要介绍了“互联网+医疗”的研究现状与发展以及选择本课题的目的和意义。
2. 系统分析。本部分对基于机器学习的互联网医疗诊断App的相关理论和技术进行分析，明确了本课题需要实现的功能和期望达到的目标。
3. 系统的设计。本部分主要是对基于机器学习的互联网医疗诊断App进行概要设计和详细设计，首先依照系统的功能性需求设计出了主要业务流程，然后介绍系统框架设计以及数据库设计。
4. 系统的实现。本部分对互联网医疗App详细介绍了各模块的实现逻辑，并给出了本系统的类图、时序图以及活动图。
5. 系统运行与效果分析。本部分主要是展示App的具体操作界面，并有标注文字解释其逻辑。
6. 系统测试。本部分对基于机器学习的互联网医疗诊断App的各模块功能和性能进行测试，并对测试结果进行分析。
7. 总结与展望。本部分对全文进行分析和总结，介绍本系统的不足之处以及对应的改进方案。

## 1.5 本章小结

本章首先讲述了基于机器学习的互联网医疗App的选题意义和目的，并介绍了国内外互联网医疗的研究和发展现状，最后详细介绍了本论文的主要工作和组织结构。

# 2 系统分析

互联网医疗诊断App是基于机器学习训练出的诊断模型之上，并且还将提供完整的服务链，包括诊断、提供治疗方案、医院服务、预约医生以及购买药品等功能。本课题将用户分为患者和医生两种，患者作为主要用户，对于患者，App将显示自主诊断、医院服务、预约医生、购买药品以及个人信息模块，患者可以在前四个模块中完成整个医疗过程，大大节省了其宝贵时间。对于医生，App将显示辅助医疗诊断、就诊人管理和个人信息模块，医生可以更好地节省精力，从而去服务更多的患者。

## 2.1 可行性分析

本部分是对于基于机器学习的互联网医疗诊断系统进行可行性分析，并给出对应的结论。

### 2.1.1 技术可行性

本系统由个人负责全程开发，系统可分为App端平台和Python医疗诊断模型。App端平台代码将用Android Studio编写，基于自带的Bottom Navigation Activity模板设计用户操作界面，以Bmob云端数据库作为系统后台，实现高效的数据交互。Python医疗诊断模型将使用PyCharm编写，以120健康网的开源数据作为数据集，基于TensorFlow技术的机器学习模型对其进行训练和迭代改进。

经过仔细分析，互联网医疗诊断App所需的技术可以在如今框架、机器学习模型已成熟的时代得到满足，因此本系统在技术方面上可行。

### 2.1.2 经济可行性

本系统可作为安卓App发布，集成开发环境Android Studio和PyCharm均属于学生免费版，运用到的框架和模型均属于开源技术，系统开发成本不高，系统后期需要不断改进医疗诊断模型，以得到用户对自主诊断功能的信任，属于人力成本。

在收益方面上，系统依靠完整的医疗服务链和准确的诊断模型可以吸引到大量的需求用户，此时在App上租赁广告位可以获取广告收益，也可以与附近医院合作获取投资和服务盈利，帮助医生减轻工作负担。同时，本系统可以和大数据公司交换匿名数据以获取收益，以及销售药物盈利和政府补助。因此，系统在经济效益上有着巨大的潜力，在经济方面上可行。

### 2.1.3 法律、政治及社会可行性分析

本系统运用的集成开发环境（IDE）均为教育版授权，使用的机器学习模型和医疗数据集也属于开源资料，系统的设计与实现均由本人独立完成。本系统使用的患者、医生信息均为匿名数据，不存在故意泄露他人隐私信息，符合法律法规要求。在政治方面上，本系统所有信息均符合社会主义核心价值观，不包含宗教信仰等信息，不传播和贩卖任何非法内容。在社会方面上，本系统的自主诊断功能保证在满足高精准的要求下才会投入市场使用，并且秉承着“小病精准，大病建议，重病就医”的治疗方案观念。因此，本系统在法律、政治和社会方面上可行。

### 2.1.4 可持续发展可行性

本系统作为安卓App以业务逻辑清晰地分为五个模块，各个模块都可以单独抽离出来复用，极大地简化了开发过程。当业务逻辑发生变化时，只需要改变对应模块的代码即可继续投入使用，更好地节约了人力成本和时间资源。同时，患者自主诊断和医生辅助诊断功能都可以及时地处理用户的请求，极大地节省了用户的操作时间。综上，本系统具有良好的可持续发展可行性。

## 2.2 需求分析

系统在需求分析阶段，需要对其提出可能的功能性需求和非功能性需求，以确定本系统需要实现的功能和达到的目标，从而为系统设计阶段做好铺垫。

### 2.2.1 需求描述

本课题的目标是实现一个基于机器学习的互联网医疗诊断移动应用，其可以提高用户医疗诊断的精准度，并且将在运营阶段中不断迭代改进医疗诊断模型。因此，本课题将开发出一个运行在Android手机上的互联网医疗诊断App，在实现完整的医疗服务链的同时，也将提供患者自主诊断和医生辅助诊断功能，根据诊断结果的正确情况对诊断模型进一步改进，实现智能医疗诊断的高精准要求，以提高用户对其的信任和满意程度。

本系统主要分为以下五个功能模块：

（1）医疗诊断模块

患者用户可以在该模块按照身体实际情况填写症状表格，然后交由机器学习训练出的诊断模型进行分析，得到可能患得疾病的报告。医生用户可以在该模块查询病人相关的疾病或症状，也可以通过辅助医疗诊断帮助其进行分析。

（2）医院服务模块

患者用户可以通过该模块，查询当地医院、预约挂号以及拨打急救电话，以便及时了解相关信息和重病抢救。

（3）医生服务模块

医生用户可以通过该模块，管理就诊患者和及时追踪患者病情。

（4）药品购买配送模块

患者用户可以通过该模块查询药品、选择购买药品。

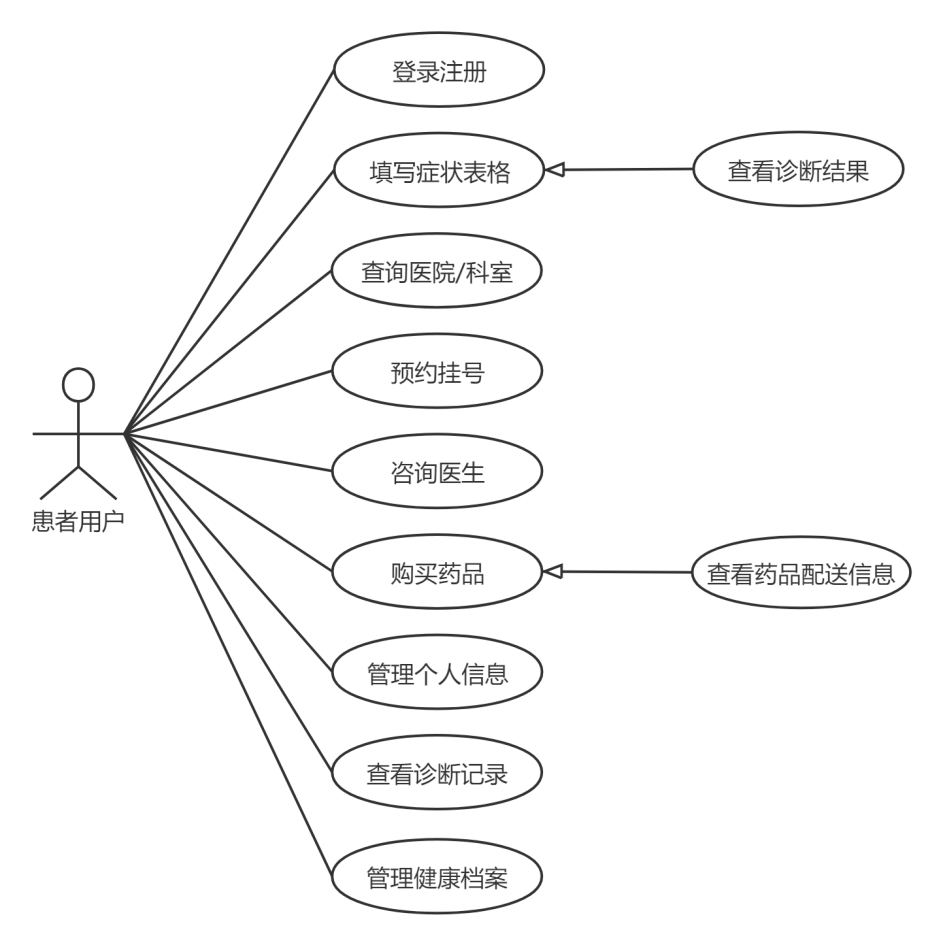
（5）个人信息模块

医生用户可以通过该模块登录注册、修改密码、管理个人信息以及设置App属性。在本模块内，患者用户在医生用户所拥有的功能基础上还可以管理个人的预约记录以及查看诊断历史记录。

### 2.2.2 角色及用例分析建模

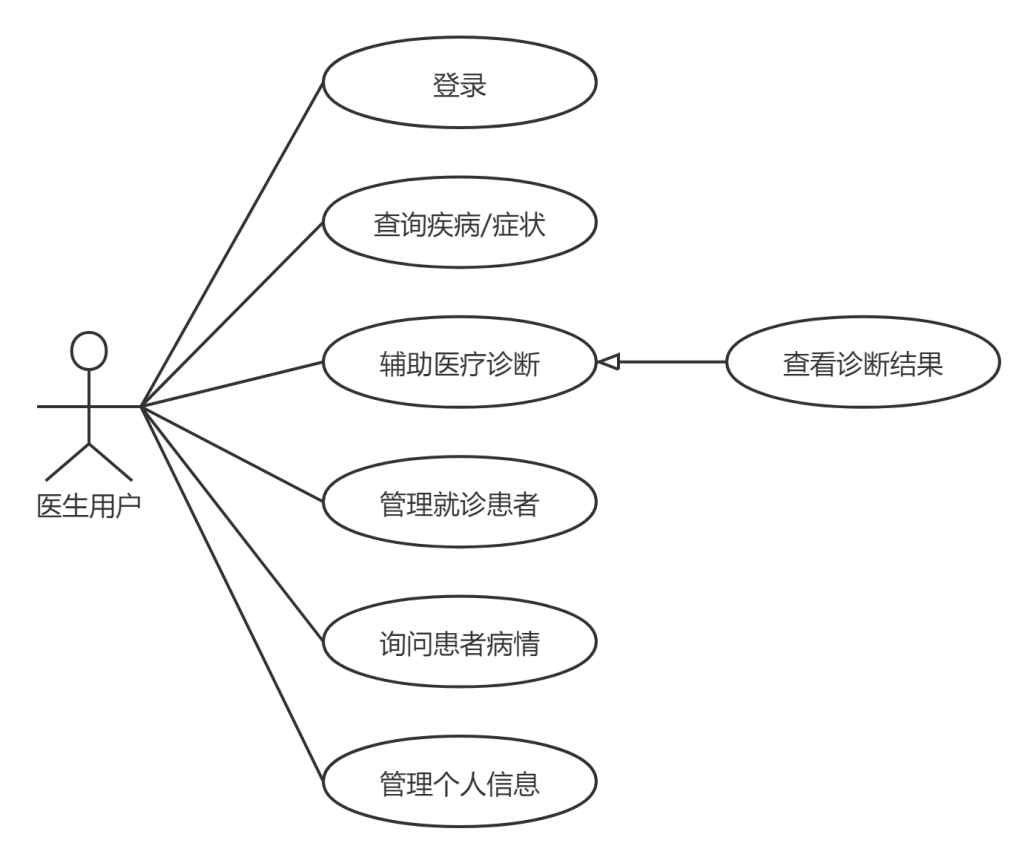
（1）用例分析

图2-1展现了患者用户用例图，患者可以登录注册、填写症状表格并查看诊断结果、查询医院/科室、预约挂号、咨询医生、购买药品并查看药品配送情况、管理个人信息、查看诊断历史记录以及管理健康档案。



