分类号：TP315UDC：D10621-408-(2019)2154-0

密　级：公开编号：**2015053025**

成都信息工程大学

学位论文

基于大数据的图片云端管理系统设计与实现

|  |  |
| --- | --- |
| 论文作者姓名： | 李莘 |
| 申请学位专业： | **数字媒体技术** |
| 申请学位类别： | 工学学士 |
| 指导教师姓名（职称）： | 鄢田云（**副教授**） |
| 论文提交日期： | **2019年5月25日** |

基于大数据的图片云端管理系统设计与实现

**摘　要**

由于智能手机的普及和社交网络的发展，用户每天都会往自己的终端添加图片，这些图片日积月累，逐渐变得分散和杂乱，让用户难以管理和利用图片素材。

本文利用图像分类、图像检索等前沿技术可以建立一个基于大数据的图片云端管理系统，最终为互联网用户提供一个智能化、安全和体验良好的图片管理平台。系统开发过程中集成了一些开源方便的开发工具，以MySQL作为数据库，采用Django + MVT架构建立基于B/S模式的云端系统，以及利用TensorFlow实现图像特征提取。本系统主要有以下功能，第一，用户在前端注册和登录，以及修改用户信息；第二，用户在前端创建图册，上传图片，设置访问权限；第三，后端对用户上传的图片素材进行分类管理；第四，通过关键字或样本图片匹配搜索图片；第五，在网络上抓取指定内容的相似图片。第六，系统会根据用户的社交行为和爱好，有针对地向其推荐个性化图片。

**关键词：**图片管理；深度学习；Django；TensorFlow；CNN

Design and Implementation of

Image Cloud Management System based on Big Data

**Abstract**

Due to the popularity of smart phones and the development of social networks, users will add pictures to their terminals every day. Over time, these pictures gradually become scattered and disorderly, making it difficult for users to manage and use pictures.

In this paper, image classification, image retrieval and other cutting-edge technologies were used to establish a cloud-based image management system based on big data. It could eventually provide Internet users with an intelligent, safe and well-experienced image management platform. Some open source and convenient development tools are integrated in the system development process. MySQL is taken as the database, Django + MVT architecture is adopted to establish a cloud system based on B/S mode, and TensorFlow is used to implement image feature extraction. This system mainly has the following six functions. First, the users can register and log in the front end and modify the user information. Second, the users in the front end can create albums, upload pictures and set access rights. Third, the backend of the users can classify and manage pictures which are uploaded by users. Fourth, the users can search images by matching keywords or sample images. Fifth, according to the users’ specified contents, the system can capture similar images for users on the web. Sixth, the system will recommend personalized pictures to users according to their social behaviors and hobbies.

**Key words:** photos management; deep learning; Django; TensorFlow; CNN

**目 录**

论文总页数：40页

[1 引言 1](#_Toc10729047)

[1.1 图片管理系统的背景 1](#_Toc10729048)

[1.2 图片管理系统的国内外研究现状 2](#_Toc10729049)

[1.3 图片管理系统的目标和意义 3](#_Toc10729050)

[1.4 图片管理系统的实现方法 3](#_Toc10729051)

[1.5 本文的组织结构 4](#_Toc10729052)

[2 需求分析及项目方案 5](#_Toc10729053)

[2.1 需求分析 5](#_Toc10729054)

[2.1.1 系统的总体需求 5](#_Toc10729055)

[2.1.2 功能需求分析 6](#_Toc10729056)

[2.1.3 非功能需求分析 9](#_Toc10729057)

[2.1.4 系统约束条件 10](#_Toc10729058)

[2.2 项目方案 10](#_Toc10729059)

[2.2.1 项目特色 10](#_Toc10729060)

[2.2.2 Django MVT架构简介 12](#_Toc10729061)

[2.2.3 CNN算法简介 13](#_Toc10729062)

[2.2.4 基于内容的图片检索技术简介 13](#_Toc10729063)

[2.2.5 网络爬虫技术简介 14](#_Toc10729064)

[2.2.6 开发工具和运行环境 15](#_Toc10729065)

[2.2.1 项目实施计划 15](#_Toc10729066)

[3 项目设计及实现 16](#_Toc10729067)

[3.1 数据库设计 17](#_Toc10729068)

[3.1 功能模块的设计与实现 19](#_Toc10729069)

[3.1.1 系统整体框图 19](#_Toc10729070)

[3.1.2 用户系统模块的设计与实现 20](#_Toc10729071)

[3.1.3 图册管理模块的设计与实现 23](#_Toc10729072)

[3.1.4 图片管理模块的设计与实现 26](#_Toc10729073)

[3.1.5 社交系统模块的设计与实现 29](#_Toc10729074)

[4 特殊问题及解决方案 30](#_Toc10729075)

[4.1 Django时间设置问题 31](#_Toc10729076)

[4.1.1 问题描述 31](#_Toc10729077)

[4.1.2 解决方案 31](#_Toc10729078)

[4.1.3 结果 31](#_Toc10729079)

[4.2 前端无法访问上传的图片的问题 31](#_Toc10729080)

[4.2.1 问题描述 31](#_Toc10729081)

[4.2.2 解决方案 32](#_Toc10729082)

[4.2.3 结果 32](#_Toc10729083)

[4.3 CNN训练模型内存不足问题 32](#_Toc10729084)

[4.3.1 问题描述 32](#_Toc10729085)

[4.3.2 解决方案 32](#_Toc10729086)

[4.3.3 结果 33](#_Toc10729087)

[4.4 部署系统路径错误问题 33](#_Toc10729088)

[4.4.1 问题描述 33](#_Toc10729089)

[4.4.2 解决方案 33](#_Toc10729090)

[4.4.3 结果 33](#_Toc10729091)

[5 结果测试及性能分析 34](#_Toc10729092)

[5.1 测试概要 34](#_Toc10729093)

[5.1.1 测试环境 34](#_Toc10729094)

[5.1.2 测试范围 34](#_Toc10729095)

[5.2 性能分析 36](#_Toc10729096)

[结 束 语 36](#_Toc10729097)

[参考文献 37](#_Toc10729098)

[致　　谢 39](#_Toc10729099)

[声　　明 40](#_Toc10729100)

# 引言

## 图片管理系统的背景

我国改革开放带来智能手机和社交网络快速发展的同时，国内的网民的规模数量也一直保持着平稳增长的趋势，截至2018年12月，我国网民规模增加到8.29亿[1]。作为主要信息传播媒体之一的数字图片，从过去几年中呈现出爆炸式增长，使得用户对图片管理的需求持续增加。在社交网络方面，以微信朋友圈、QQ空间和微博为主的社交平台，每天都会产生无数的图片资源。除了将图片发布到社交平台，用户每天都会通过拍照和下载图片往自己的终端添加图片，这些图片日积月累、反复移动使得用户难以保管和再次加工利用这些图片；另一方面，由于图片种类变得越来越多，用户寻找精美的图片和针对性的图片的需求总是得不到满足。在这个大数据时代，人们对图片管理的智能化和个性化需求不断提高，每个人针对的图片种类也不尽相同，可见，人们对图片管理的需求仍然在上升。

为了满足用户的图片管理需求，这个图片云端管理系统，需要提供以下功能。第一，用户选择并预览本地图片，最终确认上传到云端；第二，系统需要对用户的图片实时进行更新，完成分类；第三，允许用户对图片进行自定义标签，使用户根据自己的需求进行分类；第四，用户之间可以通过收藏图片、点赞和评论获得社交需求；第五，系统能够以用户指定的感兴趣的图片为目标，用爬虫从网络抓取类似的图片，并能保存到自己的图片库中；第六，系统能够根据用户的爱好定时向他推荐感兴趣的图片。最终给用户提供一个安全可靠、智能化、基于社交和面向互联网的图片云端管理平台。其中，图片管理系统的智能化一直以来都是整个系统的关键。

图像分类算法和基于内容的图像检索算法[2]是实现图片管理系统智能化的核心技术，而图像分类和图像检索的一个核心问题就是图像的特征提取[3]。图像特征就是描述图像的具有代表性的一组数据，这些描述图像的数据应该是简单的、可用于计算的。良好的图像特征应该不受光线、噪点、几何形变的干扰，并可以为后续计算图像相似度提高准确率。传统的图像特征提取方法，例如SIFT (Scale Invariant Feature Transform)[4]和HOG (Histogram of Oriented Gradient)[5]，其滤波算子通常是根据开发人员的经验和先验知识设计好的，并且在提取图像特征的过程中总是不变化的，所以具有一定自身的局限性，计算图像相似度无法达到较高的准确率。在2012年，基于深度学习的AlexNet网络模型获得ImageNet竞赛冠军，使人们重新重视神经网络，更多更深的神经网络被提出[6]。以卷积神经网络为代表的图像建模算法的出现，以及计算机硬件的性能的不断提升，能够有效地实现数字图像的特征提取，为进一步实现图像分类和图像检索打下坚实的基础。

## 图片管理系统的国内外研究现状

数字图片一直以来是互联网主要的信息传播媒介之一，也是一种典型的非结构化数据。由于数字图片是最常见的媒体资源，当今互联网中数字图片的信息量已经达到海量规模，而且图片内容也变得丰富多样。大数据时代对传统图片管理系统提出了新的挑战，传统的图像处理虽然拥有较完善的理论和较丰富的经验，但是缺乏应对大规模图像数据处理和提取信息的能力。传统的基于内容的图像搜索算法，提取的图像特征一般是低层面的特征，这是因为传统图像特征提取方法的滤波算子通常是根据开发人员的经验和先验知识设计好的，并且在提取图像特征的过程中总是不变化的，所以传统的图像特征提取方法对复杂的图像难以提取高层次、更抽象的图像特征，从而使接下来的相似度计算难以达到很高的准确率，泛化能力也不够强[7]。

