



单位代码 10006  
学 号 15231096  
分 类 号 TP312

# 北京航空航天大学

B E I H A N G U N I V E R S I T Y

## 毕业设计 (论文)

基于深度学习的数学算式识别微信小程序  
的研究与实现

学 院 名 称 高等理工学院  
专 业 名 称 计算机科学与技术专业  
学 生 姓 名 郝浩宇  
指 导 教 师 阮利

2019 年 06 月

# 北京航空航天大学

## 本科生毕业设计（论文）任务书

### I、毕业设计（论文）题目：

基于深度学习的数学算式识别微信小程序的研究与实现

### II、毕业设计（论文）使用的原始资料（数据）及设计技术要求：

1

2

3

4

5

### III、毕业设计（论文）工作内容：

微信小程序 App 端的设计与实现

基于 Node.js 的服务端的设计与实现

基于 Resnet18 的字符识别模型

基于统计直方图的算式分割算法

基于协同过滤推荐算法实现相似算式推荐

撰写毕业论文

#### IV、主要参考资料：

1

---

2

---

3

---

4

---

5

---

6

---

7

---

8

---

\_\_\_\_高等理工\_\_\_\_学院 计算机科学与技术 专业类 \_\_\_\_152314\_\_\_\_班

学生\_\_\_\_郝浩宇\_\_\_\_

毕业设计(论文)时间： \_\_\_\_2018\_\_\_\_年\_\_\_\_10\_\_\_\_月\_\_\_\_23\_\_\_\_日至\_\_\_\_2019\_\_\_\_年\_\_\_\_06\_\_\_\_月\_\_\_\_11\_\_\_\_日

答辩时间： \_\_\_\_2019\_\_\_\_年\_\_\_\_06\_\_\_\_月\_\_\_\_11\_\_\_\_日

成 绩： \_\_\_\_\_

指导教师： \_\_\_\_\_

兼职教师或答疑教师（并指出所负责部分）：

\_\_\_\_\_  
\_\_\_\_\_

\_\_\_\_系（教研室）主任（签字）： \_\_\_\_\_

注：任务书应该附在已完成的毕业设计（论文）的首页。



## 本人声明

我声明，本论文及其研究工作是由本人在导师指导下独立完成的，在完成论文时所利用的一切资料均已在参考文献中列出。

作者： 郝浩宇

签字：

时间： 2019 年 06 月



## 基于深度学习的数学算式识别微信小程序的研究 与实现

学 生：郝浩宇

指导教师：阮利

### 摘 要

摘要

关键词：关键字



# **Attention Aware Bi-directional Gated Recurrent Unit Based Framework for Sentiment Analysis**

Author: Firstname Lastname

Tutor: Firstname Lastname

## **Abstract**

abstract

**Key words:** keyword



## 目 录

1 绪论 .....	1
1.1 课题研究背景和意义 .....	1
1.2 课题研究现状 .....	1
1.2.1 识别字符算法的研究现状 .....	1
1.2.2 移动 App 研究现状 .....	2
1.3 论文组织结构 .....	3
2 相关技术介绍 .....	4
2.1 微信小程序平台技术介绍 .....	4
2.2 基于 Express 框架的 Node.js web 服务器 .....	4
2.3 MySQL 数据库 .....	4
2.4 OpenCV .....	4
2.5 本章小结 .....	4
3 需求分析 .....	5
3.1 整体需求分析 .....	5
3.2 账号模块需求分析 .....	5
3.3 算式识别模块需求分析 .....	5
3.4 错题集模块需求分析 .....	6
3.5 本章小结 .....	6
4 系统设计 .....	7
4.1 App 整体设计 .....	7
4.2 App 客户端设计 .....	7
4.3 服务器设计 .....	8
4.4 数据库设计 .....	8
4.5 网络访问接口 .....	10
4.6 本章小结 .....	11



5 系统实现 .....	12
5.1 开发环境的创建 .....	12
5.2 账号功能的实现 .....	12
5.2.1 登陆功能 .....	12
5.2.2 注册功能 .....	13
5.2.3 管理员功能的实现 .....	14
5.2.4 识别算式功能的实现 .....	15
5.2.5 错题集功能模块的实现 .....	16
5.3 基于协同过滤推荐算法实现相似算式推荐 .....	18
5.4 本章小结 .....	18
6 基于深度学习的算式识别算法 .....	19
6.1 基于统计直方图的算式分割算法 .....	19
6.2 基于 ResNet18 的字符识别模型 .....	20
6.2.1 数据集的预处理 .....	20
6.2.2 训练模型 .....	20
6.3 本章小结 .....	20
7 系统测试 .....	21
7.1 字符识别模型准确率的测试 .....	21
7.2 识别算式准确率的测试 .....	21
7.3 App 功能测试 .....	21
7.4 服务端性能测试 .....	21
8 总结 .....	22
8.1 工作总结 .....	22
8.2 问题与展望 .....	22
参考文献 .....	23
致谢 .....	24