**图2-1 患者用户用例图**

图2-2展示了医生用户用例图，医生可以登录、查询疾病/症状、辅助医疗诊断并查看诊断结果、管理就诊患者、询问患者病情以及管理个人信息。



**图 2-2 医生用户用例图**

（2）用例描述

该部分仅对系统的部分用例进行具体描述，表2-1、表2-2、表2-3、表2-4、表2-5分别是对提交症状表格、登录、预约挂号、购买药品和辅助医疗诊断五个用例的描述。

**表2-1 填写症状表格用例描述**

|  |  |
| --- | --- |
| 用例名称 | 填写症状表格 |
| 描述 | 患者用户根据身体实际情况填写对应症状，并完成表格 |
| 标识符 | PC1 |
| 角色 | 患者 |
| 前置事件流 | 前置事件流：患者成功登录App，并进入医疗诊断模块 |
| 主事件流 | 1. 患者进入医疗诊断模块 2. 患者点击自诊 3. 患者按顺序填写症状表格 4. 患者点击提交按钮 5. 若症状表格未填写完整，提示未填写完整 6. 若症状表格填写完整，提示提交成功 7. 用例结束 |
| 其他事件流 | 其他事件流：  1. 网络异常，症状表格提交失败 |

续表2-1

|  |  |
| --- | --- |
| 后置事件流 | 后置事件流：  1. App展示患者的症状分析报告 |

**表2-2 登录用例描述**

|  |  |
| --- | --- |
| 用例名称 | 登录 |
| 描述 | 患者或医生登录App |
| 标识符 | PC2/DC2 |
| 角色 | 患者/医生 |
| 前置事件流 | 前置事件流：用户进入登录页面 |
| 主事件流 | 1. 用户进入登录页面 2. 用户填写账号和密码 3. 用户选择登录身份 4. 用户点击登录按钮 5. 若信息未填写完整，提示未填写完整 6. 若信息填写完整，交由App处理 7. 用例结束 |
| 其他事件流 | 其他事件流：  1. 网络异常，用户登录失败 |
| 后置事件流 | 后置事件流：  1. 用户以患者或医生身份进入App操作 |

**表2-3 预约挂号用例描述**

|  |  |
| --- | --- |
| 用例名称 | 预约挂号 |
| 描述 | 患者用户在医院服务模块进行预约挂号 |
| 标识符 | PC3 |
| 角色 | 患者 |
| 前置事件流 | 前置事件流：患者成功登录App，并进入医院服务模块 |
| 主事件流 | 1. 患者进入医院服务模块 2. 患者选择对应医院 3. 患者选择预约挂号 4. 患者选择对应科室和时间 5. 患者点击确认预约按钮 6. 若预约信息有误，提示信息有误 7. 若预约信息正确，交由App处理 8. 用例结束 |
| 其他事件流 | 其他事件流：  1. 网络异常，预约失败 |
| 后置事件流 | 后置事件流：  1. 患者可以在界面中查看预约详细信息 |

**表2-4 购买药品用例描述**

|  |  |
| --- | --- |
| 用例名称 | 购买药品 |
| 描述 | 患者在药品购买配送模块，选择购买药品 |
| 标识符 | PC4 |
| 角色 | 患者 |

续表2-4

|  |  |
| --- | --- |
| 前置事件流 | 前置事件流：患者成功登录App，并进入药品购买模块 |
| 主事件流 | 1. 患者进入药品购买配送模块 2. 患者查找需要购买的药品 3. 患者点击购买按钮 4. 患者付款 5. 若付款失败，提示购买失败 6. 若付款成功，提示购买成功 7. 用例结束 |
| 其他事件流 | 其他事件流：  1. 网络异常，购买药品失败 |
| 后置事件流 | 后置事件流：  1. 患者可以查看购买的药品配送情况 |

**表2-5 辅助医疗诊断用例描述**

|  |  |
| --- | --- |
| 用例名称 | 辅助医疗诊断 |
| 描述 | 医生在医疗诊断模块进行辅助医疗诊断 |
| 标识符 | DC1 |
| 角色 | 医生 |
| 前置事件流 | 前置事件流：医生成功登录App，并进入医疗诊断模块 |
| 主事件流 | 1. 医生进入医疗诊断模块 2. 医生选择病人症状 3. 医生点击确定按钮 4. 若症状信息有误，提示信息有误 5. 若症状信息无误，提示提交成功 6. 用例结束 |
| 其他事件流 | 其他事件流：  1. 网络异常，症状信息提交失败 |
| 后置事件流 | 后置事件流：  1. App显示病人的症状分析报告 |

### 2.2.3 系统非功能需求

（1）性能需求

正常情况下，用户操作响应时间不超过3秒，多用户访问或网络不好时，操作响应时间不超过5秒。

（2）兼容性需求

App支持Android系统，并且需要对较低版本的系统进行兼容适配。

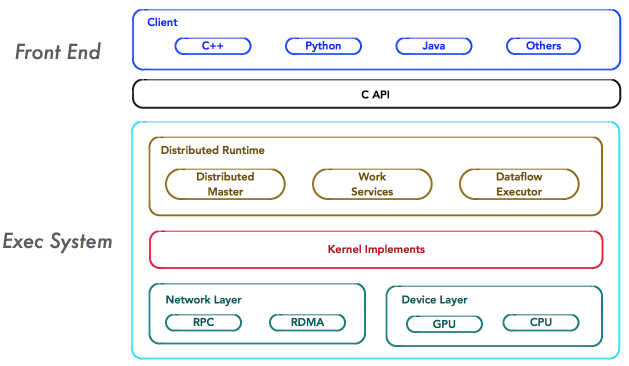
（3）交互性需求

考虑到使用手机App的用户可能是对手机终端操作不熟悉的用户，软件将使用简洁的用户界面，将用户最为需要的功能并入一条完整的服务链，使得用户使用软件变得更为简便。同时App界面设计风格应保持一致，对于异常事件或操作错误应给出醒目的警告或友善的提示。

## 2.3 方案比选

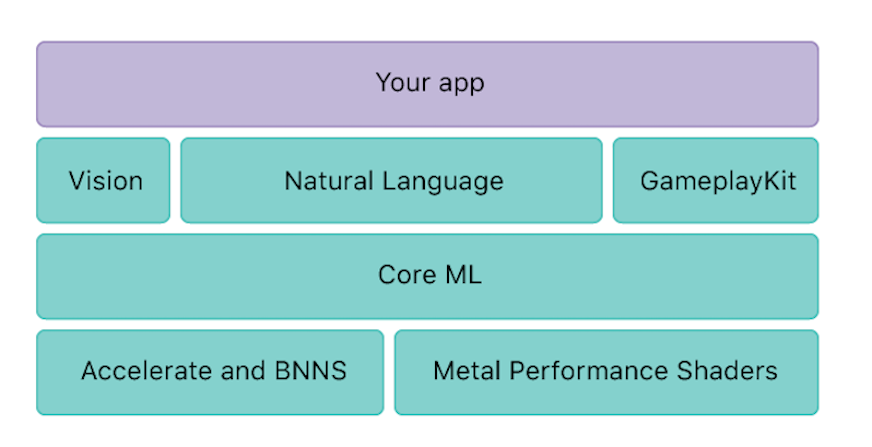
近年来，随着智能手机和移动互联网的迅速发展，手机操作系统之间的竞争越来越激烈，大部分操作系统都逐渐退出了历史的舞台，只剩下Android和iOS占据着市场的极大份额[6]。目前比较流行的机器学习App开发框架主要是TensorFlow和Core ML，两者是分别基于不同的手机操作系统开发出来技术框架。下面将对TensorFlow框架和Core ML框架两种方案进行比较，并给出优劣势和比选结果。

### 2.3.1 方案一：Android + TensorFlow

**图 2-3 TensorFlow系统架构图**

TensorFlow是基于数据流编程的符号数学系统，它具有多层结构，并广泛用于各种机器学习算法的编程中。TensorFlow的系统结构是以C API为界，使其分为前端和后端两个子系统，前端系统管理编程模型，作用于构造计算图，后端系统管理运行时环境，用于执行计算图。前端系统中Client支持多种编程语言，可以实现各种形式的模型设计，其通过session与后端的运行时相连接。Distributed Master根据Fetching参数，找到并启动执行子图片段。Kernel是OP在某种硬件设备的特定实现，主要负责执行OP的运算。

### 2.3.2 方案二：iOS + Core ML

**图 2-4 Core ML层级结构图**

Core ML是Apple的机器学习框架，Core ML能够将数据经过预处理后输入ML model文件，输出为模型的预测结果。Core ML可以将各种机器学习模型类型集成到应用程序中，除了支持超过30种类型的广泛深度学习外，它还支持标准模型，例如树状集成体，SVM和广义线性模型。 由于它基于Metal和Accelerate等底层技术构建，因此Core ML无缝地利用了CPU和GPU的优势，以提供最佳的性能和效率。

### 2.3.3 比选结论

TensorFlow和Core ML两者之间有着很大的区别，其中TensorFlow是包含了训练模型和评估模型，Core ML只支持在设备上评估模型，不能训练模型。因此，TensorFlow更偏向数据训练层，Core ML本身的API只是借助已经训练好的数据模型，最后生成对应的包装类，接受输入、输出的结果值。在操作系统方面，Android更为开放，拥有庞大的用户量，平台发展迅速，市场应用也很丰富，但版本过多、升级较快，用户信息和安全等方面也存在一定的隐患。iOS整个生态系统比较好，比较容易获取利益，应用质量也较高，但系统生态封闭，难于扩大市场，开发门槛也较高，需要熟悉Objective C语言和苹果开发环境。

