基于 TensorFlow.js 的垃圾分类识别软件的设计与实现

计算机与信息科学学院　计算机科学与技术专业　2017 级　程远

指导教师陈勇

摘要：本系统是一款以移动端为载体的垃圾分类网站，为响应国家大力提倡环保和垃圾分类的号召，于是便设计实现了本系统，解决了日常生活中因为不清楚垃圾的分类这一痛点。本系统主要实现了文字搜索和图片识别功能，整体技术栈选用Node.js的Koa框架作为后端技术支撑，以MySQL关系型数据库存储垃圾的分类信息；图像识别方面采用TensorFlow.js + Mobilenet预训练模型；前端方面采用了Vue.js框架和基于Vue.js2.0定制的Vant移动端UI组件库。

关键词：Node.js；Koa；MySQL；TensorFlow.js；Mobilenet；Vue.js；Vant移动端UI组件库；垃圾分类；图像识别

Abstract: This system is a garbage classification website based on mobile terminals. As the country vigorously promotes environmental protection and garbage classification, the system is designed and implemented, which solves the pain point of not knowing the classification of garbage in daily life. This system mainly implements text search and image recognition functions. The overall technology stack uses the Koa framework of Node.js as the back-end technical support, and uses MySQL relational database to store garbage classification information; image recognition uses TensorFlow.js + Mobilenet pre-training Model; the front-end uses the Vue.js framework and the Vant mobile UI component library customized based on Vue.js2.0.

Key words: Node.js; Koa; MySQL; TensorFlow.js; Mobilenet; Vue.js; Vant mobile terminal UI component library; garbage classification; image recognition

1　项目背景

随着我国经济的飞速发展，保护环境和垃圾分类也越来越受到国家和人民的重视。2017 年 10 月 18 日，习近平在十九大报告中指出，坚持人与自然和谐共生。必须树立和践行绿水青山就是金山银山的理念，坚持节约资源和保护环境的基本国策[1]。

但日常生活中垃圾的分类问题却是一个让人十分头疼的问题，常常有“这到底是可回收垃圾还是其他垃圾啊，这到底是厨余垃圾还是有害垃圾？”这样的疑问。于是设计了这款垃圾分类系统来帮助人们解决日常生活中“这到底是可回收垃圾还是其他垃圾啊，这到底是厨余垃圾还是有害垃圾？”这样的疑问。本系统采用移动端网页为载体使用成本低，注重功能的实用性，页面简介而又不失美观。

2 系统可行性分析与需求分析

可行性分析与需求分析主要阐述了系统设计和技术选型的可行性，系统的核心需求和功能上的设计，明确系统的主要功能和模块划分。调研业内成熟和常见的解决方案，从而更好的解决问题，完善系统。

2.1 可行性分析

2.1.1 经济可行性

本系统采用移动端网页为载体，客户端为一部装有标准 DOM 浏览器且可以访问互联网的手机即可，由于现代社会智能手机的普及，客户端的成本几乎可以忽略不计。服务器方面由于本系统设计轻量，系统注重功能的实用性和简洁性，服务器方面只需要一台常规配置的电脑即可(如：1GB 内存、1 兆带宽、30GB 硬盘)。现在各个云服务器运营商(如：阿里云，腾讯云等)对学生都有优惠价，用阿里云举例一台标准的 ESC 服务器学生价 82/年，在可承受范围内。

综上所述，经济上可行。

2.1.2 技术可行性

本系统采用传统的 Browser/Server 架构，后端技术栈：Node.js，Koa，MySQL 关系型数据库；图像识别方面采用 TensorFlow.js + Mobilenet 预训练模型。

Node.js 是一个基于 Chrome V8 引擎的 JavaScript 运行时，基于事件驱动、非阻塞式 I/O 模型让 JavaScript 代码可以运行在服务端的开发平台[2]。Koa 是一款开源的被广泛使用于 Node.js 的 Web 后端开发框架，以其轻量简洁著称。MySQL 是一款属于 Oracle 公司开源的支持事务和主从架构的关系型数据库，被广泛的使用于 Web 开发之中。TensorFlow.js 是一个开源的由谷歌公司推出的基于硬件加速的 JavaScript 库,用于训练和部署机器学习模型。