## 1 绪论

### 1.1 课题研究背景和意义

作为一门重要的基础学科,数学为其它学科提供了重要的理论支撑,掌握了数学知识就意味着掌握了领悟一种现代科学的理论和工具,学到了一种理性思维模式。我们在受教育的每个阶段几乎都要学习数学,数学不仅为学生进一步学习提供了必要准备,而且在学生终身发展的过程中也有不可替代的作用。尤其是小学数学教学在培养学生逻辑思考能力,开发智力和非智力因素上的作用不容小觑。因此从小培养学生学习数学的兴趣,建立起学好数学的自信心就显得尤为重要。家庭作业作为课堂教学的延伸,既能帮助学生巩固完善课堂所学知识,形成数学能力,又能帮助老师了解学生对知识的掌握情况,合理安排教学计划,作业的分析反馈中含有巨大的教育价值。传统的批改作业方式使老师埋在作业堆里做着机械重复的工作,老师花费大量时间在作业的批改上,却没有实现信息的反馈,对学生的反思进步没有很大的帮助。班级人数众多,每个人存在的问题经常没有记录也无法查看,很难对出错问题进行全面分析,也不能对每个同学的错误进行一一指导,而且长期进行纸面上的数学练习,学生可能会感到枯燥。随着以 4G 为代表的移动通信技术的迅速升级和智能手机的全面普及,借助移动终端的便利性,短、平、快的学习方式——“碎片化学习”逐渐流行起来,成为了一种重要的学习方式。针对传统作业中存在的问题,可以把传统的家庭作业和移动互联网技术结合起来,老师、学生和家长都能利用 App 一键检查作业,同时把错误问题上传到服务器,既节省了老师批改作业的时间,也记录下了学生的易错点,可以为后续教学的展开提供很大的帮助。

因此,开发一款能够提供检查作业功能的 App,能够为解决当前小学数学作业中存在的问题提供协助,所以本文针对算术运算检查、班级管理、错误记录反馈、统计分析、趣味练习题等需求结合深度学习和移动 App 技术,设计并实现了一款数学教学微信小程序。

### 1.2 课题研究现状

#### 1.2.1 识别字符算法的研究现状

算式识别模块是基于图像处理、模式识别等技术的综合系统,一般流程是首先进行图像收集,然后对图像进行预处理,包括图像二值化、形态学处理、字符分割、归一化



等操作, 目的是突出字符区域, 为后续的字符识别做准备; 一组算式中多个字符的图像根据字符间隙, 进行字符分割, 分割成单个字符进行识别。由于相邻数字的重叠和连接, 字符识别可能会比较困难, 为了解决这个问题, 可以使用基于定向滑动窗口的手写连接数字的分段识别法, 这种方法允许通过同时找到相邻数字和切割路径之间的互连点来根据连接配置分离相邻数字。最终可以把提取的字符特征输入深度学习模型中进行字符识别, 判断算式的结果是否正确。

字符识别可以使用多种机器学习算法实现, 各有优缺点。支持向量机在小样本学习中可以得到很高的准确率; BP 神经网络来通过不断调整神经各层间的权值和阈值来使实际样本的期望值和实际输出值的方差的平均值最小来实现数字识别, 具有自学习能力, 但学习速度慢, 容易陷入局部极值; CNN 的网络拓扑结构能和图像很好的吻合, 特征分类效果好; KNN 重新训练代价低但计算量大, 样本不平衡时预测偏差较大; 决策树易于理解和解释、运行速度快, 但缺失数据处理困难, 容易出现过拟合; 朴素贝叶斯可以处理多分类问题, 对小规模数据表现好, 但对数据形式敏感分类决策存在错误率。

从已经发表的文章中的工作来看, 现在研究点多集中于单个字符的识别或单个算式的识别, 对一张图片上多个算式的识别, 公开的成果不多, 因此研究并实现这一算法, 具有一定的现实意义。

### 1.2.2 移动 App 研究现状

移动 App 按照开发模式可以大体分为 Native App、Web App 和 Hybrid App 三种。Native App 是使用原生语言编写第三方应用程序, 现有 IOS 和 Android 两种。它运行在移动终端的操纵系统中, 可以直接调用操作系统的 API, 性能较好, 可以提供最优的用户体验。Native App 虽然性能卓越, 但开发成本也较高, 维护更新复杂, 而且不支持跨平台使用, 可移植性差, 代码重用率低, 开发门槛较高, 不适合个人开发者进行开发。