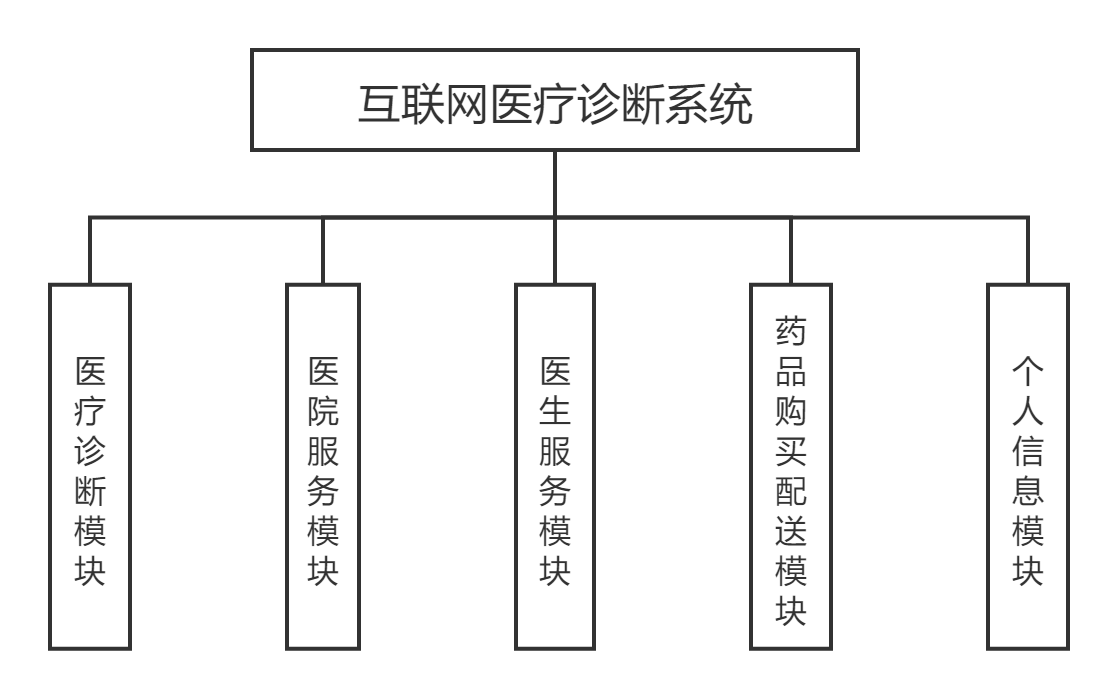
通过以上分析，可以看出Core ML在机器学习领域技术不够完善，同时，iOS生态也不适合本应用运营前期需要庞大的用户数据量来训练、改进医疗诊断模型，因此本系统采用Android + TensorFlow框架开发。

## 2.4 本章小结

本章首先对本课题进行可行性分析和需求分析，再根据课题的技术和生态需求，对两种方案进行优劣势分析，并最终确定方案。

# 3 系统的设计

## 3.1 软件体系结构



**图3-1 系统功能结构图**

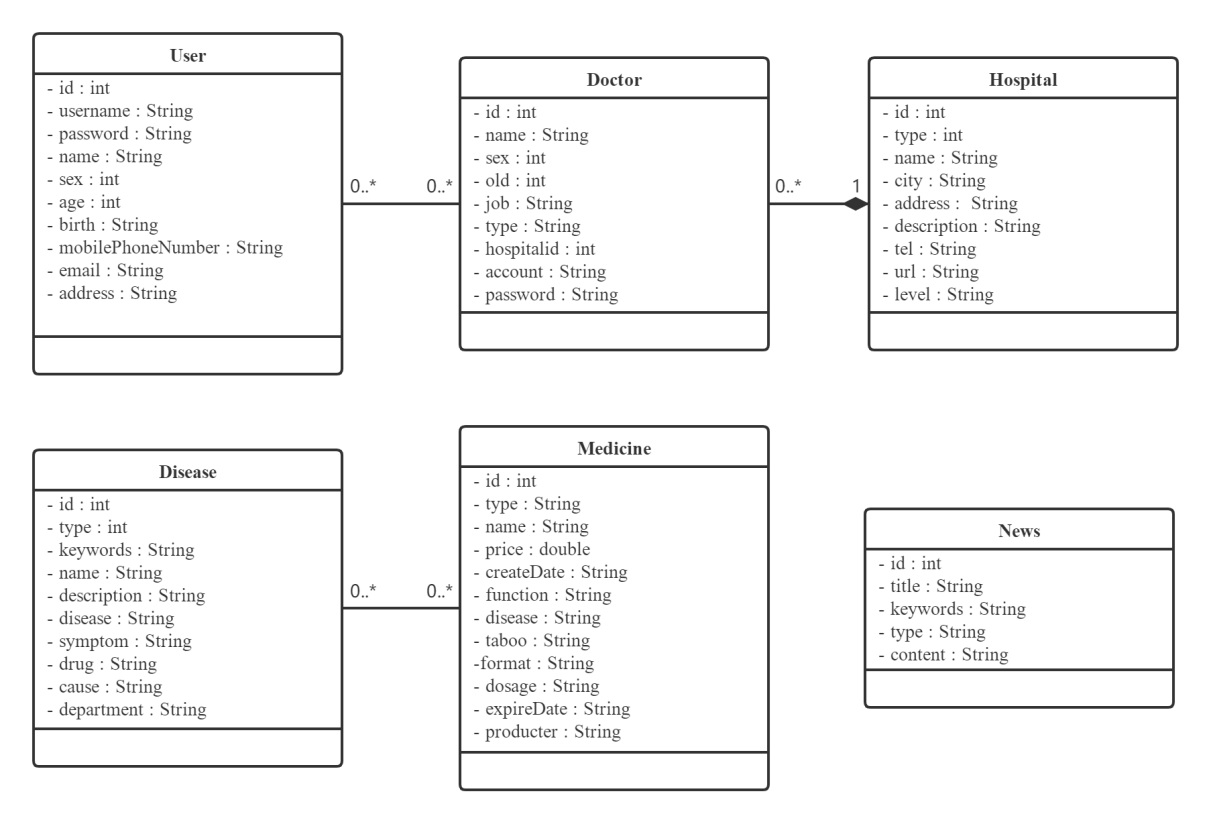
图3-1展示的是互联网医疗诊断系统的五个功能模块，五个功能模块之间相互独立，负责的业务逻辑也不相同。但是前四个功能模块也同时形成了一条完整的医疗服务链，从看病诊断到医院就医，再到咨询医生确定所患疾病，然后最后是购买药品进行治疗，第五个模块是系统所需的模块，方便用户管理个人信息和查看医疗相关记录。

本系统的核心功能模块是医疗诊断模块，通过TensorFlow在App上实现了基于机器学习的医疗诊断功能。患者用户通过根据身体实际情况填写症状表格，可以进行小病或中病的自诊，医生用户也可以借助其功能辅助医疗诊断病人。

## 3.2 功能设计

（1）类图设计

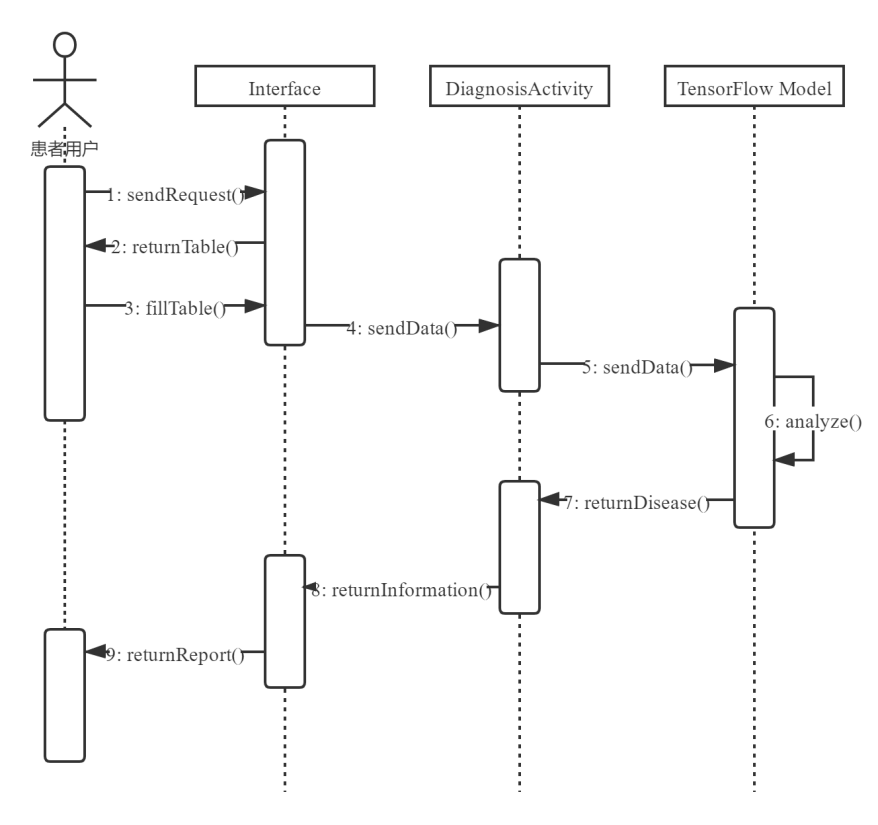
图3-2介绍了互联网医疗诊断系统中主要实体类和类之间的关系，主要实体类分别为User、Hospital、Doctor、Medicine、Disease以及News，医院和医生是一对多的关系，医院里可以有多个医生，但一个医生只能属于某个医院。患者和医生是多对多的关系，患者可以问诊多个医生，医生也可以诊断多个患者。同理，疾病和药品也是多对多的关系。