前端方面采用 Vue.js + Vant 移动端 UI 组件库，均是在业界有过长期使用的开源框架，无论是稳定性还是周边生态 Vue.js 都极为丰富和完善[3]。

综上所述，技术上可行。

2.1.3 操作可行性

需要一台 Windows 操作系统的计算机,并且安装了 Node.js 和 MySQL 数据库。TensorFlow.js 的安装，在安装 TensorFlow.js 之前需要安装一个 windows-build-tools 和 node-gyp 用来编译原生 C++模块，除此之外还需安装 Vue-cli 用来构建前端项目。

综上所述，操作上可行。

2.2 需求分析

本系统的核心功能是帮助解决人们日常生活中类似于“这到底是可回收垃圾还是其他垃圾啊，这到底是厨余垃圾还是有害垃圾？”这样的疑惑，所以核心功能便是围绕这给垃圾提供分类信息而设计。

于是本系统主要实现了通过文字搜索和图片上传识别功能，所有的功能最终落脚点均是告诉用户该垃圾应该属于哪种分类：

1. 文字搜索：日常生活中常见的垃圾可以直接通过文字进行搜索，后端检索数据库中的数据做出判断。相较于上传图片识别，文字搜索操作简单，查询效率高。
2. 图片识别：点击上传图片，将图片文件转成 tensor 之后喂给模型进行预测，得到结果之后展示该图片中的垃圾在各个分类下的可能性。为了交互友好和界面美观，最后需要把图片文件转成一个 base64 的图片展示在界面上。

3 系统设计

本系统的设计目标是设计并实现一个可以给人们提供一系列获取垃圾分类信息的平台。通过搜集和处理有关垃圾分类的信息提取出对本系统有用的数据存放数据库，用于给文字搜索做数据支撑。通过网上爬取和整理有关垃圾分类的图片并对无用的数据过滤清洗之后保存于本地用来做图像识别模型的训练支撑。在开发过程中不断学习技术锻炼个人的编程能力以及对功能和业务的理解，不断站在用户的角度上思考问题，改进用户的使用体验。

根据上述上述的可行性分析和需求分析，本系统采用传统的 B/S（Browser/Server）架构，采用现阶段业界常见的前后端分离加 RESTFUL 风格化的 API 设计。前后端均选用 JavaScript 开发语言，前端采用 Vue.js 和 Vant 移动端 UI 组件库，后端采用 Node.js 的 Koa 框架。

系统环境安装包括：（1）安装 Node.js，本系统选用 10.24.0 版本。（2）安装 windows-build-tools 和 node-gyp 编译工具，然后安装 TensorFlow.js。（3）安装 Koa 框架和 MySQL 数据库的 Node.js 的支持库。（4）安装 Vue 项目的构建工具 Vue-cli 和 Vant 移动 UI 组件库。

3.1 系统功能设计

本系统主要实现的功能为文字搜索和图像识别，文字搜索使用后端提供的 API 接口用数据库中的数据检索，图像识别用上传的图片为图像识别模型预测。

文字搜索：用户通过搜索框中输入搜索关键字，前端携带关键字请求后端提供的 API 接口，后端收到请求之后拼接查询条件去数据库中查询数据，返回给前端再由前端渲染展示。用户点击渲染出来的各个卡片可以查看详细信息。

图片识别：用户点击上传图片按钮，选择一个图片文件同时加载 Mobilenet 模型和分类信息，将图片喂给模型预测后展示预测的结果，将图片文件转成 base64 位的图片显示给用户观看，美化界面同时也使得功能更加完整直观。