Web App 是运行在浏览器上的应用, 通过把页面部署到服务器中, 用户使用浏览器访问, 无需下载安装, 解决了 Native App 无法跨平台使用的问题。但 Web App 的缺点也十分明显, 对网络的依赖很大, 访问页面需要到服务器加载资源, 速度较慢; Web App 在浏览器中运行, 受限于 HTML5 的技术特性, 几乎无法使用设备的本地资源, 很多功能无法实现, 用户体验较差。

Hybrid App 是 Native App 和 Web App 两者混合的产物, 实际就是 Native 的框架加上 Web 的内容。它既能利用移动设备的 API 带来良好的用户体验, 也具有 Web App 跨平台、高效开发、快速发布的优点, Web 内容可以做到一次发布所有平台生效, 使用网



页语言编码开发难度较小、效率较高。微信小程序作为一种特殊的 Hybrid App 不受手机操作系统的限制，可以调用手机系统的功能，在性能上接近 Native App。而且国内微信用户数目众多，加上小程序应用场景广泛，培养了很多用户群体。综合考虑以上因素，微信小程序既保证了平台的通用性、易于开发，又有广泛的用户群体，最适合本文用来实现辅助教学 App。

### 1.3 论文组织结构

本论文主要研究

本文的组织结构如下：

第一章：绪论。在绪论章节中，本文首先阐述了本课题的研究背景，

第二章：相关知识和工作介绍。

第三章：

第四章：

第五章：

第六章：

第七章：

第八章：



## 2 相关技术介绍

### 2.1 微信小程序平台技术介绍

### 2.2 基于 Express 框架的 Node.js web 服务器

### 2.3 MySQL 数据库

### 2.4 OpenCV

### 2.5 本章小结

### 3 需求分析

#### 3.1 整体需求分析

App 的主要功能是检测算式，同时可以统计、分析检测过的算式，给学生推荐常出错的习题，帮助学生学习数学。App 的用户有老师、学生和管理员三种角色，老师和学生需要注册、登陆功能，管理员有查看、添加和删除用户的需求。

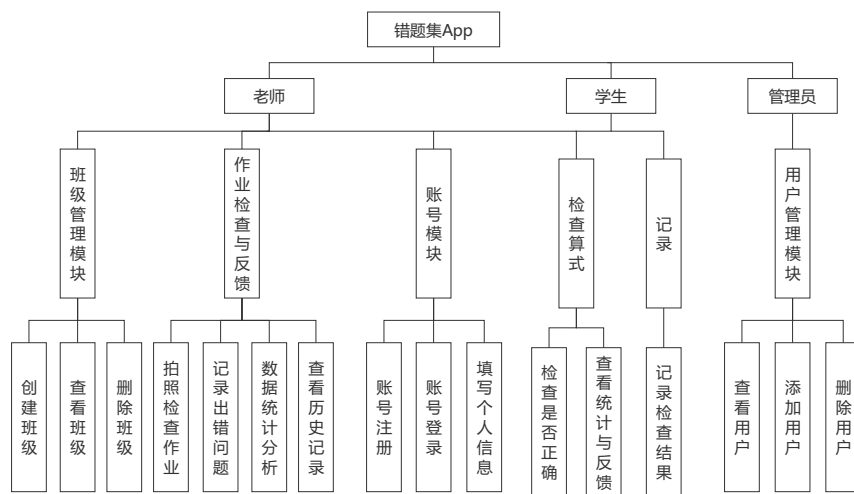


图 3.1 总体需求

#### 3.2 账号模块需求分析

用户注册：用户需要提交用户名、密码和 email 地址，同时选择自己是学生还是老师，注册登陆完成后，用户可以选择进一步完善个人信息，包括姓名、性别、年龄和班级。

用户登陆：用户提交用户名和密码，系统判断是否允许用户登陆，不同角色的用户登陆后，能使用对应的功能。

#### 3.3 算式识别模块需求分析

用户可以选择从本地选择图片或是使用相机拍照，选择图片后上传到服务器，服务器上的识别程序需要判断出每个等式的正误，并在图片上做出相应的标记，识别结束后把结果存储在数据库，同时把结果图片返回客户端，客户端需要把收到的响应呈现在视



图层。

### 3.4 错题集模块需求分析

用户可以看到自己上传的图片处理结果的历史记录和出错题目的统计分析结果。系统会根据用户的历史做题结果提供相似题目的推荐。

### 3.5 本章小结

## 4 系统设计

### 4.1 App 整体设计

客户端主要是组成页面和处理逻辑的文件，用户在客户端的操作会向服务端发送 https 请求，服务端收到请求后根据匹配的路由调用函数处理请求，处理结束后向服务端返回响应。