**图3-2 系统类图**

（2）时序图设计

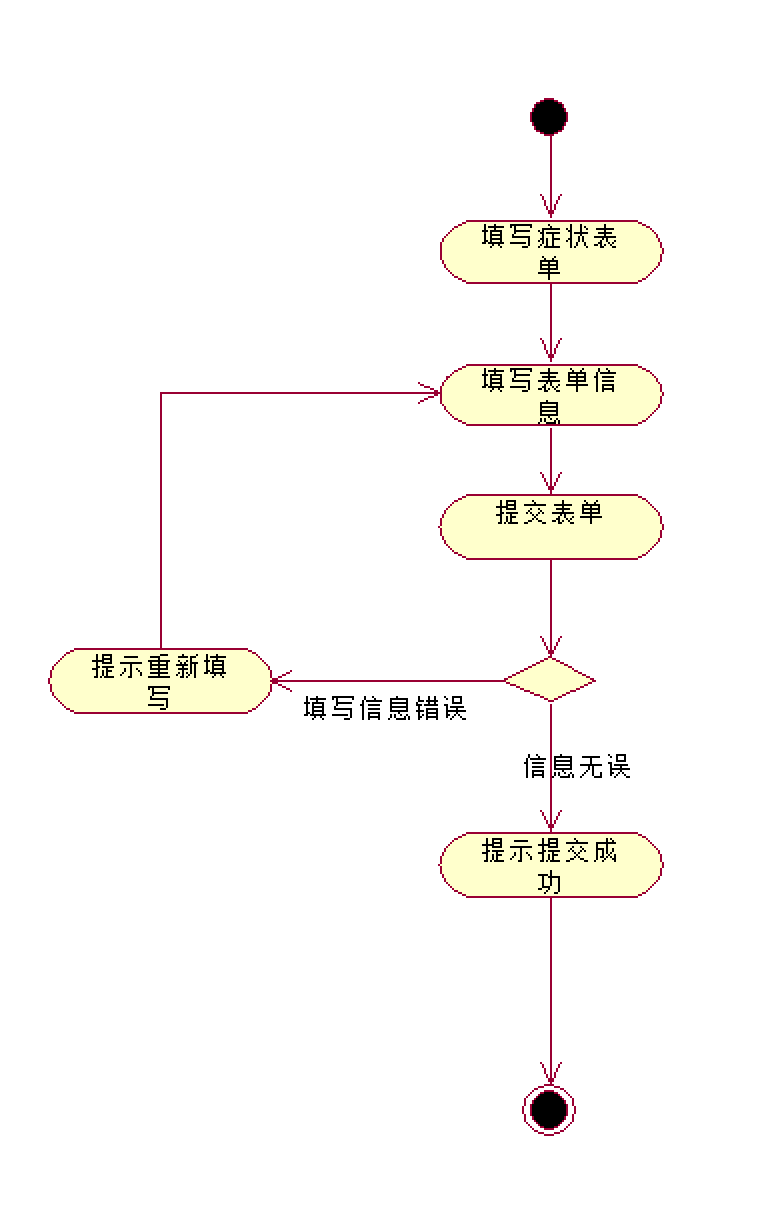
图3-3展示了患者用户自诊功能的时序图，患者用户首先选择自主诊断，会调用自诊实现方法，系统界面反馈显示症状表格，患者按身体实际情况填写完症状表格后，点击提交按钮，系统会启动诊断模型的方法，对患者的具体症状进行分析，最后得出患者可能患得的疾病，并在系统界面以报告形式展现给患者用户。



**图3-3 自诊功能时序图**

（3）活动图设计

图3-4是本系统的自诊功能活动图，患者用户首先进入自诊界面，界面会显示需要填写的症状表格，患者用户如实填写完表格后，点击提交表格后，系统会检查表格是否填写有误和是否填写完整，若未填写完善，将提示用户继续填写；若填写完整，系统会提示提交成功，并经过小段时间分析后，会在界面上展示患者用户的患病报告。

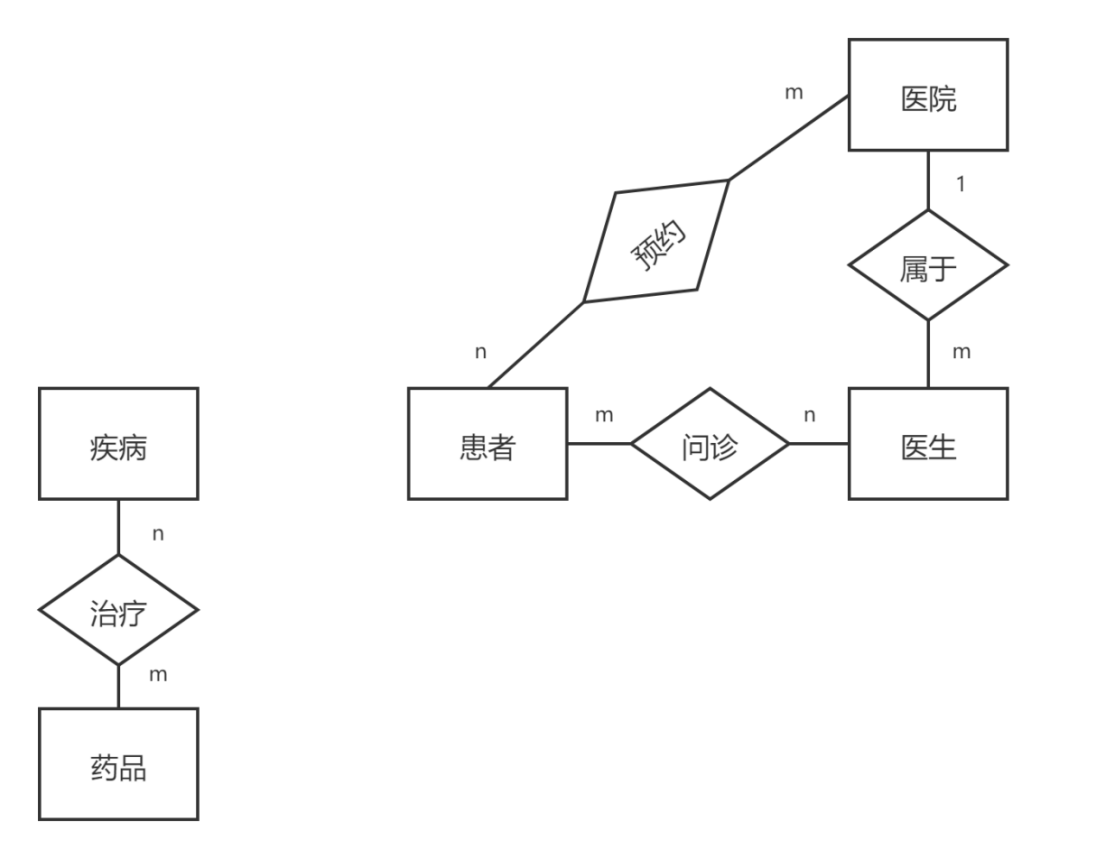


**图3-4 自诊功能活动图**

## 3.3 持久化设计

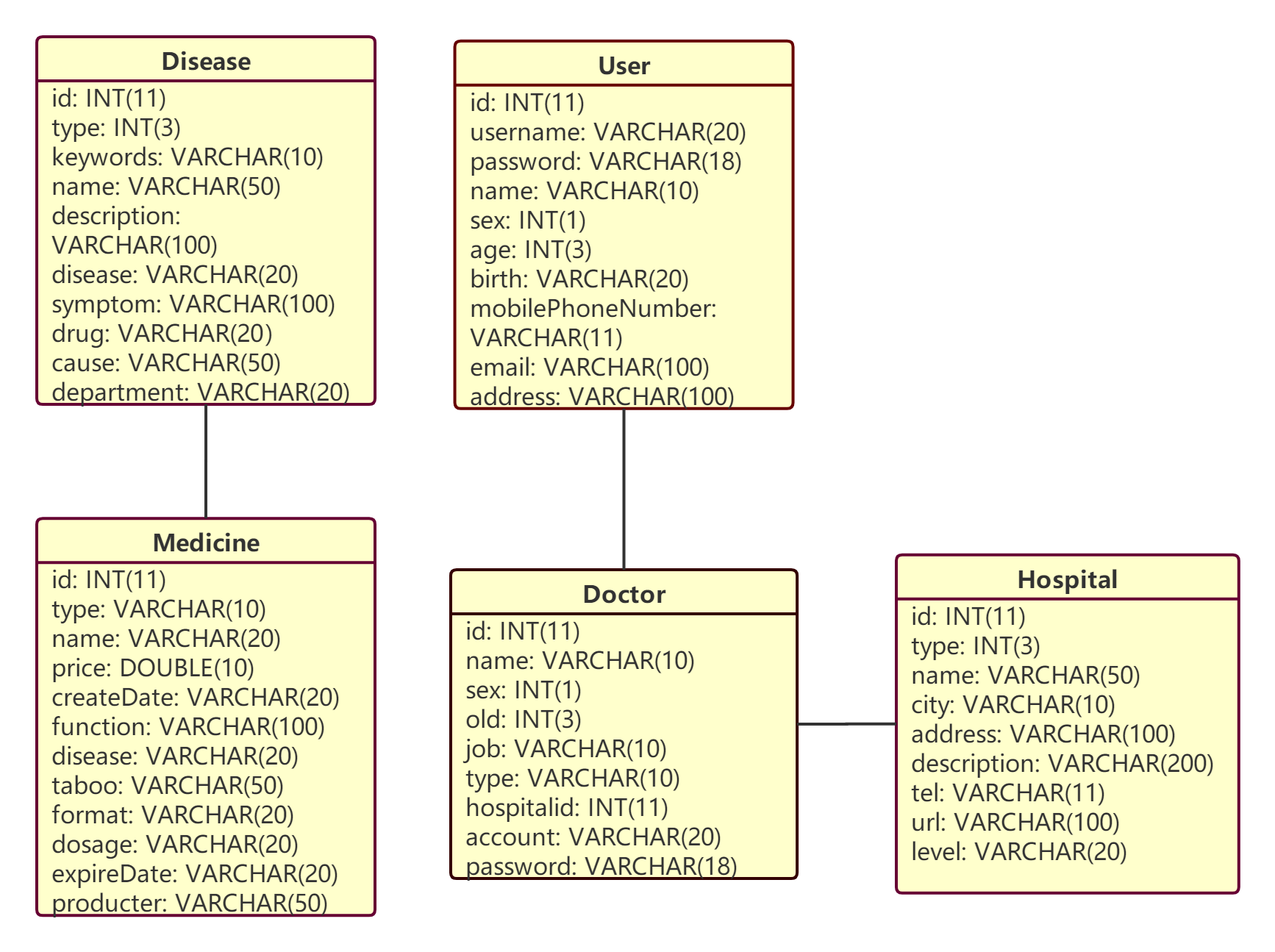
### 3.3.1 数据库逻辑关系

图3-5展现了患者、医生、医院、疾病和药品五个实体的属性和联系，患者和医生是多对多关系，患者可以咨询多个医生，医生也可以诊断多个患者。医院和医生是一对多的关系，医院可以有多个医生，但一个医生只能属于某个医院。疾病和药品之间是多对多关系，一种疾病可以用多个药品治疗，一款药品也可以治疗多个疾病。



**图3-5 E-R图**

### 3.3.2 数据库表设计



**图3-6 数据库表图**

数据库作为系统运行的后台，其设计对系统的业务流程和数据交互都至关重要，因此数据库的设计需要简便合理，并且满足本课题的基本需求。本课题采用的数据库是Bmob云端数据库，Bmob作为一个移动应用的后台管理系统，其数据操作简单易学，数据存储也足够安全，并且还拥有丰富的数据类型。Bmob在很大程度上减少了后期维护和租借服务器的精力，节省了很多成本，同时也解决了开发中后端逻辑重复开发的问题。因此，Bmob云数据库适合作为本课题移动App的后台管理系统。