最近几年，人工智能和大数据的计算机技术取得了突破性的进步，其中以卷积神经网络为代表的图像建模算法表现尤为突出，使其在图像分类和图像检索领域运用越来越广泛。基于卷积神经网络的图像分类能够关注高层次的更抽象的图像特征，这种方法的关键在于随着训练次数的增加而变化卷积核，因此，基于卷积神经网络的特征提取是利用神经网络的自主学习得到的，从而能够准确又高效地提取图像特征，大大的提高了实战中的准确率。因为图片分类和图片检索的核心是准确有效的提取出图像特征，所以，卷积神经网络算法是实现目标系统的图片分类和图片检索功能的最佳方案。

由于图片管理系统在社会一直存在需求，以前就已经有了。不过以前的图片管理系统功能比较简单，主要以单机为主，浏览和管理的图片仅仅是本地图片[8]。随着B/S模式的软件开发越来越成熟，图片管理系统逐渐从单机模式过渡到B/S模式。国内存在的基于B/S 模式的有百度图片管理系统，国外知名图片管理系统则有Google相册。相比于以前的单机图片管理系统，基于B/S模式的图片管理系统最明显优势是方便可靠，同步数据，用户不用担心图片丢失，且随时可以通过网络访问。例外，Web形式的图片管理系统更易于实现社交功能，要知道，在这个互联网的大环境下，社交永远是用户最基本的需求。

社会上虽然已经存在许多图片管理系统，但是这些应用系统基本上只能局限于用户本身的图片数据，缺乏开放性，即用户只能对自己上传的图片进行操作，没有扩展到整个互联网，也不能使不同用户之间产生社交行为。为了体现系统的开放性，我将加入以下几个特色功能，第一，用户可以通过指定图片目标在网上抓取相似的图片，并将其更新到自己的图片库中；第二，系统会根据用户兴趣和习惯推荐更多的相似图片。本项目将通过加入这些功能，使其变得更加具有开放性和个性化。

## 图片管理系统的目标和意义

图片云端管理系统的目标是一个给用户提供一个安全可靠、智能化、基于社交和面向互联网的图片管理系统。在这个大数据时代中，图片数据的产生和传播越来越迅速，图片数据变得越来越难管理，而本论文实现的图片云端管理系统为普通用户和特殊用户（专业图片设计师）提供了一款图片分类、图片搜索、网络图片抓取和个性化推荐的图片素材管理系统。

为了满足用户需求，最终实现的图片云端管理系统包括以下功能，第一，用户选择并预览本地图片，最终确认上传到后端；第二，系统需要对用户的图片实时进行更新，完成分类；第三，允许用户对图片进行自定义标签，使用户根据自己的需求进行分类；第四，用户之间可以通过收藏图片、点赞和评论获得社交需求；第五，系统能够以用户指定感兴趣的图片为目标，用爬虫从网络抓取类似的图片，并能保存到自己的图片库中；第六，系统能够根据用户的爱好定时向他推荐感兴趣的图片。

图片云端管理系统的意义是运用大数据和人工智能技术帮助用户有效管理大量杂乱无章的图片数据，使其从中体验到人工智能所带来的方便性。对于专业的图片设计师，该系统明显可以提高他们的创作的效率，让他们把更多的时间花在创作上，而不是管理图片素材上。

通过图片云端管理系统提供的功能，普通用户可以放心地将自己的图片交给这个系统管理，不用担心图片丢失，也可以通过社交获得良好的体验。对于专业图片素材设计人员，在工作过程中利用图片云端管理系统，可以快速查找相关的图片素材，也可以利用系统的分类功能有效避免图片文件零散、过于杂乱，难以获得所需的图片素材，还能利用系统的网络图片抓取功能实时地更新图片素材库。在本项目完成的这个图片云端管理系统中，设计师可以根据需求将图片分类，输入图片文件名或者选择一个图片模板，系统会快速找到目标图片或相似图片；设计师只需要在本系统内请求想要的素材，不需要费时地打开第三方浏览器搜索，本图片素材管理系统会主动在网络上进行图片素材爬取，并给设计师提供动态更新的、个性化的推荐。

## 图片管理系统的实现方法

图片云端管理系统的实现不仅集成了许多开源免费的支持软件和开发工具，而且涵盖了现代广泛运用的大数据和人工智能技术。实现图片云端管理系统的开发工具和支持软件包括Windows 10操作系统、Django框架、Python3编程语言、Pycharm代码编辑器、Chrome浏览器、Git版本管理器、MySQL数据库和Apache服务器。实现图片云端管理系统所涉及的关键技术有基于卷积神经网络的图像特征提取、分类算法、图像检索和静态网络爬虫技术。

图片云端管理系统的总体结构为B/S模式，前端由HTML + JavaScript开发实现，后端则使用Django + MVT（Model View Template）框架开发实现，以MySQL数据库作为结构化数据的存储方式，无结构化的图像数据则直接以文件形式存储，并把图片的存储路径保存在数据库中。

图片云端管理系统的开发语言为Python3，该编程语言最大的优势是简单高效，使得开发可以快速迭代，并且能专注于系统的功能需求实现，而不是把时间花在技术探索中。Python3作为脚本语言，具有良好的平台移植性，不仅可以用在Windows操作系统上，也可以用在Linux系统上及其它操作系统上，这给后期的系统部署带来便利性。

PyCharm是知名的Python3集成开发环境，已经集成了Django框架。Chrome浏览器主要用于调试图片云端管理系统，而Git是一个高效的分布式版本管理系统，对有效管理本项目带来方便。

图片云端管理系统的开发环境为Windows 10，Windows 10是微软于2015年发布的独立的操作系统版本，在人机交互、易用性和性能方面有了极大的提升，因此是开发和调试图片云端管理系统的最佳平台。本系统开发并调试完后，最终部署到Ubuntu云端服务器。

为了实现图片云端管理系统的智能化，即图片自动分类和基于内容的图片搜索功能，本项目在第一阶段使用TensorFlow编写训练卷积神经网络的代码，实现卷积神经网络提取图片特征，从而得到代表每一类图片的矢量特征，并且保存到数据库，在第二阶段，计算前向传播的预测值，最终达到图片自动分类的目的，对于待检索图片，依然使用前向传播得到矢量特征，然后计算该向量和数据库中所有矢量特征的距离，对距离进行排序便能得到最接近的目标图片 [9]。

## 本文的组织结构

本文将用五个章节来说明图片云端管理系统的实现过程，阐述思路是按照软件工程过程进行的。以下是每一章的主体内容。

第一章，主要探究了图片云端管理系统的背景和国内外的研究现状和趋势，同时也说明了本项目要实现的目标和意义，以及提出了实现的路线。

第二章，分析了用户对现代图片管理的真实需求，再从需求出发，确定了图片云端管理系统的功能模块，并对这些功能模块分别提出了具体实现方案。

第三章，阐述了实现图片云端管理系统的各个模块的具体设计和实现，首先展开系统主体功能模块，然后对每个主体模块分成子模块进行设计和实现。

第四章，罗列了在设计和开发图片云端管理系统的过程中所遇到的各种问题，以及每个问题所对应的解决办法。

第五章，给出了图片云端管理系统的测试结果和最终性能分析。

# 需求分析及项目方案

本章重点分析用户需求，第一节确定图片云端管理系统的详细功能模块，第二节说明本项目的开发工具和运行环境，以及制定本项目的实施计划。

## 需求分析

需求分析所做的工作是深入描述图片云端管理系统的功能和规则，弄清楚本系统设计的限制，以及系统内部元素的接口和数据流，用一句话概括就是，需求分析主要是确定开发图片云端管理系统的性能、功能、界面和数据等要求[10]。

### 系统的总体需求

图片云端管理系统的主要服务对象为互联网用户，也包括图片素材设计师和摄影师等特殊用户。他们有共同的需求，就是需要一个智能化的系统为他们有效地存储和管理自己大量的图片，因此，图片云端管理系统应具备以下六大功能。

（1）用户账号管理功能

因为本系统需要存储大量图片，所以需要通过注册登录对每个用户进行唯一标识，这样做可以使用户在本系统中有独立的空间，避免图片数据交叉，有利于本系统管理每个用户的数据，也保障了本系统的安全。例外，用户还需要对自己的账号信息进行修改，账号信息有昵称、头像、个人说明、手机号和密码。

（2）用户对图册的操作功能

本系统提供用户上传数据的接口，用户可以创建有具体主题的图册，并往指定图册上传图片。除了创建图册，用户还能对指定图册进行编辑修改或删除。

（3）图片分类功能

本系统对图片进行分类的目的是合理组织大量图片，使用户能够方便浏览、查找和使用图片。系统对图片分类有两种方式，第一种是通过用户输入的标签和类别进行分类图片，这种方式是按照用户自己的意愿对图片进行分类，第二种是基于图片内容进行分类图片，这种方式是系统后端利用图片分类模型对用户上传的图片进行预测分类，两种方式互不影响，可同时存在，只是在前端展示图片的结果不一样。

（4）图片搜索功能

图片搜索功能是本系统的重要功能，是为了满足用户查找自己的资源和外部图片资源的需要。同样的，对应图片的两种分类方式，图片搜索也分为基于标签和基于内容的搜索方式。

（5）图片推荐功能

为了更大体现本系统的智能化和提高用户体验，本系统可以根据用户的兴趣爱好向其推荐相似的图片，并且允许用户把这些图片添加到自己的图库中。

（6）社交功能

在互联网的大环境下，社交永远都是用户不可或缺的需求，因此本系统必须包含该功能。用户社交主要包括以下几点，第一，用户浏览其他用户的图册或图片时给予肯定，这种肯定包括通过点赞和收藏图片，以及关注图册的作者。第二，不同用户对同一图册或图片进行讨论和评论。