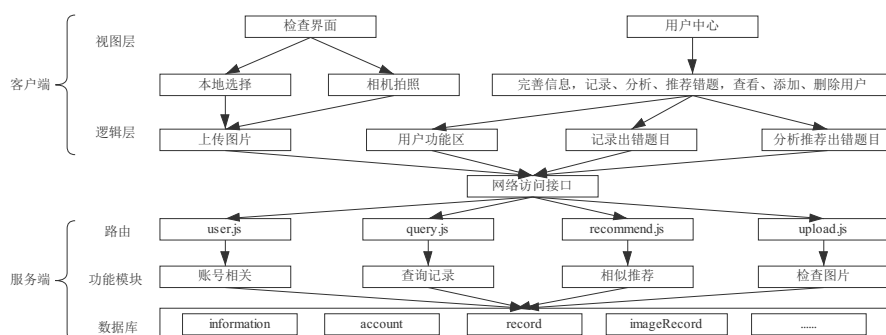


图 4.1 整体设计

### 4.2 App 客户端设计

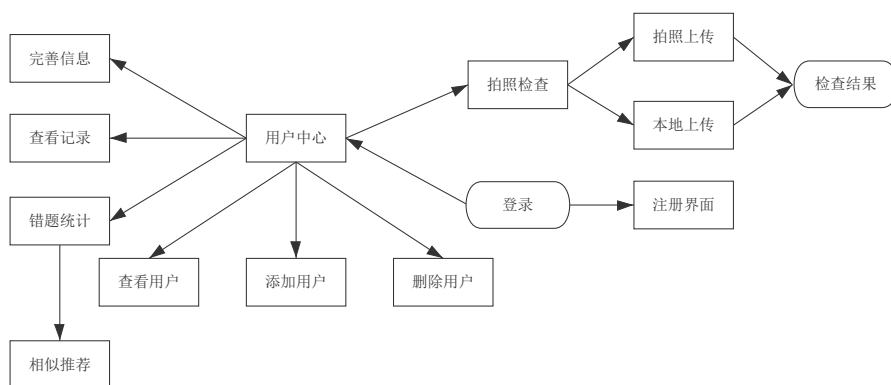


图 4.2 客户端页面流程图

根据需求实现客户端页面，用户如果未登录，在应用启动后首先会弹出登陆界面，用户登陆后会跳转到用户中心界面，不同角色的用户在用户中心看到的跳转选项是不同的。学生与老师可以使用完善信息、查看历史检查记录和错题统计功能，学生只能看



到自己的错题统计,老师可以看到所有学生的错题统计,管理员在用户中心可以使用查看、添加和删除用户的功能。客户端底部有两个导航栏:拍照检查和用户中心。拍照检查提供拍照上传和本地上传两种选择,图片上传结束后会跳转到结果界面。

### 4.3 服务器设计

服务端使用 Express 框架实现,服务端程序分为两层,第一层用于接收 App 端传来的各种数据和请求,本项目针对 App 功能编写了四个 js 文件 users.js、upload.js、query.js、recommend.js 分别用于处理用户账号、图片检查、数据查询和算式推荐请求。第二层根据第一层收到的请求,调用函数处理传来的数据,进行相关的数据库操作。如果是检查图片内容的请求,会调用子进程处理图片,子进程执行用来处理图像的 python 程序,然后把处理后的结果图片存储在硬盘上,同时把识别出的算式返回给父进程,父进程把这些内容存储到数据库并发送到客户端。

### 4.4 数据库设计

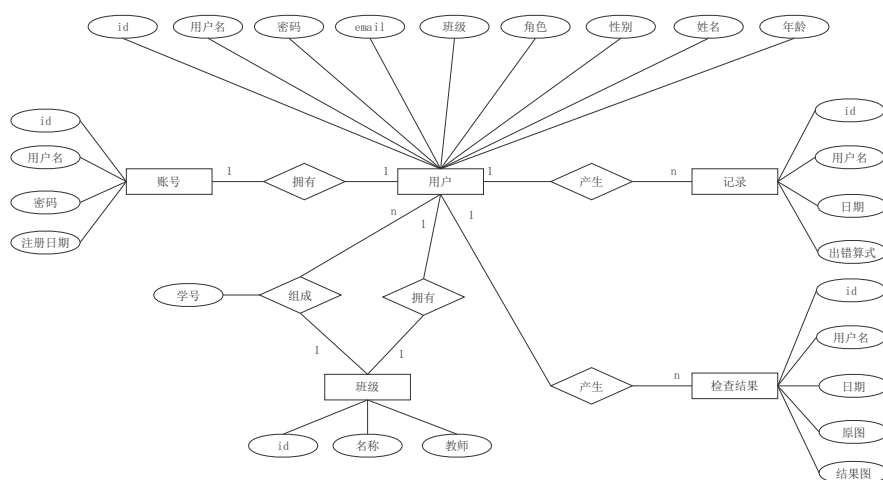


图 4.3 数据库 E-R 图





表 4.1 用户信息表

字段名	类型	代表内容	其他
id	int(11)	id	auto increment, PRI
name	varchar(45)	用户名	PRI
role	varchar(45)	电子邮箱地址	-
password	varchar(45)	角色	-
trueName	varchar(45)	真实姓名	-
gender	varchar(45)	性别	-
age	varchar(45)	年龄	-
class	varchar(45)	班级	-