具体数据库表的信息、约束和字段描述如下：

（1）患者信息表

**表3-1 患者信息表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段长度 | 约束 | 字段描述 |
| id | int | 11 | PK | 用户编号 |
| username | varchar | 20 | not null | 用户名 |
| password | varchar | 18 | not null | 密码 |
| name | varchar | 10 | not null | 姓名 |
| sex | int | 1 | not null | 性别 |
| old | int | 3 | not null | 年龄 |
| birth | varchar | 20 | null | 生日 |
| mobilePhoneNumber | varchar | 11 | not null | 电话 |
| email | varchar | 100 | null | 电子邮箱 |
| address | varchar | 100 | not null | 地址 |

（2）医生信息表

**表3-2 医生信息表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段长度 | 约束 | 字段描述 |
| id | int | 11 | PK | 医生编号 |
| name | varchar | 10 | not null | 姓名 |
| sex | int | 1 | not null | 性别 |
| old | int | 3 | not null | 年龄 |
| job | varchar | 10 | not null | 职位 |
| type | varchar | 10 | not null | 科室 |
| hospitalid | int | 11 | FK | 所属医院编号 |
| account | varchar | 20 | not null | 用户名 |
| password | varchar | 18 | not null | 密码 |

（3）医院信息表

**表3-3 医院信息表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段长度 | 约束 | 字段描述 |
| id | int | 11 | PK | 医院编号 |
| type | int | 3 | not null | 类别 |
| name | varchar | 50 | not null | 医院名称 |
| city | varchar | 10 | not null | 所在城市 |
| address | varchar | 100 | not null | 医院地址 |

续表3-3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| description | varchar | 200 | null | 医院介绍 |
| tel | varchar | 11 | not null | 联系电话 |
| url | varchar | 100 | not null | 官方网址 |
| level | varchar | 20 | Not null | 所属级别 |

（4）疾病信息表

**表3-4 疾病信息表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段长度 | 约束 | 字段描述 |
| id | int | 11 | PK | 疾病编号 |
| type | int | 3 | not null | 类型 |
| keywords | varchar | 10 | null | 关键词 |
| name | varchar | 50 | not null | 疾病名称 |
| description | varchar | 100 | not null | 疾病描述 |
| disease | varchar | 20 | null | 相关疾病 |
| symptom | varchar | 100 | not null | 症状 |
| drug | varchar | 20 | null | 治疗药物 |
| cause | varchar | 50 | not null | 病因 |
| department | varchar | 20 | null | 所属科室 |

（5）药品信息表

**表3-5 药品信息表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段长度 | 约束 | 字段描述 |
| id | int | 11 | PK | 药品编号 |
| type | varchar | 10 | null | 类型 |
| name | varchar | 20 | not null | 药品名称 |
| price | double | 10 | not null | 价格 |
| createDate | varchar | 20 | not null | 生产日期 |
| function | varchar | 100 | not null | 功效 |
| disease | varchar | 20 | null | 治疗疾病 |
| taboo | varchar | 50 | null | 禁忌人群 |
| format | varchar | 20 | not null | 规格 |
| dosage | varchar | 20 | not null | 剂量 |
| expireDate | varchar | 20 | not null | 有效期 |
| producter | varchar | 50 | null | 生产商 |

## 3.4 社会健康、文化、法律相关设计

在社会健康方面，该系统的自我诊断功能保证只有在满足高精度要求的情况下才投放市场，并秉承“小疾病的预防，大病的建议”的概念。以及严重疾病的医疗”。在文化方面，该系统中的所有信息均符合社会主义的核心价值观，不包含宗教信仰之类的信息，并且不分发或出售任何非法内容。该系统中使用的集成开发环境（IDE）由教育版本授权。所使用的机器学习模型和医学数据集也是开源材料。系统的设计和实现由我独立完成。本系统中使用的患者和医生信息是匿名数据，没有故意泄露他人的私人信息，并且符合法律法规的要求。

## 3.5 本章小结

本章是对互联网医疗App的系统设计进行详细描述，首先通过分析系统的功能需求将系统划分为了五个功能模块，再对各个模块进行功能设计和持久化设计，给出了类图、时序图、活动图、E-R图和数据库表，最后说明了系统设计在社会健康、文化、法律方面上的影响。

# 4 系统的实现

## 4.1 医疗诊断

机器学习技术可以通过自动学习数据，并从中获取信息和特征，从而做出准确的判断。基于机器学习的医疗诊断模型可以对大量的诊断数据进行学习，使得计算机拥有对新患者诊断预测疾病的能力，降低传统诊断模式的误诊率。同时，传统的机器学习方法已经不再适用于学习越来越复杂的疾病数据，因此集成学习成为一种新的受欢迎的机器学习技术[15]。

### 4.1.1 数据来源及预处理

本系统所采用的数据集为Kaggle官方网站的开源数据集（https://www.kaggle.com/iamhungundji/covid19-symptoms-checker），其中包含了新冠肺炎可能的患者的个人信息和相关症状信息。

初始数据信息可分为患者信息和诊断结果，患者信息是患者的一些可能影响到疾病诊断的个人信息，诊断结果则为患者是否感染新冠肺炎的信息，包括数据ID、性别、年龄、肺炎相关症状以及诊断结果。

**表4-1 数据集样例**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Id | 性别 | 年龄 | 发烧 | 身体疲惫 | 干咳 | 呼吸困难 | 喉咙痛 | 是否感染 |
| 1 | 1 | 10-19 | 1 | 0 | 1 | 0 | 1 | 1 |
| 2 | 1 | 60+ | 1 | 1 | 0 | 0 | 1 | 1 |
| 3 | 2 | 20-24 | 0 | 0 | 1 | 1 | 1 | 0 |
| 4 | 1 | 25-59 | 0 | 0 | 1 | 1 | 1 | 1 |
| 5 | 2 | 0-9 | 1 | 1 | 1 | 1 | 1 | 1 |

由于从网络中收集到的数据集可能存在大量的错误数据和冗余信息，因此获取到数据首先要对其进行预处理和清洗操作。根据对数据的初步了解和作可视化比较，发现患者可能存在对同一个症状描述情况不一样的问题，比如对病情程度不熟悉，或者对症状描述不同等情况，这会导致症状和疾病的关系发生不对应的情况。因此本系统采取数据抽取的方式，首先确定好数据保留的特征，此外为了保证爬取数据的质量，应参考以下几个原则[16]：

1. 用于训练和测试的数据应具有一定的数量，这样才能够保证医疗诊断模型得到足够的训练。
2. 数据应该保留真实性，保留可能会影响到诊断的个人相关信息，比如性别以及年龄。
3. 数据信息应该足够丰富，含有跟诊断相关的重要信息，包括症状信息、发生症状的部位以及判断的疾病结果。

### 4.1.2 诊断模型算法

（1）实现介绍

基于机器学习的医疗诊断模型用到的核心算法是分类算法，常用的分类算法有支持向量机、Logistic回归、决策树和K近邻算法等，同时还有将多种分类算法结合在一起的集成学习算法，其中就包含AdaBoost、GDBT、Boosting等算法[16]。本系统使用TensorFlow技术实现逻辑回归算法，从而构造出本系统的诊断模型。

经过预处理后的数据集共包含198000条数据，将其前70%作为训练集，后30%作为测试集，首先对其进行归一化处理，使得逻辑回归效果更好，使用的归一化公式如下：

（4.1）

xmax和xmin分别代表当前列的最大值和最小值，xtrain即该条数据在当前列的取值，经过归一化处理后，数据取值都在0和1之间，使得逻辑回归训练效果更好。使用的预测公式如下：

（4.2）

y作为诊断结果，0为未感染，1为感染，A代表7个输入的特征变量的权重矩阵，b作为预测的偏值，A和b也是验证诊断模型是否准确的重要变量。

1. 实现代码

# 将数据分为训练集（70%）和测试集（30%）

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x\_data, y\_data, test\_size=0.3)

def normalize\_cols(m):

col\_max = m.max(axis=0)

col\_min = m.min(axis=0)

return (m-col\_min) / (col\_max - col\_min)

# 对数据做归一化处理

x\_train = np.nan\_to\_num(normalize\_cols(x\_train))

x\_test = np.nan\_to\_num(normalize\_cols(x\_test))

# 设置训练变量

A = tf.Variable(tf.random.normal(shape=[7, 1]))

b = tf.Variable(tf.random.normal(shape=[1, 1]))

# 定义模型所需参数

model\_output = tf.add(tf.matmul(feature, A), b)

loss = tf.reduce\_mean(tf.nn.sigmoid\_cross\_entropy\_with\_logits(logits=model\_output, labels=label))

my\_opt = tf.compat.v1.train.GradientDescentOptimizer(0.1)

train\_step = my\_opt.minimize(loss)

# 训练部分

for i in range(200):

sess.run(train\_step, feed\_dict={feature: x\_train, label: y\_train})

temp\_loss = sess.run(loss, feed\_dict={feature: x\_train, label: y\_train})

loss\_vec.append(temp\_loss)

temp\_acc\_train = sess.run(accuracy, feed\_dict={feature: x\_train, label: y\_train})

train\_acc.append(temp\_acc\_train)

print(sess.run(A))

print(sess.run(b))

temp\_acc\_test = sess.run(accuracy, feed\_dict={feature: x\_test, label:y\_test})

test\_acc.append(temp\_acc\_test)

print('Epoch: {}, Loss: {}'.format(i, temp\_loss))

print('Accuracy: {}'.format(temp\_acc\_train))

### 手机屏幕截图 描述已自动生成4.1.3 训练结果

**地图的截图

描述已自动生成图4-1 损失值曲线**

**图4-2 准确率曲线**

## 4.2 用户操作管理

本系统将用户的操作封装到Bmob云函数进行，包括登录、注册、修改密码以及更新用户信息，提高了用户信息的安全性，防止用户的隐私发生泄露。

### 4.2.1 用户状态

（1）实现介绍

Bmob云数据库有自带的BmobUser类，用于实现用户操作和用户管理，创建数据库时会自动化生成 User表，其中用来存储系统所有用户的信息，方便后台管理员对用户信息进行操作。由于BmobUser类自带登录、登出的方法，分别是loginin( )和loginOut( )，所以在处理用户登录、登出以及查看用户状态时只需要调用相应的方法即可，不需要在Activity和Fragment之间用Intent反复传递信息，具体实现代码如下。（本系统User类继承自BmobUser类）