以上描述的图片云端管理系统的六大模块，可以通过用例图更加形象地表示，如图2-1系统总体用例图所示。



图2‑1 系统总体用例图

### 功能需求分析

经过上一节分析可知，图片云端管理系统需要为用户提供六大功能，分别是用户的账号管理、图册管理、图片分类、图片搜索、图片推荐和社交功能，本节将从这六大功能仔细展开分析。

（1）用户账号管理功能需求

用户账号管理功能作为图片云端管理系统的入口，需要提供注册、登录、修改用户信息和退出系统的功能。注册页面，需要用户输入昵称、密码和手机号，注册成功后用户信息会保存到MySQL数据库中。登入页面，需要用户输入昵称和密码，系统后端收到登录请求后需要对其进行验证。已注册的用户拥有个人空间，个人空间是展示用户信息的页面，用户在这个页面能够修改自己的信息。用户在登录状态能够退出登录，退出系统意味着用户的数据缓存将被清除，并且不能使用系统的大部分功能。该功能需求可用图2-2用户账号管理用例图表示。



图2‑2 用户账号管理功能需求用例图

（2）图册管理功能需求

在用户的个人空间中，除了用户信息，还有用户的图册。用户可以创建图册并且往图册上传图片，用户需要输入图册的标题，添加标签和类别，用户也可以选择添加图册的说明文字。除了创建图册，用户还能够浏览和修改图册的基本信息，并且可以增加新的图片和删除已包含的图片，最后，还能够删除选中的图册。该功能需求可以通过图2-3图册管理用例图表示。

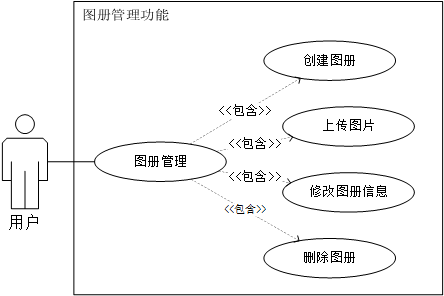


图2‑3 图册管理功能需求用例图

（3）图片分类功能需求

为了帮助用户合理组织图片，图片云端管理系统提供了图片分类功能。图片分类的方式有种，第一种方法需要用户的参与，即需要用户输入图片所在图册的类别和标签，其中类别是必须输入的信息且只有一个值，而标签可以为空或多个。第二种方法则是系统通过图像分类算法进行分类的，这种分类结果由于是从系统角度判别的，所以可以作为参考，也可以和第一种方法形成互补。该功能需求如图2-4 用例图所示。

（4）图片搜索功能需求

图片搜索功能是为了帮助用户方便快速找到所需图片，图片搜索分为两种方式，第一种是在输入框输入关键字，关键字可以是图片名字，也可以是类别和标签。第二种方法是利用基于图片内容的图像搜索算法实现的，简称以图搜图[11]，利用本系统的以图搜图功能，用户能够根据选择的目标图片获得更多的相似图片。该功能需求如图2-4 用例图所示。

（5）图片推荐功能需求

本系统中的图片推荐功能是系统本身的行为，推荐的依据是用户的图片浏览历史，以及用户社交过程中点赞和收藏的行为记录，根据这些社交行为便能推断出用户感兴趣的图片。本系统允许用户将系统推荐的图片添加到自己的图库中。该功能需求如图2-4 用例图所示。



图2‑4 图册分类搜索推荐用例图

（6）社交功能需求

社交是用户最基本的需求，所以本系统需要一个基本的社交功能。用户社交主要包括以下两点，第一，用户浏览其他用户的图册或图片时给予肯定，这种肯定包括通过点赞和收藏图片，以及关注图片的作者。第二，不同用户对同一图册或图片进行评论和回复。该功能可用图2-5用户社交功能需求用例图表示。



图2‑5 用户社交功能需求用例图

### 非功能需求分析

本小节主要说明图片云端管理系统的非功能需求，主要包括前端页面响应时间、用户的隐私保护、用户的图片版权问题、后端存储空间的大小和以图搜图的准确率。

（1）前端页面响应时间的需求分析

由于图片云端管理系统是基于B/S模式的Web系统，所以需要多多考虑前端获得响应的时间长短。响应时间包含了传输时间和系统处理请求的时间，而用户的浏览器从接收完所有字节开始到浏览器加载完页面，运行完所有JavaScript程序，呈现给用户的这段时间是不统计的，这部分属于页面渲染性能，需要利用专门的工具进行分析诊断。传输时间主要取决于用户终端接入的宽带和整个网络时延，所以响应时间主要和后端处理请求的时间长短有关[12]。为了提高本系统的用户体验，响应时间需要在网络状态良好的情况下不超过0.4秒，达到这一目标需要后端做缓存处理，可以用后端会话系统缓存用户大部分常用数据。

（2）用户隐私保护的需求分析

虽然图片云端管理系统可以满足用户的社交功能，允许用户访问其他用户，但是有些用户比较注重自己的隐私，不想将自己的图片暴露给其他用户，所以本系统必须能够让用户设置个人空间和图册的访问权限，访问权限的等级分为三个等级，分别是所有人可以访问、只有关注的人可以访问和仅自己访问。

（3）用户的图片版权保护的需求分析

当今社会，大家的版权意识越来越突出，本系统必须尊重用户的图片的版权。图片云端管理系统的用户有普通用户，也有专业的图片设计师和摄影师等特殊用户，对于后者，他们会关心图片的版权问题，对此，本系统的解决办法是，只允许用户收藏和浏览其他用户的图册，而不允许用户下载其他用户的图片。

（4）后端存储空间的大小

由于图片云端管理系统要存储大量的图片数据，本系统需要大量的存储空间，增加存储空间的方案有，利用分布式系统存储图片数据、增加存储硬件数量或空间大小和定时清理长时间没有登录的用户的缓存数据。

（5）基于图片内容搜索的准确率高低的需求分析

基于图片内容的图片搜索，简称以图搜图，是本系统的一大特色，这种方式搜索的准确率的高低将直接影响用户的体验，所以其准确率需要达到80%以上，准确率越高越能体现本系统的智能化。

### 系统约束条件

图片云端管理系统涉及到的系统约束条件主要有开发工具的授权与开源问题和相关网络法律法规。

（1）开发工具的授权与开源问题

为了降低开发成本和避免侵犯需授权的开发工具软件和代码的版权，在开发本系统的过程中，规定全部开发工具都要求使用免费开源的，或者购买下来，实现系统的核心功能的代码也必须由开发人员亲自编写，不得粘贴其他非开源的代码。对于本项目的所有代码和开发成果，开发人员不得共享或者泄露。

（2）相关互联网法律法规约束

中国是一个法治国家，在互联网之内也有完善的相关网络法律法规，例如《网络安全法》、《信息网络传播权保护条例》等。本系统坚决不允许存在危害网络安全的行为，也不允许用户制造和传播暴力、色情等不良信息。

## 项目方案

本节将先后介绍图片云端管理系统的项目特色、项目所涉及到的核心技术、开发工具与运行环境和本项目的实施计划。

### 项目特色

本小节从图片云端管理系统的功能特色和技术特色来阐述本项目的特色。

（1）图片云端管理系统的功能特色

相对于国内外的其它图片管理应用，本系统的功能特色包括社交功能、图片自动分类、基于内容的图片搜索（以图搜图）和图片推荐功能。

1. 图片管理系统的社交功能

虽然社会上已经存在一些图片管理应用，但这些应用有些是单机应用，或者是非单机应用却没有社交功能，使用户仅仅局限于自己的空间，不能与其他用户进行交互。而本系统已经提供社交功能，使用户能够浏览其他用户的图册，并且能通过点赞和收藏来表达喜欢和肯定，以及能对图册进行评论或回复评论。

1. 图片自动分类功能

用户上传本地图片后，系统不仅可以根据用户输入的标签来进行图片分类，也会利用卷积神经网络算法对用户的图片进行自动分类。

1. 基于图片内容的搜索（以图搜图）

本系统不仅能以图片名字和标签作为关键字搜索目标图片，也可以通过图片的内容进行搜索。当用户浏览到一张喜欢或需要更多类似的图片时，能够选择当前图片作为样本图片，然后在网络上搜索相似的图片，结合网络爬虫技术能将这些相似图片保存到自己的图片库当中。

1. 图片推荐功能

本系统的图片推荐功能属于系统行为，并且基于系统的社交模块，系统可以根据用户的社交动作推送更多相似的图片。通过社交模块，系统会记录用户在使用系统的过程中浏览图片和图册的行为，以及记录用户点赞和收藏图片和图册的行为，利用这些历史行为系统就能向用户推荐其感兴趣的相似图片。

（2）图片云端管理系统的技术特色

在开发本系统的过程中，除了利用Django的MVT架构，还使用了一些人工智能算法，包括CNN分类算法、基于内容的图片搜索和网络爬虫。

1. 开发语言为Python

本系统使用Python编程语言，采用Django的MVT（Model View Template）架构来搭建整个系统的框架结构，相比于Java + MVC（Model View Controller），Django + MVT架构开发速度更快，更方便，因为该框架使用Python编程语言，把数据库的SQL语句操作全部封装成API，只需调用便能快速访问数据库，利用Template技术还可以使前端的HTML组件得到复用，大大的提高了开发人员的效率。