表 4.2 账号表

字段名	类型	代表内容	其他
id	int(11)	id	auto increment, PRI
name	varchar(45)	用户名	PRI
password	varchar(45)	密码	-
date	varchar(45)	注册日期	-

表 4.3 错题记录表

字段名	类型	代表内容	其他
id	int(11)	id	auto increment, PRI
name	varchar(45)	用户名	PRI
date	varchar(45)	出错日期	-
formula	varchar(45)	出错算式	-

表 4.4 检查结果表

字段名	类型	代表内容	其他
id	int(11)	id	auto increment, PRI
name	varchar(45)	用户名	PRI
date	varchar(45)	出错日期	-
originalPath	varchar(255)	原图的路径	-
resultPath	varchar(255)	结果图片的路径	-



表 4.5 班级表

字段名	类型	代表内容	其他
id	int(11)	id	auto increment, PRI
name	varchar(45)	用户名	PRI
teacher	varchar(45)	老师用户名	-

4.5 网络访问接口

客户端与服务端进行数据交换的网络接口

<https://www.haohybh.cn/users/>

查询用户是否已经登陆，如果未登陆 https 响应返回” none”；

<https://www.haohybh.cn/users/login>

查询用户名和密码是否匹配，如果匹配返回用户角色；

<https://www.haohybh.cn/users/addUser>

添加用户，如果操作成功返回 success: ” yes”，失败返回 success: “no”；

<https://www.haohybh.cn/users/deleteUser>

删除用户，如果操作成功返回 success: ” yes”，失败返回 success: “no”；

<https://www.haohybh.cn/users/queryName>

查询用户名是否存在，如果已经存在返回 exist: “yes”，不存在返回 exist: “no”；

<https://www.haohybh.cn/users/queryAllUser>

查询所有用户的用户名，如果存在用户返回所有用户的用户名，否则返回 exist: “no”；

<https://www.haohybh.cn/users/addAttributes>

添加用户信息，如果操作成功返回 success: ” yes”，失败返回 success: “no”；

<https://www.haohybh.cn/fileUpload>

提交要检查的图片，处理结束后返回 path: “xxx.jpg”；

<https://www.haohybh.cn/check>

查询本用户的错误记录，如果存在相应的记录返回查询结果，否则返回 exist: “no”；

<https://www.haohybh.cn/check/teacher>

查询所有用户的错误记录，如果存在相应的记录返回查询结果，否则返回 exist: “no”；

<https://www.haohybh.cn/check/history>

查询本用户的历史检查记录，如果存在相应的记录返回查询结果，否则返回 exist: “no”；



#### 4.6 本章小结



## 5 系统实现

### 5.1 开发环境的创建

App 端：本项目的 App 端和服务端采用 JavaScript 语言进行开发，App 端使用微信开发者工具作为开发环境，微信开发者工具是一个可以模拟微信客户端表现的开发和调试工具；

识别模块：识别模块采用 python 语言开发，版本为 3.6.8，开发环境是 PyCharm；

服务端：服务端使用 Nginx 作为 web 服务器，JavaScript 的运行环境是 Node.js，开发环境是 WebStorm，使用的数据库是 MySQL 8.0.13。本项目部署在腾讯云学生机上进行运行和测试，具体配置信息为一颗核心处理器，2G 内存，1Mbps 带宽，50GB 系统盘，公共镜像为 CentOS 7.2。

### 5.2 账号功能的实现

#### 5.2.1 登陆功能

用户输入用户名和密码后通过表单提交给服务器，如果提交失败 App 端会提示用户检查是否为本机网络连接问题，如果提交成功服务端在收到网络请求后，在数据库中查询用户是否存在和用户的密码，把表单提交的密码与从数据库取出的密码比较后确定密码是否正确，把比较结果发送给 App 端。App 端根据收到的结果决定是否让用户登陆。如果用户名和密码匹配，在服务端生成新的 session，http 响应把用户角色返回给客户端。