3.2 文字搜索模块

文字搜索模块通过系统后底部 Tabbar 第一项进入，输入想要检索的关键字，

由前端做合法性校验，过滤掉无效的输入,例如：空白字符、全是数字、全是各种符号等不合法的关键字，若输入关键不合法，将拦截掉请求提示用户再次输入搜索关键字，若合法则发起请求获取数据并展示，流程图如图 3-1 所示。

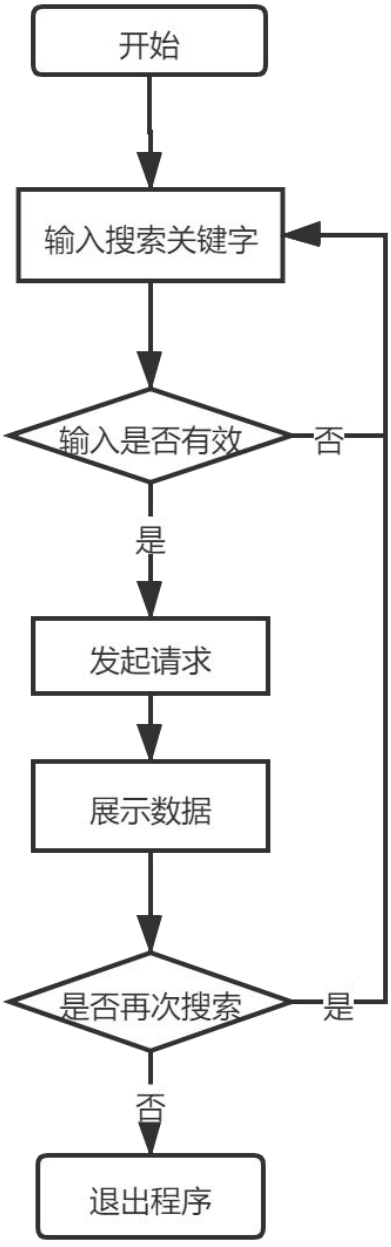

图 3-1　文字搜索流程图

3.3 图片识别模块

图片识别模块通过系统后底部 Tabbar 第二项进入，点击头部的上传图片按

钮，选择上传的图片文件，由于文件选择控件可以选择任意类型的文件，而显然对于图片识别模块上传的文件必须是图片才是合法的，于是在将图片转成tensor喂给模型预测之前需要对上传文件的类型做合法性校验，排除掉一些非法的文件例如：doc 文件，MP3 文件，text 文件等等。上传文件合法性校验通过之后将图片转成 tensor 喂给模型进行预测并展示预测结果，流程图如图 3-2 所示。

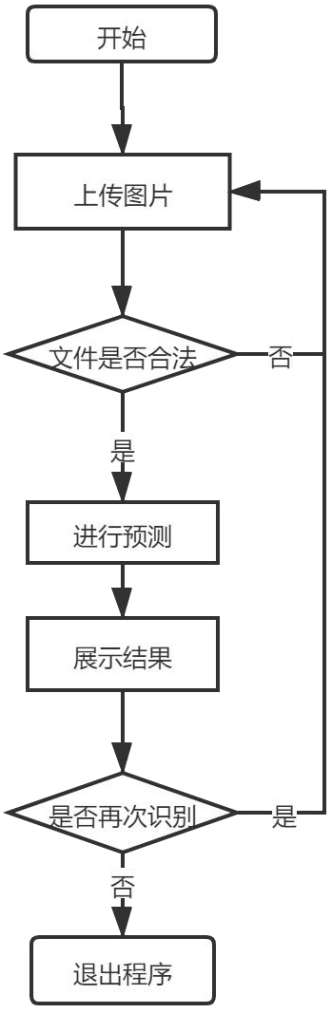

图 3-2　图片识别流程图

4　数据集搜集与数据库分析

本系统的目标是设计并实现一个可以给人们提供一系列获取垃圾分类信息的平台。而要完成这些功能，必不可少的就是数据的搜集，无论是训练模型所用的数据集，还是数据库中提供搜索用的数据都需要搜集整理。