1. CNN算法的运用

CNN[13]特征提取算法的目的是提取图片的矢量特征，用于实现之后的图像分类和图片搜索。基于CNN的图像特征提取方法相比于传统的特征提取方法，例如SIFT和HOG算法，具有自学习的能力，能够提取高层次、更加抽象的图特征。

1. 图片搜索算法的运用

图片搜索算法用于实现本系统的以图搜图功能，首先利用CNN特征提取算法提取样本图片的矢量特征，然后使用欧氏距离计算其与数据库中各个类别的矢量特征的距离，即相似度，通过最大相似度即可找到目标图片。

1. 网络爬虫的运用

网络爬虫技术主要用于配合图片搜索算法来爬取相似图片，并将这些相似图片保存到用户自己的图库中。

### Django MVT架构简介

图片云端管理系统采用Django MVT[14]搭建整个系统的前端和后端，Django MVT（Model View Template）是一个由Python实现的、开源的Web开发框架，注重代码复用，强调DRY（Don’t Repeat Youself）原则，可以帮助开发者快速搭建复杂的网站。之所以不使用传统的MVC（Model View Controller）架构，是因为MVC会增加代码量，相应地也会增加软件开发的成本，系统结构和实现的复杂性也会大大增加。使用Django 的MVT框架可以快速搭建后台，利用Template技术可以复用前端HTML组件，大大提高开发效率。由于MVT框架已经封装了数据库，访问数据库只需要调用封装好的API而不用反复编写SQL语句，所以对数据库的操作更加方便快捷，从而可以有效对数据增删改查，提高数据层的执行效率。以下详细介绍Model、View和Template三层架构。

（1）Model（模型）

Model层主要实现把数据库操作封装成API，把数据库里的元祖封装成Python对象，访问数据只需调用API就能做到和SQL语句一样的效果，因为Model层会根据Python对象的类型映射到数据模型，反过来，SQL查询到的结果转换为Python对象，然后直接访问对象的属性。例外，通过修改Django的数据库配置就可以轻松更换数据库。

（2）View（视图）

View层的作用是接收前端的请求、处理请求和返回处理结果，所以视图层也可以称为业务逻辑层。View层处理的业务逻辑主要包括处理表单、处理资源请求和后端参数的更改。

（3）Template（模板）

Template层的作用是致力于表达前端的外观，而不是业务逻辑，实现了业务逻辑和显示内容的分离。一个视图可以使用任意一个模板，一个模板也可以提供多个视图使用，这样便可以快速生成HTML页面。

### CNN算法简介

图片云端管理系统的图片自动分类功能和以图搜图功能都用到了CNN，在本项目中， CNN主要用于构建图片分类模型和提取图片的矢量特征。CNN全称Convolutional Neural Networks，中文名为卷积神经网络，是最近几年研究火热、运用广泛的深度学习算法，其在图片分类领域具有很好的表现能力，也可用于提取图像特征。

传统的图片分类，例如SVM（Support Vector Machine）和BP（back propagation）神经网络，有以下不足。第一，前期需要进行大量预处理，包括增强、复原、压缩和分割[15]；第二，权值太多导致计算量太大；第三，滤波算子是人为设计好的导致特征提取不能获得高层次的图片特征。而CNN作为革命性的图像分类算法已经基本解决以上三个问题，其优点有权值减少、局部连接和权值共享等，而且大大地简化了图片的预处理步骤，甚至能够直接将原始图片投入到训练模型中。

CNN模型的结构[13]一般分为三层，依次为输入层、隐藏层和输出层，隐藏层包括卷积层、激励层、池化层和全连接层。其中卷积层是整个网络模型的关键，作用是特征提取和特征映射。激励层的作用是将线性运算非线性化，池化层的作用是降低数据的维度、减小数据量。全连接层的作用是将最终输出结果归一化，保证维度一致，以便后续使用。CNN的结构可用图2-6表示。

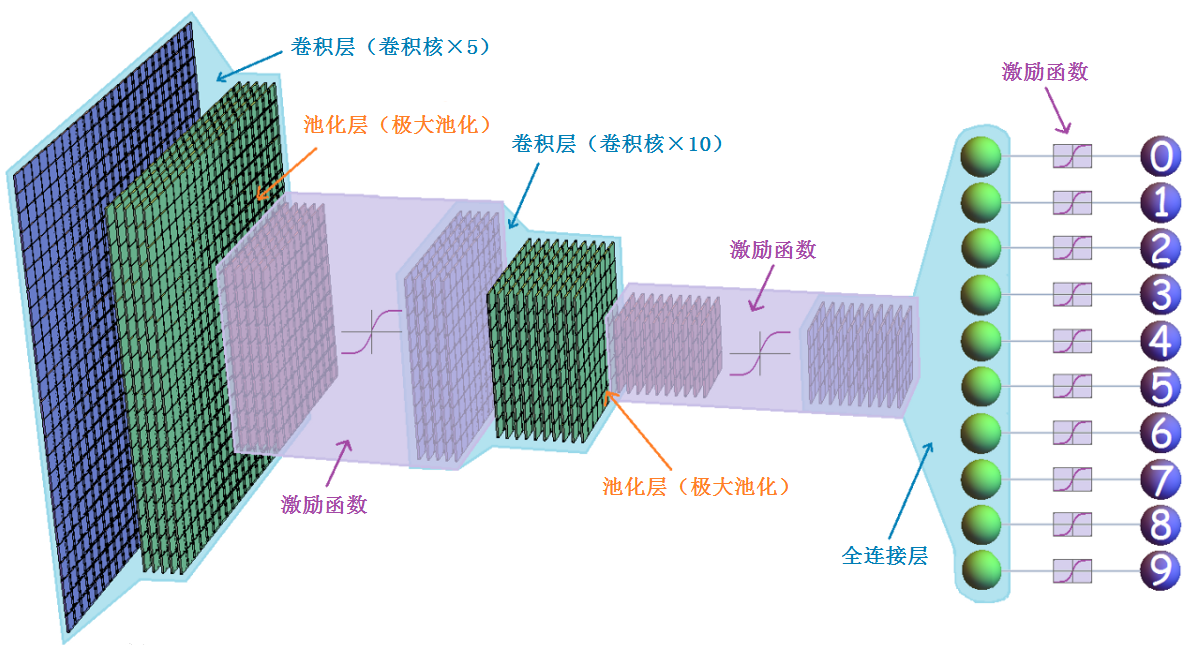


图2‑6 CNN结构图

### 基于内容的图片检索技术简介

图片云端管理系统的图片搜索分为两种方式，一种是以图片名字或标签作为关键字搜索图片，第二种是基于图片内容的搜索方式，简称以图搜图。

以图搜图的过程如图所示，首先提取图片库里每张图片的特征，提取的特征为CNN训练输出的向量，然后把每张图片的特征矢量保存在数据库中。当要搜索某一张图片，先用CNN的前向传播提取矢量特征，然后将该特征向量和数据库里的每一个特征向量进行相似度计算，对所有的相似度结果进行排序，取前面最佳的几个，其对应的图片就是搜索的结果[16]。图片搜索过程可用图2-7表示。

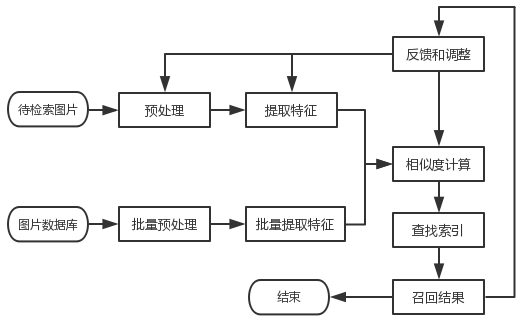


图2‑7 图片检索流程

### 网络爬虫技术简介

网络爬虫技术[17]在图片云端管理系统的作用是在网络上抓取相似图片，并可以添加到用户的图库达到动态更新图库的目的。网络爬虫又称为网络机器人，它的本质是利用脚本主动向目标Web服务器请求资源，并对资源再次加工利用，根据目标网页资源的加载情况，爬虫又分为静态网页爬虫和动态网页爬虫。网络爬虫的结构可用图2-8表示。

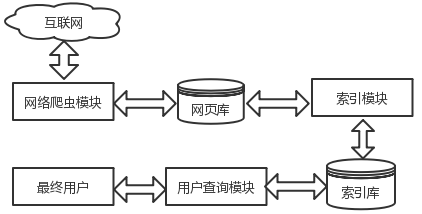


图2‑8 网络爬虫结构

本项目使用Requests和Beautiful Soup两个Python工具包建立网络爬虫。爬虫抓取的目标图片主要来源是百度图片，即利用百度图片搜索图片的URL请求含有目标图片的网页，然后从这个网页中提取每张图片的资源路径，通过图片的资源路径便能获得目标图片。

### 开发工具和运行环境

（1）为了提高开发效率，本项目的开发过程使用了大量的开源框架和免费工具，工具如表2‑1所示。

表2‑1 开发工具表

|  |  |
| --- | --- |
| 开发工具名称 | 软件用途 |
| JQuery | 前端JavaScript编写 |
| Bootstrap3 | 前端界面设计 |
| PyCharm | Python集成开发环境 |
| Django | 搭建Web应用 |
| MySQL | 存储数据 |
| TensorFlow | 构建图片分类模型 |
| Chrome | 调试本系统的前端功能 |
| Git | 版本管理 |
| Apache | 部署最终的系统 |