小程序没有 cookie 机制，需要自行实现 cookie 来保持客户端和服务端的连接，session 存储在服务器上，cookie 有效期设为 1800 秒，客户端收到 http 响应后把响应头中的 set-cookie 取出，赋值到全局变量中，添加到后面每次 http 请求的请求头中，用来保持连接，确定是那个用户的操作。



1:14 错题集

请登录

用户名: 请输入用户名

密码: 请输入密码

登录

没有账号, 去注册

图 5.1 登陆界面

### 5.2.2 注册功能

App 端把用户填写的表单提交给服务器, 表单提交的数据包括用户名、用户角色、电子邮箱地址和登陆密码, 服务器收到表单内容后把用户添加到数据库中, 每个用户的用户名是唯一的, 所以在用户填写表单时要查询该用户名是否已被使用, 如果用户名已经被占用、两次输入的密码不一致或表单未完全填写, 会分别提示用户该名称已经被占用、密码与确认密码不一致和信息填写不完整, 并且用户无法提交数据。注册成功后会跳转到用户登陆界面。

1:14 错题集

注册

用户名: 请输入用户名

角色: 学生

email: 请输入email

登录密码: 请输入密码

确认密码: 请再次输入密码

☐ 阅读并同意《相关条款》

注册

已有账号, 去登录

图 5.2 注册界面



### 5.2.3 管理员功能的实现

查看用户：管理员可以看到所有用户和他们的角色以及所在班级。点击查看用户按钮后客户端会向服务端发送查询请求，服务端返回所有用户的用户名、角色和班级，客户端收到响应后把数据以特定的样式呈现给管理员。

添加用户和注册类似，按要求填写好数据后，提交表单后服务端把相应的信息写进数据库就完成了添加用户，如果用户名已存在或两次输入密码不一致或未填写完整，会分别提示用户名已被使用、输入密码不一致和请把信息填写完整。

删除用户只需填写要删除的用户名，服务端接受请求后将数据库中用户名为请求中用户名的记录全部删除，删除用户就完成了。如果用户名不存在或者未填写用户名会弹出相应的提示，点击删除用户按钮会弹出确认窗口防止用户错误操作。



图 5.3 管理员界面

### 5.2.4 识别算式功能的实现

识别算式模块主要功能是把图片上传到服务器,服务器收到用户检查请求后开启子进程执行识别程序,识别和检测程序检测图片中的算式之后,把结果标记在图片上,并把检查获得的数据存储到数据库,最后通过 http 响应以 json 或字符串的形式把结果图片的 url 发送给 App 端。App 端通过 http 请求获取数据后,通过数据绑定把图片呈现在视图层。

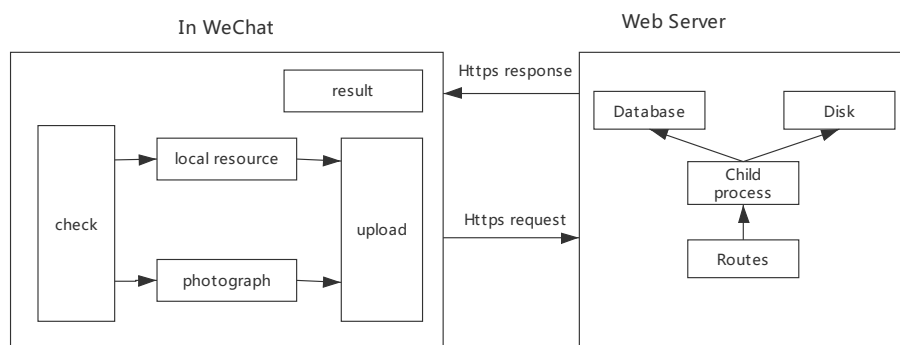


图 5.4 识别过程

### App 端上传图片 and 展示结果

拍照和从相册中选择是上传图片的两种来源,可以点击相机图标打开摄像头拍照,或者从相册中选择图片。



图 5.5 识别算式界面



使用 `ctx.takePhoto` 调用摄像头拍摄图片, 然后使用 `wx.chooseImage` 和 `wx.uploadFile` 把图片上传到服务器上的临时目录; 完成图片上传后会跳转到结果界面, 结果界面通过 `http` 请求获取检查结果图片的 `url`, 在页面完成初次渲染时把结果呈现在视图层。