4.1 数据集收集

图片识别所用到的模型是截断的 Mobilenet 预训练模型之后，在其后面加上自定义的隐藏层和输出层改造成的一个适用于本系统的一个垃圾分类模型[4]。定义好了模型之后需要喂给该模型训练数据集用于训练该模型。

首先在网上按照系统设计的类别去搜集和爬取各种各样的图片，归档整理之后保存下来，以便后续的使用。最终一共搜集到了四个大文件夹，对应着不同的垃圾类别，每个文件夹中有各自包含的垃圾名称及其图片，数据集一共有 246 种垃圾。图片数量接近 10 万张，所有文件的总体积约有 10GB，如图 4-1 所示。

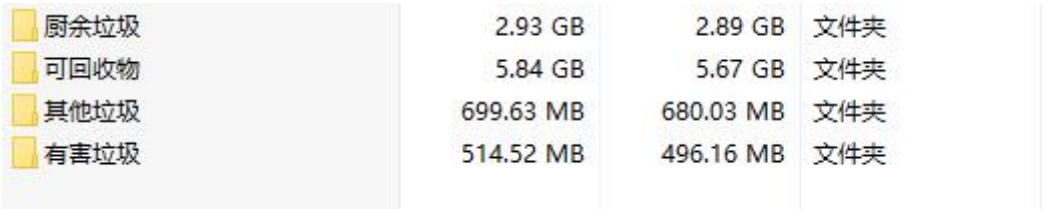

图 4- 1 训练数据集图

4.2 数据库分析

本系统文字搜索模块采用的是用关键字去数据库查询的方式实现，而数据库中的数据是预先搜集好的，相较于传统的数据库系统不同本系统数据库中的数据几乎属于纯静态的数据。

本系统仅采用了一张数据表 refuleclassify 表来存放垃圾分类信息。其逻辑结构如表 4-1。该逻辑结构表示垃圾和其分类的对应关系，理论上应该把为了满足数据库设计的三大范式[5]，应该把 category 字段单独成立一张数据表，但由于垃圾类别信息变动较小，且为了消除连表查询的性能损耗和简化 SQL 查询的复杂性故而并未拆解独立成表。

表 4-1 refuleclassify 垃圾分类表

序号	字段名称	字段类型	大小	允许为空	最大长度	备注
1	id	Int	11	否	11	垃圾 id
2	refusename	VarChar	20	否	255	垃圾名称

3	category	VarChar	20	否	255	属于分类

5 系统实现

本系统采用Node.js中简单轻量的Koa框架做后台技术支撑，图像识别的模型和预测使用TensorFlow.js实现，前端方面采用Vue.js和Vant移动端UI组件库实现。

5.1 开发技术栈

5.1.1 Node.js

Node.js 是一个基于 Chrome V8 引擎的 JavaScript 运行环境，使用事件驱动、非阻塞式 I/O 的模型，本质上就是为文件系统、数据库之类的资源提供接口。向文件系统发送一个请求时，无需等待硬盘（寻址并检索文件），硬盘准备好的时候非阻塞接口会通知Node。该模型以可扩展的方式简化了对慢资源的访问， 直观，易懂。并且使用Node.js使得前后端开发语言得到了统一，有利于代码风格和规范的一致性。

5.1.2 Koa

Koa 是由 Express 原班人马打造的，致力于成为一个更小、更富有表现力、更健壮的Web框架。使用Koa编写Web应用，通过组合不同的generator可以免除重复繁琐的回调函数嵌套，并极大地提升错误处理的效率。Koa不在内核方法中绑定任何中间件，它仅仅提供了一个轻量优雅的函数库，使得整个框架更加轻量以及在编写Web应用更加的得得心应手。