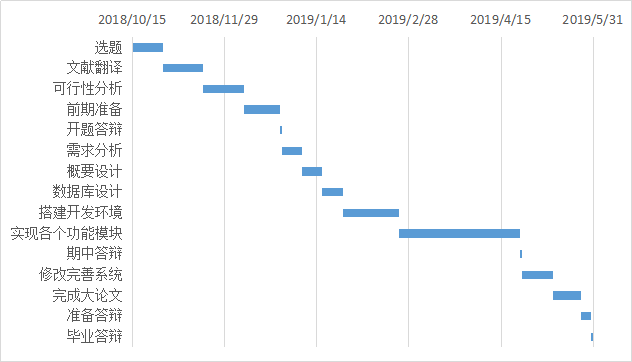
（2）图片云端管理系统分为前端和后端，前端的运行环境在用户终端，而后端需要部署到云服务器上，本系统运行环境如表2‑2所示。

表2‑2 运行环境表

|  |  |
| --- | --- |
| 运行环境名称 | 属性 |
| 用户终端 | 笔记本电脑、台式机 |
| 用户的操作系统 | Windows7或更高、Ubuntu、iOS |
| 用户的前端 | Chrome、Firefox |
| 部署平台 | 腾讯云服务器 |
| 后端的操作系统 | Ubuntu16.0.4 |
| CPU | 1核 |
| 内存 | 2G |
| 网络带宽 | 1 Mbps宽带 |
| 磁盘 | 50G |

### 项目实施计划

本项目在2018/10/15选题，计划前期花大约三个月完成文献翻译、可行性分析、需求分析、概要分析和数据库分析等，中期2019/1/15到2019/4/15 大约三个月完成开发环境搭建、实现系统各个功能模块和测试系统，后期2019/4/15至2019/5/29大约一个半月完成相关文档编写，最终在2019/5/31答辩结束。整个项目实施计划的时间表见图2-9任务计划甘特图。

图2‑9 任务计划甘特图

# 项目设计及实现

本章将详细说明图片云端管理系统的具体设计和实现过程，第一节给出了开发本项目的数据库设计，第二节将展开本系统的各个模块详细阐述实现的过程，包括功能模块的流程图、核心代码和运行效果。



图3‑1 系统实体关系图

## 数据库设计

本项目的图片云端管理系统采用MySQL数据库存储结构化数据，而大量的无结构化的二进制数据，即图片，则存储在文件系统中，并且在数据库中指明图片在文件系统中的路径。

本系统中的实体主要有用户、图册、图片、标签和评论，这些实体的主键都为自动增长的整型数值。关系对象包括点赞和收藏。用户的属性有昵称、密码、头像、手机号、性别、个人说明和个人空间的封面图片；图册的属性有名字、类别、描述和创建时间；图片的属性包括文件名、二进制数据和判断是否为图册的封面；标签的属性有名字；评论的属性有内容。这些实体的关系大部分是一对多，但是评论实体比较特殊，评论和本身是一对多的关系。具体的实体对象及其属性和实体之间的关系在图3-1给出。根据以上说明，本项目在数据库中建立了用户表、图册表、图片表、标签表、评论表、收藏表和点赞表，见表3-1。

表 3‑1数据库表

|  |  |  |
| --- | --- | --- |
| 序号 | 表名 | 表信息说明 |
| 1 | myapp\_user | 用户对象 |
| 2 | myapp\_album | 图册对象 |
| 3 | myapp\_photo | 图片对象 |
| 4 | myapp\_album\_tag | 标签对象 |
| 5 | myapp\_comment | 评论行为 |
| 6 | myapp\_ collection | 收藏行为 |
| 7 | myapp\_ favour | 点赞行为 |

这里依次给出每个实体对象的数据字典。

（1）用户对象的字段包括昵称、密码、头像、手机号码、性别、个人说明和个人空间的封面图片，其中昵称约束为唯一值，性别的域为男、女和保密，头像和封面的类型为图片，手机号码的作用是帮助用户找回密码。用户关系表的定义见表3-2。

表 3‑2用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 数据类型 | 长度 | 说明 |
| 1 | pk | IntField | 4 | 主键，自增长 |
| 2 | nickname | CharField | 16 | 昵称，唯一，null=False |
| 3 | pwd | CharField | 16 | 登录密码，null=False |
| 4 | phone | CharField | 11 | 手机号码，null=False |
| 5 | gender | IntField | 4 | Choices = （男，女，保密） |
| 6 | figure | ImageField | 1 | 头像 |
| 7 | cover | ImageField | 1 | 个人空间封面 |
| 8 | signature | CharField | 50 | 个人说明 |

（2）图册对象的字段包括名字、类别、描述文字、创建时间和所属用户，除了描述文字可以为空，其他字段都不允许为空。图册关系表的定义见表3-3。

表 3‑3图册表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 数据类型 | 长度 | 说明 |
| 1 | pk | IntField | 4 | 主键，自增长 |
| 2 | name | CharField | 16 | 名字，null=False |
| 3 | category | IntField | 4 | 类别，null=False |
| 4 | note | TextField | 1000 | 图册描述 |
| 5 | create\_time | DateField | 4 | 创建时间，null=False |
| 6 | user | IntField | 1 | 外键，所属用户，null=False |

（3）图片对象的字段包括文件名、二进制数据、是否为封面和所属图册，并且都不允许为空。图片关系表的定义见表3-4。

表 3‑4图片表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 数据类型 | 长度 | 说明 |
| 1 | pk | IntField | 4 | 主键，自增长 |
| 2 | name | CharField | 16 | 文件名，null=False |
| 3 | cover | BooleanField | 1 | 是否为图册封面，null=False |
| 4 | data | ImageField | 1 | 二进制数据，null=False |
| 5 | album | IntField | 4 | 外键，所属图册，null=False |
| 6 | label | IntField | 4 | 图片类别 |

（4）标签对象的字段包括名字和所属图册，其关系表的定义见表 3‑5。

表 3‑5标签表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 数据类型 | 长度 | 说明 |
| 1 | pk | IntField | 4 | 主键，自增长 |
| 2 | name | CharField | 16 | 名字，null=False |
| 3 | album | IntField | 4 | 外键，所属图册，null=False |

（5）评论对象的字段包括内容、父级评论和所属图册。评论关系表的定义如表 3‑6所示。

表 3‑6评论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 数据类型 | 长度 | 说明 |
| 1 | pk | IntField | 4 | 主键，自增长 |
| 2 | content | TextField | 1000 | 内容，null=False |
| 3 | origin | IntField | 4 | 外键，父级评论，null=False |
| 4 | album | IntField | 4 | 外键，所属图册，null=False |
| 5 | time | DatetimeField | 4 | 评论时间 |
| 6 | user | IntField | 4 | 发表评论的用户 |
| 7 | view | BooleanField | 1 | 用户是否已经查看 |

（6）用户对象和图册对象的点赞关系，点赞关系表的定义见表 3‑7。

表 3‑7评论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 数据类型 | 长度 | 说明 |
| 1 | pk | IntField | 4 | 主键，自增长 |
| 2 | user | IntrField | 4 | 外键，用户对象，null=False |
| 3 | album | IntField | 4 | 外键，图册对象，null=False |

（7）用户对象和图册对象的收藏关系，收藏关系表的定义见表 3‑8。

表 3‑8评论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 数据类型 | 长度 | 说明 |
| 1 | pk | IntField | 4 | 主键，自增长 |
| 2 | user | IntrField | 4 | 外键，用户对象，null=False |
| 3 | album | IntField | 4 | 外键，图册对象，null=False |

## 功能模块的设计与实现

本节首先给出图片云端管理系统的整体框图，然后根据该框图依次阐述每个功能模块的设计方法和实现的细节，并给出每个功能模块的最终运行效果。

### 系统整体框图

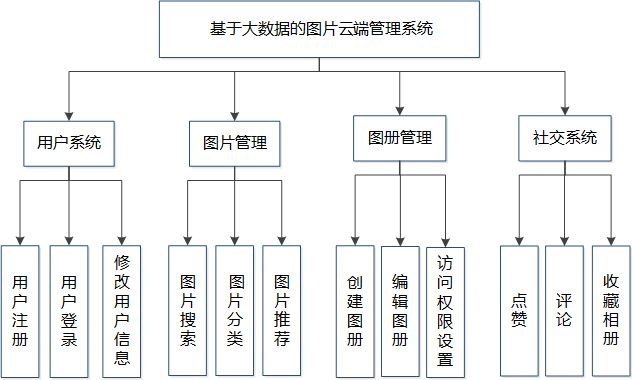


图3‑2 系统整体框图

本项目的图片云端管理系统采用B/S模式，总体可以分为四大模块，分别是用户系统模块、图片管理模块、图册管理模块和社交模块，用户系统模块又可以分为账户注册、登录和修改用户信息三个功能模块，图片管理模块可分为图片分类、图片搜索和图片推荐三个功能，图册管理分为创建图册、编辑图册和设置访问权限三个功能，社交系统分为点赞、评论和收藏图册三个功能，如图3-2所示。

### 用户系统模块的设计与实现

用户系统的主要功能是管理用户，是整个系统的基本功能，因为系统的其它功能模块正常有序运行都建立在用户的账号上。本小节将详细介绍用户系统模块的设计与实现，包括账户注册、登录和修改账户信息。

（1）用户注册功能

当一个游客访问本系统时，首先进入的是主页，此时，由于没有注册登录，只能浏览主页的信息。主页的信息主要是系统推送的较热门的图册，选择热门的图册目的在于吸引住游客，使其最有可能地成为本系统的用户。当游客想进一步使用本系统的服务时，可以点击主页的右上角的注册按钮进入注册页面。注册账号时，需要游客输入昵称、密码和手机号码。系统会实时验证昵称的唯一性、密码的合理性和手机号码的可用性。只要哪一步验证不成功，系统都会提示游客重新输入，直至全部验证成功。数据验证成功后，系统便根据输入的信息在用户表上增加一个新用户，并且将注册页面重定位到登录页面。注册功能实现过程见图3-3注册流程图。