（2）实现代码

// 登录事件

private void signIn(final View view) {

final User user = new User();

user.setUsername(username);

user.setPassword(password);

// 调用BmobUser的登录方法

user.login(new SaveListener<User>() {

@Override

public void done(User bmobUser, BmobException e) {

if (e == null) {

User user = BmobUser.getCurrentUser(User.class);

Toast.makeText(LoginActivity.this, "登录成功：" + user.getUsername(), Toast.LENGTH\_SHORT).show();

Intent intent = new Intent(LoginActivity.this, MainActivity.class);

intent.setFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(intent);

finish();

} else {

Toast.makeText(LoginActivity.this, "登录失败：" + e.getMessage(), Toast.LENGTH\_SHORT).show();

}

}

});

}

// 注册事件

private void signUp(final View view) {

final User user = new User();

user.setUsername(username);

user.setPassword(password);

user.setEmail(email);

// 调用BmobUser的注册方法

user.signUp(new SaveListener<User>() {

@Override

public void done(User user, BmobException e) {

if (e == null) {

Toast.makeText(RegisterActivity.this, "注册成功", Toast.LENGTH\_SHORT).show();

} else {

Toast.makeText(RegisterActivity.this, "注册失败：" + e.getMessage(), Toast.LENGTH\_SHORT).show();

}}});}

// 查看用户登录状态

if (BmobUser.isLogin()) {

role = (int) BmobUser.getObjectByKey("role");

// 获取当前用户

User user = BmobUser.getCurrentUser(User.class);

if (role == 1) {

Toast.makeText(this, "用户好，您已经登录：" + user.getUsername(), Toast.LENGTH\_LONG).show();

} else {

Toast.makeText(this, "医生好，您已经登录：" + user.getUsername(), Toast.LENGTH\_LONG).show();

}

} else {

Toast.makeText(this, "您还未登录" , Toast.LENGTH\_LONG).show();

}

### 4.2.2 修改重置密码

（1）实现介绍

BmobUser类对于用户已登录状态修改密码，和未登录状态重置密码都提供了相应的方法。对于登录状态修改密码，只需要调用updateCurrentUserPassword( )，向其传入旧密码和新密码即可，对于未登录状态重置密码，用户需要输入用户名和邮箱，若信息正确，则用户可以依靠验证邮件重置密码，使得开发人员对用户管理变得更简单。

（2）实现代码

// 用户登录状态修改密码

BmobUser.updateCurrentUserPassword(password1, password2, new UpdateListener() {

@Override

public void done(BmobException e) {

if (e == null) {

Toast.makeText(ResetpwdActivity.this, "修改密码成功", Toast.LENGTH\_SHORT).show();

finish();

} else {

Toast.makeText(ResetpwdActivity.this, "修改密码失败：" + e.getMessage(), Toast.LENGTH\_SHORT).show();

}

}

});

// 用户未登录状态重置密码

bmobQuery.findObjects(new FindListener<User>() {

@Override

public void done(List<User> list, BmobException e) {

// 如果查询有结果，则邮箱和用户名对应，否则不对应

if (e == null) {

// 邮箱重置密码

User user = list.get(0);

String userEmail = user.getEmail();

BmobUser.resetPasswordByEmail(userEmail, new UpdateListener() {

@Override

public void done(BmobException e) {

if (e == null) {

Toast.makeText(ForgetActivity.this, "重置密码请求成功，请到" + email + "邮箱进行密码重置操作", Toast.LENGTH\_SHORT).show();

} else {

Toast.makeText(ForgetActivity.this, "重置密码请求失败！" + e.getMessage(), Toast.LENGTH\_SHORT).show();

}}});

} else {

Toast.makeText(ForgetActivity.this, "邮箱和用户名不对应！", Toast.LENGTH\_SHORT).show();

}}});

## 4.3 医院服务

### 4.3.1 预约挂号

（1）实现介绍

用户可以在医院服务模块选择预约挂号功能，用户需要依次选择医院、科室、预约日期，然后再调用Bmob的SaveListener将预约信息存入数据库。其中由于Android自带的CalendarView只能返回年月日，还需要SimpleDateFormat将用户选择的预约时间转为Date类型。此外预约过程需要经过几个Activity来确认预约信息，因此需要通过Intent将预约信息传递到下一个操作的Activity。

（2）实现代码

// Intent在Activity之间传递预约信息

intent.putExtra("department", department);

intent.putExtra("years", years);

intent.putExtra("months", months);

intent.putExtra("days", days);

intent.putExtra("date", date);

intent.setClass(this, ReservationChooseActivity.class);

// 从Calendar获取预约时间

calendar.setOnDateChangeListener(new CalendarView.OnDateChangeListener() {

@Override

public void onSelectedDayChange(CalendarView view, int year, int month, int dayOfMonth) {

//显示用户选择的日期

years = year;

month++;

months = month;

days = dayOfMonth;

date = year + "年" + month + "月" + dayOfMonth + "日";

}

});

// 将String转换成Date类型

String date = year + "-" + month + "-" + day;

SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd");

time = simpleDateFormat.parse(date);

BmobDate bmobDate = new BmobDate(time);

// 添加预约信息

reservation.save(new SaveListener<String>() {

@Override

public void done(String s, BmobException e) {

if(e==null){

Toast.makeText(ReservationConfirmActivity.this, "预约成功！", Toast.LENGTH\_SHORT).show();

}else{

Toast.makeText(ReservationConfirmActivity.this, "预约失败：" + e.getMessage(), Toast.LENGTH\_SHORT).show();

}

}

});

### 4.3.2 医院导航

（1）实现介绍

本系统的医院导航功能采用了百度地图API接口，通过下载其SDK包和配置App key得以使用导航功能。系统采用将医院的经纬度存入数据库，当用户选择好医院，打开导航功能，系统将医院的经纬度信息传入百度地图用于设置中心位置，这样用户就可以了解到目标医院的具体位置和周围环境。

（2）实现代码

// xml布局文件中调用百度地图组件

<com.baidu.mapapi.map.MapView

android:id="@+id/map"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

MapView mapView = (MapView) findViewById(R.id.map);

BaiduMap mBaidumap = mapView.getMap();

// 设置目标医院的经纬度

LatLng cenpt = new LatLng(latitude, longtitude);

// 定义地图状态

MapStatus.Builder builder = new MapStatus.Builder();

builder.target(cenpt).zoom(18.0f);

// 定义MapStatusUpdate对象，以便描述地图状态将要发生的变化

mBaidumap.animateMapStatus(MapStatusUpdateFactory.newMapStatus(builder.build()));

## 4.4 医疗信息显示和查询

### 4.4.1 信息显示

（1）实现介绍

布局文件使用ListView将疾病、医院、药品信息显示出来，需要为每个类编写Adapter，从而将对应信息放置到item布局文件里，同时还需要Fragment或Activity从Bmob数据库中查询信息，并为ListView设置适配器，以下代码以疾病信息作为例子。

（2）实现代码

// 设置疾病的ViewHolder，用于对item布局进行管理

class ViewHolder {

TextView name;

TextView department;

TextView cause;

TextView symptom;

TextView description;

}

// 设置item布局信息

viewHolder.name.setText(dis.getName());

viewHolder.department.setText(dis.getDepartment());

viewHolder.cause.setText(dis.getCause());

viewHolder.symptom.setText(dis.getSymptom());

viewHolder.description.setText(dis.getDescription());

private void initData() {

disList.clear();

// 查询疾病数据

BmobQuery<Disease> query = new BmobQuery<>();

query.findObjects(new FindListener<Disease>()

@Override

public void done(List<Disease> list, BmobException e) {

if (e == null) {

for (int i=0; i<list.size(); i++) {

Disease dis = list.get(i);

// 向ArrayList添加查询到的疾病信息

disList.add(dis);

adapter = new DiseaseAdapter(getActivity(), disList);

// 为ListView设置疾病适配器

disease\_view.setAdapter(adapter);

}}}});}

### 4.4.2 信息查询

（1）实现介绍

通过在EditText中添加文本修改监听器，用户输入文字，就会触发监听器，从而对数据进行条件查询，再重新显示ListView的信息，同时ListView的高度也会随信息数量变化。

（2）实现代码

// 给搜索框添加监听事件

search\_disease.addTextChangedListener(new TextWatcher() {

@Override

public void onTextChanged(CharSequence s, int start, int before, int count) {

String name = s.toString();

changeClient(name);

}}

public void changeClient(String name) {

// 根据输入条件筛选疾病数据

disList.clear();

BmobQuery<Disease> query1 = new BmobQuery<>();

if (!name.equals("")) {

query1.addWhereEqualTo("keywords", name);

}

query1.findObjects(new FindListener<Disease>() {

@Override

public void done(List<Disease> list, BmobException e) {

if (e == null) {

for (int i=0; i<list.size(); i++) {

Disease dis = list.get(i);

disList.add(dis);

adapter = new DiseaseAdapter(getActivity(), disList);

disease\_view.setAdapter(adapter);

setListViewHeight(disease\_view);

adapter.notifyDataSetChanged();

}}}});}

// ListView自适应变化高度

private void setListViewHeight(ListView listView) {

if (adapter == null) {

return;

}

View itemView = adapter.getView(0, null, listView); // 获取其中一项

itemView.measure(0,0);

int itemHeight = itemView.getMeasuredHeight(); // 获取其中一项的高度

int itemCount = adapter.getCount(); // 获取总得项数

int height = itemHeight \* itemCount;

ViewGroup.LayoutParams params = listView.getLayoutParams();

params.height = height;

}

## 4.5 本章小结

本章主要介绍了系统部分功能的实现过程，并配以代码和文字进行讲解。

# 5 系统运行与效果分析

## 5.1 界面设计概要

本系统希望为用户提供简洁一致的用户界面，去除掉多余的按键和功能，使用户操作变得更为简便，并且将业务流程分为五个功能模块，它们相互独立的同时也提供了完整的服务链。用户在使用App时，系统对于错误操作有着友好且详细的提示，可以提高用户的使用体验。

## 5.2 用户管理

### 5.2.1 用户登录注册

系统仅面向注册用户服务，访客可以在个人模块点击头像进入登录界面，如果还没有账号，可以进行在线注册。系统对于用户名、密码和邮箱都有格式限制，当用户输入格式有误时，系统将给以友好的提示，提醒用户重新输入。用户实现登录注册功能是通过Bmob自带的SignIn ( )和SignUp( )方法，因此可以很好地保障用户信息的安全。



**图5-1 用户登录界面**



**图5-2 用户注册界面**

### 5.2.2 用户信息管理

系统为用户提供了查看个人信息、修改密码以及忘记密码等操作，用户可以通过查看个人信息和修改密码很好地管理个人信息，并且修改密码会核对旧密码是否正确。用户忘记密码，可以通过向系统请求重置密码，系统会向用户注册时的邮箱发送验证邮件，用户可以通过邮件的链接来修改密码，从而防止用户信息泄露。