5.1.3 TensorFlow.js

TensorFlow.js 是一个用于使用 JavaScript 进行机器学习开发的库。使用 JavaScript 开发机器学习模型，并直接在浏览器或 Node.js 中使用机器学习模型，并且可以和 Python 中已经实现好的模型进行互转并使用，这让TensorFlow.js拥有了很多可供选择的模型，另外语言是JavaScript是的本系统的所有语言得到了统一有利于代码风格和规范一致。

5.1.4 Vue.js

Vue.js 是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue.js 被设计为可以自底向上逐层应用。Vue.js 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合，并且 Vue.js 的中文文档和社区完善且活跃，若是开发中遇见问题查找资料也更加容易。

5.1.5 开发环境

开发工具：Visual Studio Code

开发平台：Windows10 64bit

开发环境：Node.js 10.24.0

数据库：MySQL5.7.26

数据库管理工具：Navicat for MySQL

开发语言：CSS、HTML、JavaScript

调试工具：Chrome 浏览器、Postman

5.2 模型训练

使用 TensorFlow.js 训练模型，以便提供给前端使用该模型进行图片预测。

第一步：调用 train/data.js 加载训练集的图片数据转成 tensor，读取垃圾分类文件夹下 train 目录中的图片数据，并且根据 train 目录下的文件夹名字生成分类文件保存到 classes.json 文件中。注意，图片文件约有 10GB 为了防止内存溢出导致训练无法进行，建议使用 TensorFlow.js 中定义的 tf.tidy()方法防止内存不足而导致程序崩溃。

第二步：调用 tf.loadLayersModel()方法加载 Mobilenet 预训练模型，截取其中前 86 层，加上自定义的隐藏层和输出层，改造为一个符合本系统需求的垃圾分类模型，把第一步中加载并转换好的 tensor 数据喂给该模型进行训练。为了更好的训练效果建议打乱训练顺序进行训练[6]，等待训练结果。当训练完毕之后调用 model.save(`file://路径`)把训练好的模型保存到本地，以便后续给前端使用。训练过程如图 5-1 所示，训练结果如图 5-2 所示。

```
dropout (Dropout)            [null,1,1,256]            0
_________________________________________________________________
conv_preds (Conv2D)          [null,1,1,1000]           257000
_________________________________________________________________
reshape_2 (Reshape)          [null,1000]               0
_________________________________________________________________
act_softmax (Activation)     [null,1000]               0
=================================================================
Total params: 475544
Trainable params: 470072
Non-trainable params: 5472
_________________________________________________________________
Epoch 1 / 20
eta=0.0 ----------------------------------------------------------------------------
```

图 5-1　模型训练过程图

图 5-2　模型训练结果图

5.3 后端接口

后端主要提供文字搜索模块的数据支持，定义 Koa 的路由用来响应前端的请求。当接受到前端的请求之后提前请求携带的搜索关键字，根据 congfig 中提供的数据库连接参数连接数据库，根据搜索关键字模糊查询数据库中的数据，而后筛选过滤返回给前端展示，返回数据如图 5-3 所示。

```
[
  {
    "id": 213,
    "refusename": "沾了油污的报纸",
    "category": "可回收物"
  },
  {
    "id": 334,
    "refusename": "油渍的报纸",
    "category": "其他垃圾"
  },
  {
    "id": 1450,
    "refusename": "湿报纸",
    "category": "其他垃圾"
  },
  {
    "id": 2461,
    "refusename": "旧报纸",
    "category": "可回收物"
  }
]
```


图 5-3　返回数据示例图

5.4 前端实现

前端通过 Vue.js 和 Vant 组件库编写页面，通过 JavaScript 的 XMLHttpRequest 内置对象向后端发起 HTTP 请求访问数据，图片识别方面采用 TensorFlow.js加载上述训练好的模型,将用户上传的图片喂给该模型进行预测。