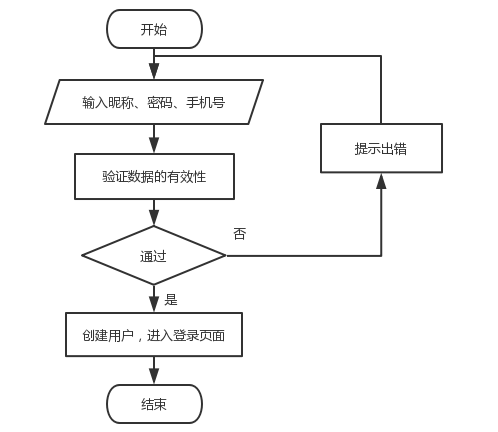


图3‑3 用户注册流程图

实现用户注册的验证代码如下，在后端，先获得昵称和手机，检查数据库中是否存在已有用户的昵称、手机号和新用户注册的昵称、手机号一样，若一样则注册失败，反之则注册成功，实现的代码如下。

if request.is\_ajax(): #注册验证

field = request.GET.get('field')

ret = 't' #ture

exist = False

if field == 'nickname':

val = request.GET.get('nickname')

exist = User.objects.filter(nickname=val)

elif field == 'phone':

val = request.GET.get('phone')

exist = User.objects.filter(phone=val)

if exist:

ret = 'f' #false

msg = {'msg':ret}

return HttpResponse(json.dumps(msg))

（2）用户登录功能

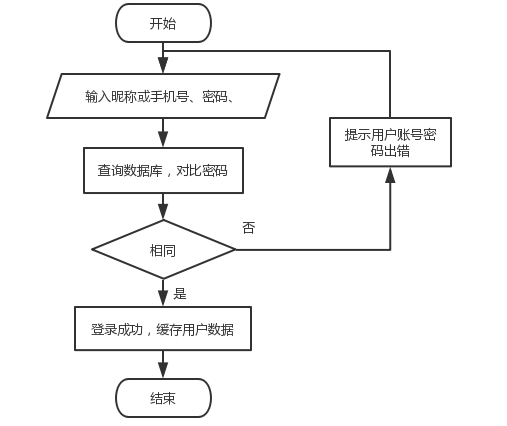


图3‑4 用户登录流程图

本系统的用户登录功能主要作用是验证用户的合法性，已注册用户登入时，需要在页面输入昵称或手机号，还有正确的密码。用户之所以可以输入昵称或手机号，是因为昵称和手机号在数据库中的定义具有唯一性，可以唯一标识一个用户。系统收到登录请求时，先根据前端输入的昵称或手机号查询用户表，然后将输入的密码和数据库保存的密码进行对比，两者相同则登录成功，反之登录失败，并把结果返回给用户。登录过程见图3-4用户登录流程图。

用户登录的后端验证代码如下，后端收到登录数据首先判断账号字段的类型是昵称还是手机号，然后根据类型进行查询用户，对比输入的密码是否和数据库保存的密码相同，若相同则登录成功，反之则失败。

if request.is\_ajax(): #注册验证

field = request.GET.get('field')

ret = 't' #ture

if request.method == 'POST':

val = request.POST.get('user')

try:

if len(val) == 11 and val.isdigit():

m\_user = User.objects.get(phone=val)

else:

m\_user = User.objects.get(nickname=val)

pwd = request.POST.get('pwd')

if m\_user.pwd == pwd: #登录成功，记录user信息

request.session['logined'] = 1

request.session['user\_id'] = m\_user.pk

return response

else: raise

except Exception as err:

print("login ---- ", err)

return render(request, 'login.html',

{'msg':'用户名或密码错误', 'username':val, 'pwd':pwd})

（3）修改用户信息功能

本系统的修改用户信息功能的作用是让用户编辑或者修改用户信息，用户信息包括昵称、密码、头像、性别和个人说明，其中比较关键的是从本地选择图片并保存到后端数据库中。上传头像时，利用浏览器访问文件系统的接口选择图片，并把图片显示在页面，用户确认后即可上传到数据库中。以下是上传头像的前端JavaScript代码。上传头像过程见图3-5修改用户头像流程图。

$("#imgInput").change(function (event) { //选择头像

var murl = null;

var imgfile = this.files[0]

if (window.createObjectURL != undefined) {

murl = window.createObjectURL(imgfile);

} else if (window.URL != undefined) {

murl = window.URL.createObjectURL(imgfile);

} else if (window.webkitURL != undefined) {

murl = window.webkitURL.createObjectURL(imgfile);

}

document.getElementById("imgPreview").src = murl;

});

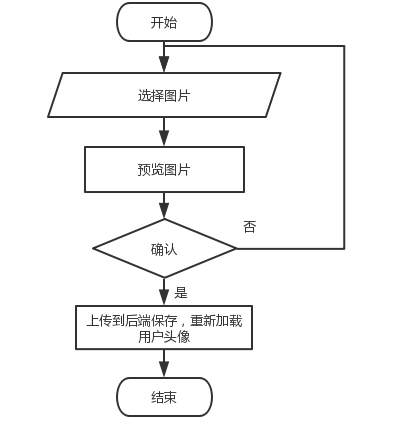


图3‑5 修改用户头像流程图

### 图册管理模块的设计与实现

本系统的图册管理模块包括创建图册、编辑图册和设置访问权限三个功能。本小节将依次详细介绍实现过程。

（1）创建图册功能

用户在个人空间页面可以创建图册并上传多张图片，输入的图册信息有图册标题、类别、标签和相关描述。其中创建图册的关键信息是图片，图片由用户在本地文件选择，并且预览，预览包括删除不需要的图片和将某张图片设置为图册的封面，最终点击确认上传到后端，核心代码如下，对应流程图见图3-6。

var imgArr = new Array();

var cover\_label = -1;

//上传图片

$("#imgs\_input").change(function () {

var count = $("#count").text().trim();

count = new Number(count);

var imgfiles = this.files;

var n = imgfiles.length;

for (var i=0; i<n; i++){

imgArr.push(imgfiles[i]); //将图片放入数组中

var imgUrl = getFileURL(imgfiles[i]);//调用函数调取图片地址

//显示图片

var $newElem = $("<div class='img-card'></div>");

$newElem.css('background-image',"url("+ imgUrl +")");

var $newChild = $("<div class='cover-del'></div>");

var $setcover = $("<span>设为封面</span>");

var $del = $("<span>删除图片</span>");

$newChild.append($setcover);

$newChild.append($("<span> | </span>"));

$newChild.append($del);

$newElem.append($newChild);

$newElem.attr("index", (i+count).toString());

$("#add-img").before($newElem);

//设为封面事件

$setcover.on('click', function () {

var $old = $("#upload>div>img")[0];

var $p = $(this).parent().parent();

$p.prepend($old);

cover\_label = new Number($p.attr('index'));

//alert(cover\_label);

});

//删除事件

$del.on('click', function () {

var index = new Number($(this).parent().parent().attr('index'));

//alert(index);

$(this).parent().parent().remove();

var count = $("#count").text().trim();

count = new Number(count);

count -= 1;

$("#count").text(count.toString());

//删除对应file

imgArr.splice(index,1);

//重新设置index

var imgcards = $(".img-card");

for (var j=0; j < imgcards.length; j++){

imgcards[j].setAttribute('index', j.toString());

}

//如果删除的是封面，则设置第一张为封面

if (index.toString() == cover\_label.toString()){

$("#upload>div:first").prepend("<img src=\"/static/img/cover-image-label.png\">");

cover\_label = 0;

if (count <= 0) cover\_label = -1;

}

});

}

});

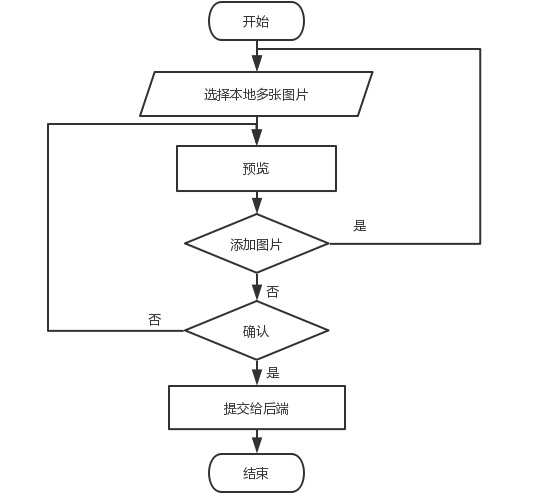


图3‑6 创建图册流程图

（2）编辑修改图册功能

用户在个人空间，可以对已创建的图册进行编辑修改。编辑修改图册的界面和创建图册的界面是一样的，不同的是数据库操作，创建图册需要用户输入信息并在数据库添加一条图册数据，而编辑图册只是在原来的数据上进行编辑修改，然后在数据库更新这个图册的数据。所以，编辑修改图册的流程和创建图册的流程是一样的。见图3-6 创建图册流程图，界面效果图如图3-7。