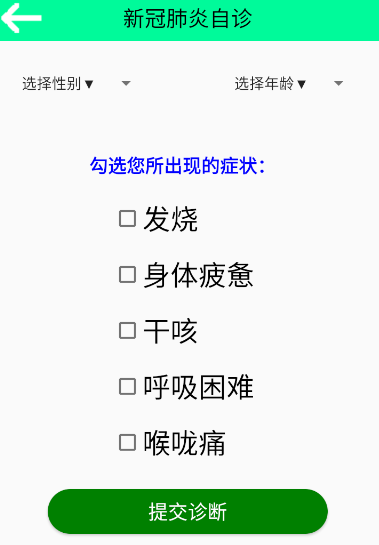
**图5-3 个人信息界面**



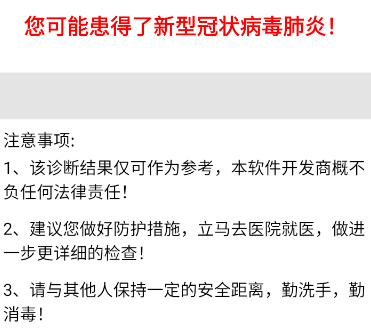
**图5-4 密码管理界面**

## 5.3 诊断购药

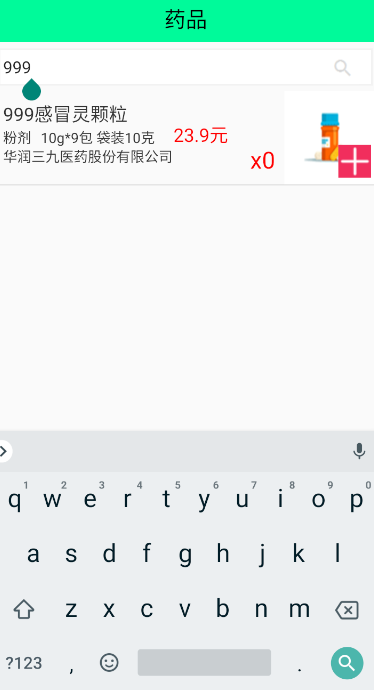
### 5.3.1 疾病自诊



**图5-5 疾病自诊界面**

**图5-6 诊断结果界面**

如图5-5和图5-6所示，用户首先根据个人信息和身体情况选择性别、年龄和身体出现的症状，点击提交诊断按钮，系统将对用户症状信息进行分析，得出用户患得新型冠状病毒肺炎的概率，并转到对应的诊断结果界面。

5.3.2 药品购买

**图5-7 药品购买界面**

如图5-7所示，用户点击药品购买模块，可以在界面中看到药品的详细信息，通过点击添加按钮，可以直接将药品添加到购物车中，也可以通过搜索栏查询所需要的药品，用户点击结算按钮便可直接生成订单。

## 5.4 医院医生服务

### 5.4.1 医院服务

**图5-8 医院服务界面**

如图5-8所示，用户可以在医院服务模块查看医院的详细信息，并且可以通过搜索栏查询对应的医院，界面也会显示医院的地址和联系电话等重要信息，方便用户及时就医。用户还可以通过医院导航功能，查看医院具体位置和周边情况，必要时可以通过App启动地图导航即刻前往。





**图5-9 医院预约界面**

如图5-9所示，用户在医院服务模块中，选择好需要就医的医院，点击预约挂号后，再选择科室和具体日期，通过确认预约后，即可完成预约。

### 5.4.2 医生服务



**图5-10 医生服务界面**



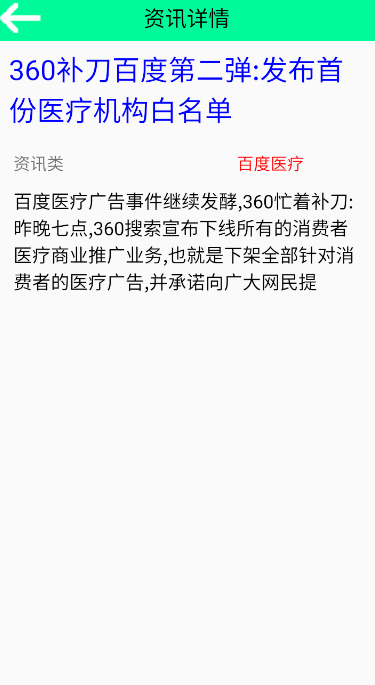
**图5-11 医患记录界面**

如图5-10和图5-11所示，医生可以医生服务模块中查看个人职位信息，包括职称、所在科室以及所属医院。医生可以查看诊断记录，在医生诊断界面中显示了诊断的相关信息，包括确诊疾病、诊断时间以及医院、医生和患者的信息。通过点击诊断记录，医生还可以查看患者的个人信息和具体联系方式，并且可以向患者发送消息，以实现就诊后跟踪病情和必要的生活嘱咐。

## 5.5 医疗信息管理

### 5.5.1 疾病资讯

如图5-10所示，用户可以在医疗诊断模块中，查询相关疾病，系统会显示疾病的具体信息，包括对应科室、病因、症状以及简略的介绍等信息。诊断模块中还可以浏览与健康相关的资讯，点击即可查看资讯的详细内容。



**图 5-12 疾病资讯界面**

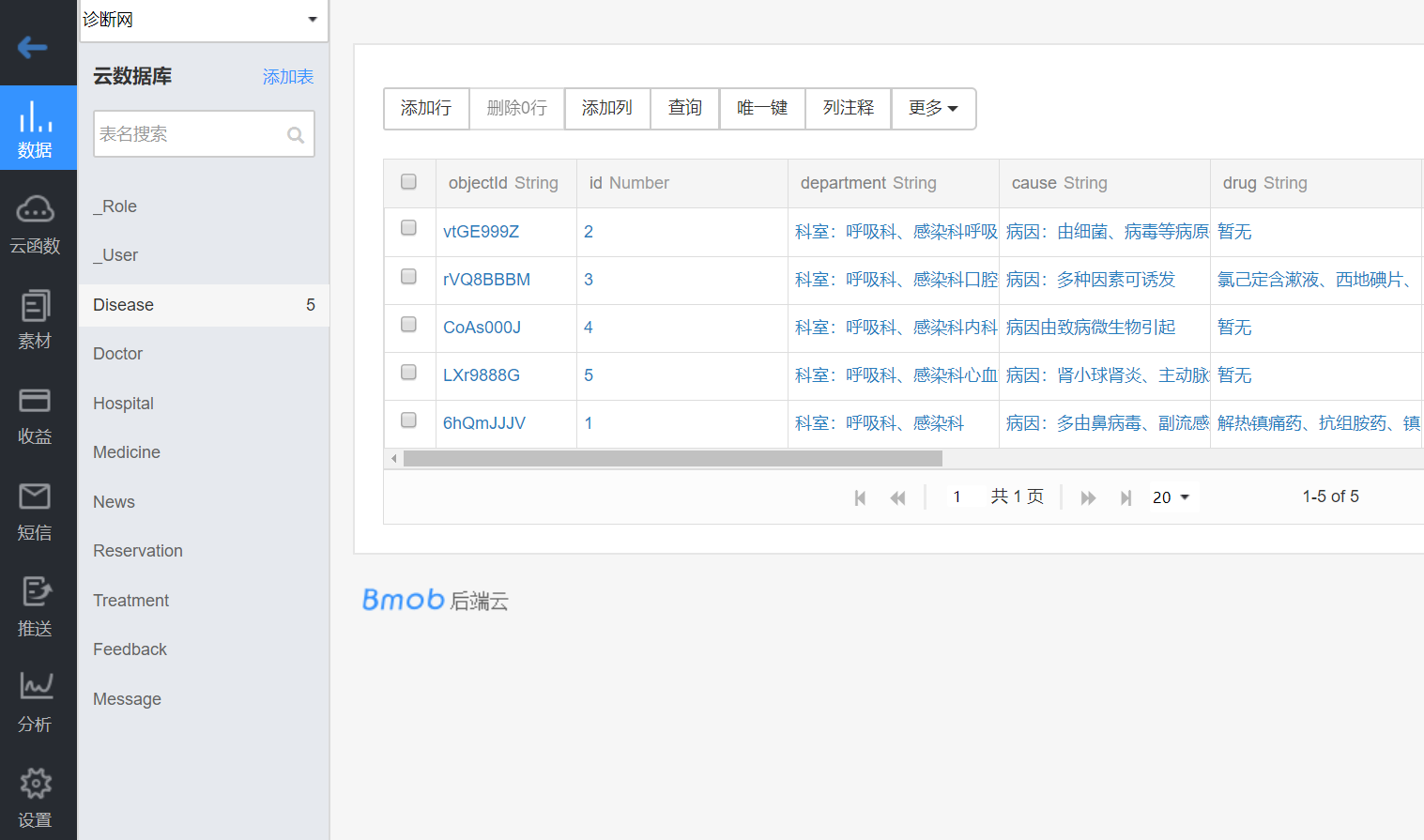
### 5.5.2 预约就诊信息

**图5-13 预约就诊信息界面**

如图5-11所示，用户可以在个人模块查看自己预约和就诊的历史记录，在我的预约界面，还可以通过退号功能来取消对应的预约。

## 5.6 Bmob管理后台

本系统通过Bmob云数据库来实现管理系统后台，如图5-12所示，系统管理员通过登录后可以对系统中各表进行增删改查，并且还可以设置系统用户的操作权限，通过云函数和设置功能可以进一步完善系统的后台管理功能。

**图5-14 Bmob后台管理界面**

## 5.7 本章小结

本章主要是对系统运行的效果进行图片展示，并配以文字描述其操作流程，最后介绍了本系统的Bmob后台管理工具。

# 6 系统测试

## 6.1 测试方法

本系统作为移动应用软件，有多种测试方法，其主要分为黑盒测试、白盒测试、灰盒测试以及自动化测试，黑盒测试主要是对软件功能和界面进行测试，白盒测试主要是验证系统逻辑结构是否合理，灰盒测试则是介于前两者之间的测试方法。本系统采用黑盒测试方法，通过改进软件的主要功能和用户界面，从而更好地提升用户的操作体验。

## 6.2 测试方案及计划

### 6.2.1 系统功能介绍

本系统作为一个医疗移动应用，为了方便用户线上诊断的同时，也为用户提供了完整的医疗服务链，包括医院预约挂号、医生咨询、药品购买以及个人信息和医疗记录管理等功能。因此，系统测试主要是对以上功能进行黑盒测试，验证其功能的有效性和完备性。