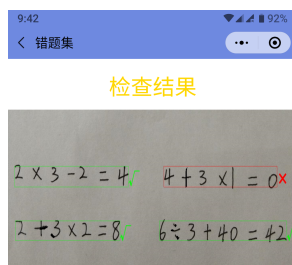


图 5.6 识别结果界面

## 服务端处理图片

服务端是一个 Node.js Express 项目, 通过 **Multer** 中间件获取上传的文件, 并将它们存储在硬盘中。同时创建子进程来调用处理图片的 python 程序, 使用 `python` 命令行参数来输入图片路径和产生结果路径, python 程序获得图片输入后, 首先进行转灰度图、二值化, 然后统计字符像素的直方图, 选取合适的阈值切分算式, 同时标记每个字符在图片中的位置和算式的序号, 方便后面标记结果。把切分好的图片填充和调整到  $28 \times 28$  的大小后输入到模型中识别, 把识别的结果拼接成算式, 判别是否正确, 把结果标记在原图上, 把结果图存在硬盘中, 并打印出错的算式, 父进程获得子进程的输出转化成字符串后, 把这些数据存储在数据库中, 最后产生 json 格式的 `http` 响应把结果图片的 `url` 传给

### 5.2.5 错题集功能模块的实现

学生在用户中心可以完善自己的个人信息, 查看自己提交过的历史错题记录和对自我错题的统计分析以及相应错题的相关推荐。





图 5.7 用户中心界面

查看检查的历史记录，点击错题记录后客户端发送查询历史记录请求，服务端收到请求后在数据库查询相应数据，然后通过响应把数据发送到客户端，客户端按照日期展示历史检查结果，点击图片后可以放大预览。

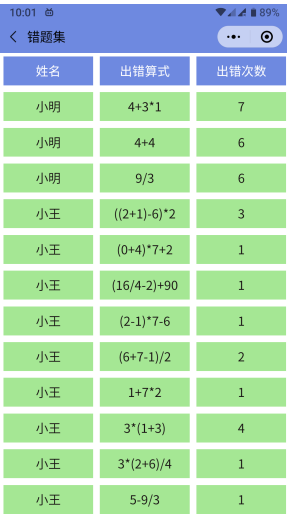
统计自己出过错的算式，点击错题分析按钮发送查询请求，服务端查询该用户的出错题目和次。客户端收到结果后可以看到自己出错的题目和出错的次数，点击出错算式后的推荐链接后，服务端会根据用户名和算式计算出与该算式相关的其它算式，在跳转后的页面呈现给用户。

出错算式	出错次数	相似推荐
$(16/4-2)+90$	1	推荐
$(0+4)*7+2$	1	推荐
$3*(2+6)/4$	1	推荐
$5-9/3$	1	推荐
$6+8/4*2$	1	推荐
$1+7*2$	1	推荐
$9*7-8*6$	1	推荐
$(2-1)*7-6$	1	推荐
$((2+1)-6)*2$	3	推荐
$999+1-9$	2	推荐
$(6+7-1)/2$	2	推荐
$3*(1+3)$	4	推荐

图 5.8 错题分析图

老师可以查看所有人的出错记录和对应的次数，角色为老师的用户在点击错题分析按钮后，会发送查询所有用户出错记录的请求，服务端鉴定完用户名和角色后查询对应

的数据，然后把查询结果发送到客户端，客户端把查询结果呈现在视图层。



姓名	出错算式	出错次数
小明	4+3*1	7
小明	4+4	6
小明	9/3	6
小王	((2+1)-6)*2	3
小王	(0+4)*7+2	1
小王	(16/4-2)+90	1
小王	(2-1)*7-6	1
小王	(6+7-1)/2	2
小王	1+7*2	1
小王	3*(1+3)	4
小王	3*(2+6)/4	1
小王	5-9/3	1

图 5.9 错题统计图

5.3 基于协同过滤推荐算法实现相似算式推荐

基于物品的协同过滤推荐算法预先根据所有用户对物品的评价计算出所有物品之间的相关性，然后把与此件物品最相关的物品推荐给用户。为了实现相似算式推荐，可以把用户做错该算式的次数作为用户对此道题的评价，使用所有用户做错算式的次数计算每两道算式间的皮尔逊相关系数。

皮尔逊相关系数的计算公式:

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

将算式按照皮尔逊系数排序，从高到低取出该用户未做的若干道算式，作为该道算式的推荐。也可以把该用户对几道算式的评分作为权重，加权排序后得到新的算式与出错算式的相关性。

5.4 本章小结



## 6 基于深度学习的算式识别算法

### 6.1 基于统计直方图的算式分割算法

为了检查算式正确与否,需要识别出算式中的每一个字符,本文的思路是识别出每一个字符后,拼接成算式然后判断是否正确,并且本文训练模型的数据集是单个字符,只能识别含有单个字符的图片。因此,把算式切分成字符就十分必要。

使用手机随手拍出的照片经常是光照不均匀的,对图像的二值化操作和投影直方图的计算会产生很大干扰。所以需要根据背景像素来确定阈值,使用局部阈值自适应算法来实现图片二值化。本文使用 OpenCV 的 `adaptiveThreshold` 函数将灰度图二值化,自适应方法使用高斯分布加权和计算局部阈值,二值化类型选择 `THRESH_BINARY`,大于局部阈值的像素点设为最大值。

$$r(x,y) = \begin{cases} 255, & s(x,y) > t(x,y) \\ 0, & \text{others} \end{cases}$$