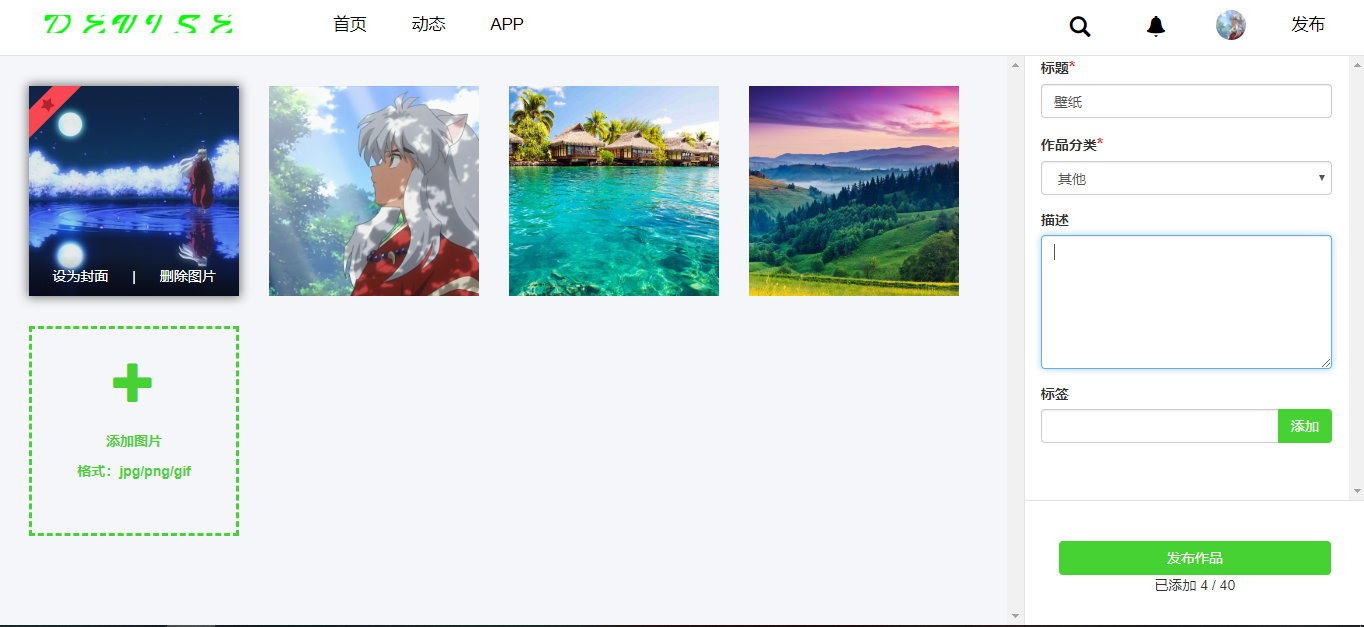


图3‑7 编辑图册信息效果图

（3）设置访问权限功能

设置图册访问权限功能的作用是保护用户的隐私，图册访问权限的等级分为三级，分别为所有人可以访问、关注的人可访问和仅仅自己可见。因为权限只有三个等级，设置方法是在用户前端使用select下拉列表显示三个权限等级，供用户选择。界面效果如图3-8所示。



图3‑8 设置访问权限效果图

### 图片管理模块的设计与实现

本系统的图片管理模块包括图片分类、图片搜索和图片推荐功能，为了实现这些，本项目功能使用了基于CNN的图片分类方法和矢量特征提取方法。本小节将依次详细介绍实现过程。

（1）图片分类功能

本项目使用CNN实现图片分类功能，具体过程是运用TensorFlow构建训练模型[18]，训练的类别有7类，分别为人像、风景、美食、文档、建筑、动物和动漫，从零开始编号表示每类标签。先用网络爬虫在网络上下载这七类图片，然后人工过滤那些无效图片，最后进入图片预处理和训练过程，训练过程见图3-9。

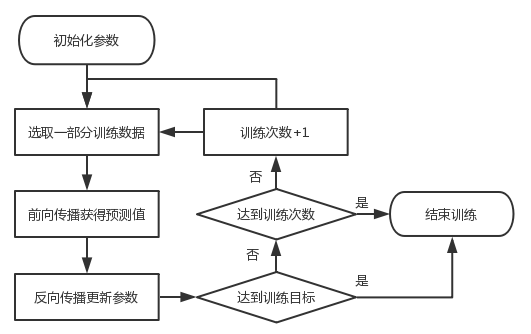


图3‑9构建分类模型流程图

得到分类模型后，将模型保存在后端，当用户上传图片到后端，后端系统利用前向传播获得预测值，即可得到相应的类别，最后在数据库写入图片的类别字段。在前端，用户在个人空间的分类栏目可以看到系统自动完成的分类图片。使用分类模型预测类别的核心代码如下。

//前向传播获得预测值

with tf.Session() as sess:

image\_raw\_data = tf.gfile.GFile(img\_file, 'rb').read()

image = tf.image.decode\_jpeg(image\_raw\_data)

if image.dtype != tf.float32:

image = tf.image.convert\_image\_dtype(image, dtype=tf.float32)

image = tf.image.resize\_images(image, [224, 224])

input\_data = sess.run(image)

x\_input = [input\_data]

# 找到已有的模型，进行读取

saver = tf.train.import\_meta\_graph('./m\_model/m.ckpt.meta')

saver.restore(sess, tf.train.latest\_checkpoint('./m\_model'))

graph = tf.get\_default\_graph()

x = graph.get\_tensor\_by\_name("x-input:0")

#batch = graph.get\_tensor\_by\_name("batch-input:0")

logits = graph.get\_tensor\_by\_name("logits\_eval:0")

logits = tf.nn.softmax(logits)

y = tf.argmax(logits, 1)

softmax, predict = sess.run([logits, y], feed\_dict={x: x\_input})

（2）基于图片内容的图片搜索功能

图片云端管理系统的图片搜索分为两种方式，第一种是通过用户输入的标签和类别进行分类图片，这种方式是按照用户自己的意愿对图片进行分类，第二种是基于图片内容进行分类图片，这种方式是系统后端利用图片分类模型对用户上传的图片进行预测分类，两种方式互不影响，可同时存在，只是在前端展示图片的结果不一样。

基于内容的图片搜索的过程如图所示，首先提取图片库里每张图片的特征，提取的特征为CNN训练输出的向量，然后把每张图片的特征矢量保存在数据库中。当要搜索某一张图片，先用CNN的前向传播提取矢量特征，然后将该特征向量和数据库里的每一个特征向量进行相似度计算，相似度计算的方法为欧式距离计算，对所有的相似度结果进行排序，取前面最佳的几个，其对应的图片就是搜索的结果。实现过程见图3-10图片搜索流程图。

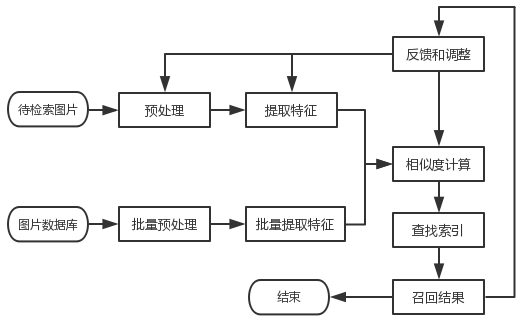


图3‑10图片检索流程图

（3）图片推荐功能

本系统的图片推荐功能可以提高用户的使用系统的体验。实现图片推荐功能没有使用专业的推荐算法，而是仅仅凭借用户的历史社交行为来决定推荐的图片内容。通过系统的社交功能，用户可能会浏览其他用户的图册，如果看到喜欢的图册就会点赞，或者收藏，或者评论。系统会记录这些用户的社交行为，因为用户点赞、收藏和评论过的图册带有标签和类别字段，系统根据这些标签和类别便可以推荐相同类别的图片给用户。图片推荐功能的实现过程见图3-11图片推荐流程图。

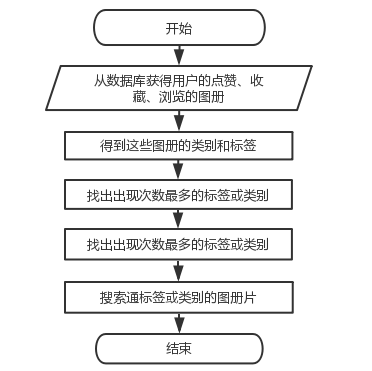


图3‑11 图片推荐流程图

### 社交系统模块的设计与实现

本系统的社交模块的作用是满足用户的社交需求，提高用户体验。用户在本系统的社交行为包括点赞、收藏图册和对图册进行评论。本小节将依次详细介绍这三个功能的实现过程。

（1）用户点赞功能

用户浏览其他用户的图册时，可以点击点赞按钮为图册点赞，以表示喜欢或对该用户的肯定。用户给图册点赞后，前端会往后端传递一条点赞的信息并存储在数据库，信息包括发出点赞的用户和被点赞的图册，然后该图册所属用户在登入之后便能收到点赞的消息。点赞效果如图3-12所示。

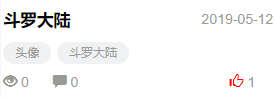


图3‑12 用户点赞效果图

（2）用户收藏图册功能

用户浏览其他用户的图册时，可以点击收藏按钮收藏该图册，并在数据中保存这条收藏信息，包括主动收藏用户和被收藏图册。收藏成功后，用户在个人空间能看到自己收藏的图册，但是用户只能浏览图册而不能编辑删除图册。图册收藏的界面如图3-13所示。