5.4.1　文字搜索

页面采用 Vant 的 search 组件接收用户的输入用，在发起 HTTP 请求之前对用户的输入进行合法性校验，如果校验未通过则提示用户重新输入，如果校验通过则使用 JavaScript 的 XMLHttpRequest 内置对象[7]发起 HTTP 请求后端接口，用 collapse 组件把数据展示给用户，效果如图 5-4 所示：

图 5-4　文字搜索示例图

5.4.2　图片识别

图片识别当用户点击上传图片按钮，选择一张图片之后把图片转换成 tensor，使用 TensorFlow.js 的浏览器版本加载上述训练好的模型，调用模型的 predict()方法把该 tensor 喂给模型预测，得到的预测结果，转成百分制的得分，展现给用户，效果如图 5-4-2 所示：

图 5-5　图片识别截图

6　程序测试

程序测试是检验程序合格性的重要手段，常见的测试方式有功能测试，模块测试，压力测试，黑盒测试，白盒测试，线上回归[8]等等。要想保证一个程序的稳定性和健壮性，在投入发布到线上环境之前一定要经过充分的测试，根据测试结果加以改进。

6.1 测试结果

本系统经过严格的多轮测试，在开发过程中经过了充分的自测并且修改了开发中已发现的bug，在后端的接口开发过程中使用了Postman接口调试工具进行接口的测试，模型预测方面也经过了充分的验证，其在各个分类下的图片预测结果都较为理想。

前端界面在各个不同版本和内核的浏览器下都有良好的表现和兼容，各个不同的机型下都能正常运行，尽管在网络较为不稳定的情况下依然能较为快速的响应或给出友好的提示。

综上所述，测试通过。

7 总结

通过本次毕业设计融会贯通了以前所学的知识，同时也了解到了最新最前沿的技术，对自己的开发能力有了不小的提升。整个系统涉及到了Node.js、Vue.js、MySQL数据库、基于TensorFlow.js的图像识别等等，拓宽了前端技术视野，使得前端也能设计和开发一些有关神经网络方面的东西，把神经网络机器学习等技术从后端迁移到了浏览器前端，相信未来神经网络人工智能等领域也会有前端的一席之地。

开发方面本次系统采用前后端分离的开发模式加RESTFUL风格化的API设计。JavaScript方面使用了更新、更简介、更直观的ES6语法，如：箭头函数、变量声明、Async/Await写法、回调地狱的Promise解决方案[9]。前端采用MVVM的设计模式[10]将视图层和模型层的逻辑解耦，方便了后期维护和扩展。

选择做这样一个垃圾分类的APP，一方面是为了学习和实践最新的技术栈，另一方面也是为了做一个有意义能够用自己所学的知识为环保出一份力，尽管它功能还很单一，但我会在后期慢慢优化它，提高它的识别率，增加它的数据集等等。

参考文献：

[1] 习近平.谈新时代坚持和发展中国特色社会主义的基本方略[R] .北京:新华网,2017 1-3

[2] 朴灵.深入浅出 Node.js,[M].北京: 人民邮电出版社, 2018.8

[3] 张耀春,黄轶等.Vue.js 权威指南,[M].北京: 电子工信出版社, 2016.9
[4] 黄跃珍,王乃洲等. 基于改进型 MobileNet 网络的车型识别方法[J]. 电子技术与软件工程, 2019 (1): 27
[5] 王珊,萨师煊.数据库系统概论（第四版）[M].北京:高等教育出版社,2018.2
[6] 董子源.基于深度学习的垃圾分类系统设计与实现[D].北京:中国科学院大学(中国科学院沈阳计算技术研究所),2020.
[7] [美] Nicholas C.Zakas. Professional JavaScript for Web Developers (3rd Edition) [M].New Jersey: John Wiley & Sons, Inc, August 2019
[8] Ron Patton. 软件测试（第 2 版）[M].北京:机械工业出版社，2017.9
[9] 阮一峰. ES 6 标准入门（第 3 版）[M].北京:电子工业出版社，2019.7
[10] 张容铭. JavaScript 设计模式 [M].北京:人民邮电出版社，2015.5