### 6.2.2 测试目的

（1）测试系统功能是否错误，是否存在漏洞。

（2）测试系统初始化和终止时是否有误。

（3）测试系统是否能够正确接收输入数据并产生正确的输出结果。

（4）测试系统是否有数据结构错误或外部信息访问错误。

### 6.2.3 测试功能

本系统主要对以下功能进行测试，如表6-1所示。

**表6-1 测试功能范围**

|  |  |
| --- | --- |
| 功能名称 | 描述 |
| 用户登录 | 用户通过输入用户名、密码进行登录 |
| 用户注册 | 用户通过输入邮箱、用户名、密码进行注册 |
| 忘记密码 | 用户输入邮箱、用户名，通过邮件验证重置密码 |
| 购买药品 | 用户选择药品购买，并生成订单 |
| 查看就诊记录 | 医生/患者通过各自模块查看就诊历史记录 |
| 医院预约 | 用户在医院模块选择对应条件预约 |
| 搜索信息 | 用户搜索疾病、医院和药品 |
| 在线诊断 | 用户通过填写症状，系统诊断生成疾病报告 |

## 6.3 测试过程及结果分析

### 6.3.1 测试用例及结果

本系统针对以上需要测试的功能设计了测试用例，如表6-2所示。

**表6-2 功能测试用例**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试功能 | 测试用例 | 预期结果 | 结果 |
| 用户登录 | （1）用户名、密码为空  （2）用户名、密码输入错误  （3）用户名、密码输入正确 | （1）提示信息不能为空  （2）提示用户名密码不对应  （3）提示用户登录成功 | 通过 |
| 用户注册 | （1）用户名、密码和邮箱不符合规范  （2）用户名、密码和邮箱为空  （3）两次输入密码不一致  （4）用户输入信息正确 | （1）提示对应信息不符合规范，请重新输入  （2）提示信息不能为空  （3）提示两次密码输入不一致  （4）提示用户注册成功 | 通过 |
| 忘记密码 | （1）用户名、邮箱为空  （2）用户名和邮箱错误  （3）用户输入信息正确 | （1）提示信息不能为空  （2）提示用户名和邮箱不对应  （3）提示通过邮件链接重置密码 | 通过 |
| 购买药品 | （1）未选择药品结算  （2）选择药品结算 | （1）提示未选择任何药品购买  （2）提示购买成功，生成订单 | 通过 |
| 查看就诊记录 | （1）患者查看就诊记录  （2）医生查看诊断记录 | （1）显示就诊记录  （2）显示诊断记录 | 通过 |
| 医院预约 | （1）未选择预约日期  （2）未选择科室  （3）选择信息预约 | （1）提示用户未选择日期  （2）提示用户未选择科室  （3）提示预约成功 | 未通过 |
| 搜索信息 | 1. 模糊搜索 2. 输入完整检索词 | （1）显示搜索结果  （2）显示搜索结果 | 未通过 |
| 在线诊断 | （1）用户未填写症状  （2）用户正确填写症状 | （1）提示用户填写  （2）显示疾病报告 | 通过 |

### 6.3.2 测试结果分析

本系统通过对以上重要功能进行黑盒测试，检查到两个系统存在的缺陷。如图6-1所示，用户在预约时未选择预约的科室和日期，系统不会提示用户重新选择，便进入下一个界面，在后期改进阶段会设置相关信息的友好提示。如图6-2所示，系统搜索功能不支持模糊搜索，因为Bmob数据库对普通用户不提供模糊条件搜索功能。除此之外，还未对系统的健壮性进行测试，因此本次测试存在一定的缺陷。

## 

## **图6-1 预约缺陷界面 图6-2 模糊搜索缺陷界面**

## 6.4 本章小结

本章主要是对系统的重要功能进行黑盒测试，首先选择需要测试的功能，再根据功能设计对应的测试用例，最后给出测试结果以及对测试中发现的缺陷进行分析。

# 7 总结与展望

## 7.1 总结

互联网技术和传统医疗领域相结合，既可以合理分配医疗资源，又使得互联网继续保持高速发展，同时也能够让医生和患者得到更好的就诊和治疗的体验。本文首先在绪论中依据相关文献，分析了互联网医疗领域和机器学习技术在全球发展的现状，进而选择了适合本系统开发的框架以及相关技术。在系统设计阶段，根据系统所需业务流程对系统功能模块进行了划分和设计，同时也给出了系统数据库的详细设计，并配以图表描述。在系统实现阶段，对系统所采用的机器学习模型和重要功能进行了详细介绍，最后给出了系统运行效果图和测试用例及测试结果。本系统采用了TensorFlow技术在移动终端实现机器学习功能，基于Android原生的Bottom Navigation Activity模板对用户操作界面进行了设计，使得用户界面简单易懂，交互也足够友好。本系统实现了在线诊断、查看疾病、医院、药品的相关信息、预约挂号、医院导航、医患交流、购买药品、管理个人信息以及查看就诊、预约记录等功能，极大地节省了患者就诊和医生诊断的时间，并且提供了完整的医疗服务链，相信本系统可以作为一个很好的互联网医疗移动应用。

## 7.2 展望

本系统由于是个人开发，同时又涉及到很多医疗服务，其中也包括机器学习和TensorFlow技术的实现，因此在开发过程中存在一定的缺陷，同时也因时间、经济和人员限制，在系统分析阶段中一些预期的功能也没有得以实现，本人在后期会逐步完善本系统。本系统在未来所要实现的功能如下所示：

1. 本系统的医疗诊断模型需要更多的临床诊断数据进行训练，并对其加以改进，从而提高诊断正确率。
2. 为用户展示疾病报告后，预期根据数据库分析给出相应的治疗方案。
3. 医院导航功能由于百度地图导航SDK包过大，不适合系统前期开发，在后期希望能够实现定位、多种交通工具导航以及救护车紧急呼叫功能。
4. 本系统的预约、就诊记录都需要和医院的系统连接才能够体现其功能性，因此希望后期能够找到当地医院合作对系统进行进一步分析改进。
5. 本系统的药品购买和配送模块需要进一步优化其业务流程，将药品供给和配送外包给专业公司管理开发，从而使得药品服务更加完善、人性化。

**参考文献**

1. 谢平.“互联网+智慧医疗”现状与发展前景探究[J].数字通信世界,2020(03):157.
2. G. S. Handelman,H. K. Kok,R. V. Chandra, et al.eDoctor: machine learning and the future of medicine[J].Journal of Internal Medicine,2018,284(6):603-619. DOI:10.1111/joim.12822.
3. 汪鹏,吴昊.国内外移动互联网医疗应用现状及未来发展趋势探讨[J].中国数字医学,2014,9(01):8-10.
4. Samant, Piyush,Agarwal, Ravinder.Machine learning techniques for medical diagnosis of diabetes using iris images[J].Computer Methods and Programs in Biomedicine: An International Journal Devoted to the Development, Implementation and Exchange of Computing Methodology and Software Systems in Biomedical Research and Medical Practice,2018,157:121-128.
5. 符雨嫣,宣建伟,韩屹,金春林,王海银.美国“互联网+医疗”服务及对我国的启示[J].中国卫生经济,2020,39(04):94-96.
6. Machine learning, medical diagnosis, and biomedical engineering research - commentary[J]. Foster et al. BioMedical Engineering OnLine 2014, 13:94
7. 周烨.互联网医疗服务发展现状及标准化研究[J].标准科学,2018(11):93-95+122.
8. 李国帆. 基于服务的分布式互联网医疗系统的设计与实现[D].北京交通大学,2019.
9. 李晓然.智能手机操作系统——以Android与iOS对比[J].经济师,2020(02):222-224+226.
10. 软件测试方法和技术[M]. 清华大学出版社有限公司, 2005.
11. 陈钦界. 基于机器学习的智能医疗诊断辅助方法研究[D].国防科学技术大学,2017.
12. 陈希格.一种基于5G互联网医疗诊断平台及方法:CN201910590705.6[P].2019-09-27.
13. 耿明菲.基于医疗大数据分析的门诊辅助诊断模型[J].自动化技术与应用,2019,38(10):51-55,84.
14. 林悦.“互联网+智慧医疗”现状及发展展望[J].中国医疗器械信息,2019,25(18):15-16.
15. 叶明. 基于集成学习的疾病辅助诊断关键技术研究与实现[D].郑州大学,2019.
16. 尚凌月. 疾病辅助诊断方法的研究与应用[D].东北大学,2016.

**致谢**

嗨，四年前的自己，我就要毕业啦！你是否还在为处理人际关系而苦恼，随它去吧，大学里的人都是志同道合才聚在一起，而不是聚在一起才志同道合，你有你的路，好好走就好就可以了。你是否在担心自己能否申请到好的大学，不用太大压力，不要看到别人成功了就贬低自己，最终适合你的才是最好的。感谢四年的大学时光开启了我的成长之路，让我一步一步更接近自己的梦想！

我想感谢南昌航空大学，感谢她提供给了我很多有着美好回忆的地方，夜晚教学楼的天台、烈日下无人的操场、能够静下心好好学习的机房，当然还有最重要的图书馆，在这里我认识到了很多优秀的同学、朋友，听着每天早晚亲切的广播，独自随着人群进出门闸，走出大门那一阵凉风让我感觉今天又是努力满满的一天呀！感谢学校的每一个角落，它们都承载着我独自漫步的时光，让我日后再走过又能勾起很多奋斗努力的回忆。感谢学校里可以自习的教室，让我和同学们可以自由地沟通交流，挥洒汗水去做课程设计和项目。我还想感谢自己在图书馆里读过的每一本书和杂志，它们充实着我的大学时光，让我通晓了很多道理，也更清楚地认识到了自己，谢谢学校带给我的一切！

我想感谢学校里每一位教过我的老师，他们辛勤备课，总是能够耐心地解答我的疑问，教会了我很多专业知识和技术。感谢Manar老师对课堂的认真准备和对我的鼓励，感谢段喜龙老师为数学建模竞赛努力争取的D301机房，感谢蔡虹老师教会了我很多数据库知识，感谢项均老师和梁旗军老师让我有了项目经验，感谢很多对我有恩的老师，你们不仅教会了我很多专业知识，也让我更清楚自己想要成为什么样的人，谢谢亲爱的老师们！

我想感谢在软件学院里认识的朋友和几位室友，虽然我总是习惯一个人去做事情，不怎么合群，但在我需要他们帮助的时候，他们还是义无反顾地去帮我，他们真的很好，也让我知道自己应该怎么去对待朋友，谢谢同学们！

我想感谢我的父母和姐姐，他们给了我温暖的港湾，即使有很多烦心事，也能够和他们倾诉，他们会细心地提醒我和人相处要注意的事情，会不时的问我在学校的情况，他们真的很关心我，谢谢家人们！

最后，感谢自己没有白费这四年的大学时光，希望自己在远方能够好好地继续读研究生，做好自己的事情，待人善良友好，成为那个想要成为的自己！