$s(x,y)$  是当前点的像素值,  $t(x,y)$  是区域内的高斯均值减去函数的最后一个参数 `double C` 的结果,  $r(x,y)$  是当前点二值化后的像素值。

获取二值化的图像后,利用字符间隔的特点,把二维的图像矩阵先按行统计,把二维的矩阵缩减成一列,得到每一行的投影直方图,设定阈值和得到的直方图比较可以得到每一行算式的上边沿和下边沿,然后就可以把水平方向的一行算式切割出来;获取单个算式后,把二维的图像矩阵按列统计,缩减成一行,把设定的阈值与缩减好的矩阵比较获取每一个字符的左边缘和右边缘。在切割图片获取单个字符图片的同时记录它们在原图上的位置和所在算式的序号。

通过上一步一张图片上的算式被分割成了单个字符,由于是紧贴着边缘剪切,图片的长宽比有可能比较极端,不能直接输入到模型中识别。即使根据图片的长宽比,进行上下和左右填充,在一行算式排列倾斜时,也会存在图片内的字符被缩放的过小,导致无法正确识别的现象。本文采取获取字符紧贴边缘的图片后再填充的策略,首先获取图像中像素不为零的点,然后找到包含这些点的最小矩形,根据找到的矩形切割图片获取字符紧贴边缘的图片,如果图片的长和宽均不足原图的  $1/10$  或不存在最小矩形,则认为该字符为图片中的干扰,放弃识别这个字符。最后根据图片的长宽差进行填充,调整

图片大小为  $28 \times 28$ 。将图片导入模型识别出对应的字符后，拼接成算式字符串，然后判断算式是否正确，根据判断的结果，在原图上做出对应的标记。

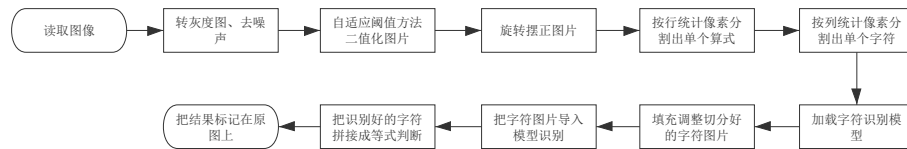


图 6.1 识别步骤

## 6.2 基于 ResNet18 的字符识别模型

### 6.2.1 数据集的预处理

本模型的数据集由 `minist` 数据集和手写运算符组成，数字图片来自 `minist` 数据集，图片大小为  $28 \times 28$ ，运算符数据集来自 `kaggle`，图片大小为  $45 \times 45$ 。首先需要把没有规律的图片名称改为按数字递增，方便划分训练集、测试集和加载到神经网络。运算符图片的大小为  $45 \times 45$ ，且图片内的字符接触图片边缘，而本文从整张图片中切分出的字符图片内的字符与图片边缘不相交，所以需要填充图片边缘后再调整大小。将运算符图片的上下左右边缘各填充 10 个像素变为  $65 \times 65$ ，然后调整大小为  $28 \times 28$ 。

### 6.2.2 训练模型

训练集共有 60000 张数字图片和加、减、乘、除、等号、左右括号每种运算符 5000 张；测试集中有 10000 张数字图片和 10500 张运算符图片，每种运算符占 1500 张，将手写运算符拼接到 `minist` 数据集后打乱顺序后导入残差网络训练。使用随机梯度下降算法进行训练优化，初始学习率设为 0.1，每 30 轮降低 10% 的学习率，冲量参数设为 0.9，权重衰减设为  $5e-4$ 。导入到 ResNet18 后训练 90 轮后在测试集上的识别准确率为 99.776%。



图 6.2 训练步骤

## 6.3 本章小结



## 7 系统测试

### 7.1 字符识别模型准确率的测试

表 7.1 模型准确率测试

name	accuracy	number of samples	incorrect results
0	100%	150	none
1	98%	100	( and )
2	99%	100	4
3	99%	100	(
4	100%	100	none
5	98%	100	( and )
6	97%	100	( and 5
7	91%	100	), + and 1
8	100%	100	none
9	100%	100	none
+	100%	100	none
-	100%	100	none
*	100%	100	none
\	97%	100	= and 2
=	97.2%	110	2 and 5
(	100%	110	none
)	98%	100	7

### 7.2 识别算式准确率的测试

### 7.3 App 功能测试

### 7.4 服务端性能测试



## 8 总结

### 8.1 工作总结

### 8.2 问题与展望



## 参考文献



## 致谢

致谢部分