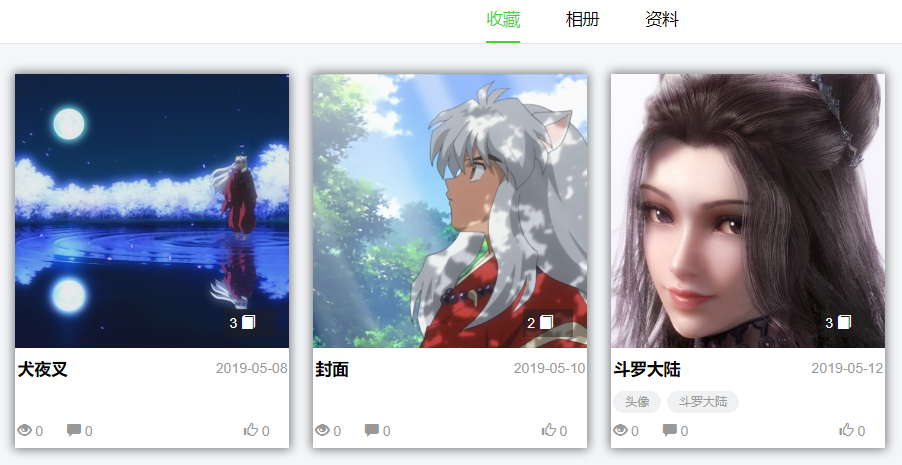


图3‑13 收藏图册效果图

（3）用户评论图册功能

用户在浏览其他用户的图册时，可以发表评论或回复其他用户的评论，保存在数据库的评论包括的信息有评论内容、父级评论和所属图册。评论的界面效果如图3-14所示。



图3‑14 编辑图册信息效果图

# 特殊问题及解决方案

在本项目开发的过程中，遇到了一些比较特殊的问题，产生这些问题的原因主要有对开发工具不熟悉和对关键技术没有经验。本章将描述本项目开发过程中出现的每一个问题，以及解决问题的方案和结果。

## Django时间设置问题

### 问题描述

在数据库设计中，给图册表添加创建时间，创建新的图册对象却抛出异常，出错信息如下。

File"D:\Another\Anaconda3\lib\site-packages\django\utils\dateparse.py",line107,inparse\_datetime

match = datetime\_re.match(value)

TypeError: expected string or bytes-like object

### 解决方案

经过在网上查找，得知出现这种问题的原因是没有给时间字段设置默认的时间，继而知道解决办法。

首先在工程里的migrations文件夹中，删除最近那次的数据库迁移文件，如同0018\_auto\_20190509\_0032.py，再给models.DateTimeField添加default值，然后重新执行数据库迁移命令，代码如下。

from django.utils import timezone

class Album(models.Model):

name = models.CharField(max\_length=16)

category = models.IntegerField(choices=album\_category\_choice, default=0) #类别

note = models.TextField(null=True, blank=True) #描述

create\_time = models.DateTimeField(default=timezone.now) #创建时间

修改完模型的定义后，重新执行数据库迁移命令，如下。

python manage.py makemigrations

python manage.py migrate

### 结果

再次创建图册时，不再抛出异常，并且在数据库中查询到新增加的图册对象有创建的时间字段。

## 前端无法访问上传的图片的问题

### 问题描述

用户在前端上传了自己的头像，并且在media文件夹下确实能看到上传成功的头像，但是在前端却不能访问到这张头像图片。详细的错误信息如图4-1所示。

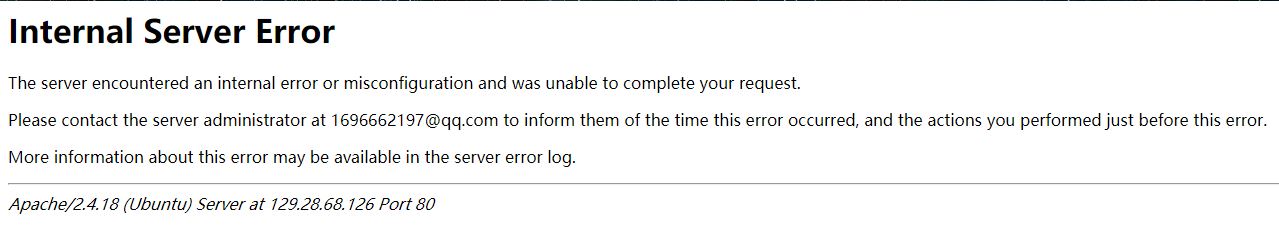


图4‑1 无法访问上传的图片的问题

### 解决方案

猜想可能是后端的配置出了问题，所以参考Django官网的文档[14]，最后找到了原因。原来真的是配置文件少了一些相关内容，需要在工程里的URL文件里加上media的路径。添加的内容如下。

from django.views.static import serve

from . import settings

urlpatterns = [

url(r'^$', views.index, name='index'),

url(r'^media/(?P<path>.\*)', serve, {"document\_root":settings.MEDIA\_ROOT}),

]

因为在后端使用ImageField图片类型定义头像字段figure，而且figure.url表示头像的路径，所以为了能加载出头像，需要将figure.url传到前端，在前端，将img标签的src属性值设置为figure.url，便能成功加载出头像。

### 结果

使用上面的解决办法，不仅实现了在前端加载头像，也实现了加载其它任何上传的图片。

## CNN训练模型内存不足问题

### 问题描述

为了实现图片分类，使用了VGGNet网络训练分类模型[19]。在预处理原始图片时，程序总是在运行一半的时候突然中止，并抛出OUT OF MEMORY异常，错误信息如下。

E tensorflow/stream\_executor/cuda/cuda\_driver.cc:924] failed to alloc 17179869184 bytes on host: CUDA\_ERROR\_OUT\_OF\_MEMORY

### 解决方案

一开始在网上寻找解决办法，网上说法是，出错原因在于没有设置任何GPU配置参数情况下报错的，TensorFlow[20]默认使用所有GPU资源，但是GPU内存占用快满时，系统会拒绝分配，所以TensorFlow抛出CUDA\_ERROR\_OUT\_OF\_MEMORY，要添加GPU设置，如下。

config.gpu\_options.per\_process\_gpu\_memory\_fraction=0.8

然而照着这种办法还是没解决。实际上，我这里的原因是没有将原始图片转化成TFrecord格式，直接把图片数据全部读取到内存进行处理，以至于图片数据总量大于gpu已有的内存大小，所以程序在中途异常退出。综上所述，解决办法是先将所有图片转化成TFrecord格式，然后进行下一步处理，核心代码如下。

# 将数据转化为tf.train.Example格式。

def \_make\_example(label, image):

image = image.tobytes()

example = tf.train.Example(features=tf.train.Features(feature={

'label': \_int64\_feature(label),

'image\_raw': \_bytes\_feature(image)

}))

return example

### 结果

预处理程序正常运行直到结束，中途没有抛出OUT\_OF\_MEMORY异常。

## 部署系统路径错误问题

### 问题描述

本系统部署的环境为腾讯云服务器、ubuntu16.0.4、Apache2、MySQL5.7、Django2.2、python3.5（ubuntu自带）。照着网上教程部署了Django网站，然而在自己电脑输入云服务器的公网ip得到的总是Apache的error页面。在控制台输入cat /var/log/apache2/error.log，可以查看Apache的日志。日志显示在wsgi.py 文件里出错，并且出现大量的以下信息。

ImportError: No module named 'django'

### 解决方案

对于以上问题，一直以为是/etc/apache2/sites-available/mysite.conf文件配置出错，于是修改这个文件的内容，但是依然没有解决问题。为了找到问题的原因，依次执行以下命令，但是结果一切正常，没有异常，执行的命令及结果如图4-2所示。

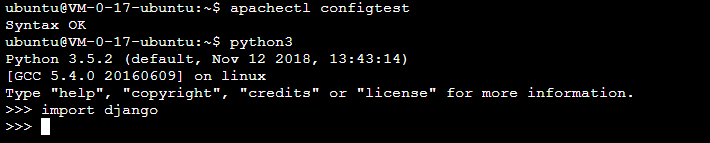


图4‑2 检查问题出处

经过大量的查找，最终在StackOverflow上找到答案，原来是Django等Python包安装路径不正确，导致权限不够。一开始安装用pip3 install django，用pip3 show django命令，显示Django在/home/ubuntu/.local/lib/python3.5/site-packages，所以是安装路径出错。正确安装是用sudo pip3 install django，其它Python包也是要加上sudo，这样安装路径就是/usr/local/lib/python3.5/dist-packages。

### 结果

重新正确安装Django后，系统可以正常运行，在前端能正常访问主页。

# 结果测试及性能分析

本章将阐述使用自动化测试和手动测试方法对实现好的本系统进行测试，接着，再对系统的性能进行分析。

## 测试概要

系统测试的目的是找出bug，并加以修改，最终使系统更加完善。基于B/S模式的系统测试，除了需要检查系统的各个功能是否按照要求正常运行，还要检查系统在不同用户的浏览器上是否正常显示。

### 测试环境

由于主要对前端进行测试，以下给出前端测试环境。

1. 操作系统：Window7或更高版本、Ubuntu、iOS等。
2. 浏览器：Chrome、IE、Firefox等。
3. CPU：Intel Core i5-6200U 2.3GHz。
4. 网络带宽：1 Mbps。
5. 内存：4G。
6. 磁盘：50G。

### 测试范围

本节从功能、性能、和前端兼容性三个方面对最终实现好的图片云端管理系统进行较全面的测试。

（1）功能测试

这里使用手动测试方法对本系统的每一个功能依次进行测试，测试结果见表5-1功能测试表。

表 5‑1功能测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 功能名称 | 操作 | 结果 |
| 1 | 用户注册 | 输入用户昵称、账号密码和手机号 | 成功创建用户 |
| 2 | 用户登录 | 输入用户昵称和账号密码 | 正确提示成功与否 |
| 3 | 修改用户信息 | 编辑用户的信息并更新到数据库 | 成功更新用户信息 |
| 4 | 创建图册 | 输入图册标题、标签和描述，上传图片 | 成功添加图册 |
| 5 | 编辑图册 | 删除图册、修改信息，增加、删除图片 | 成功修改并更新 |
| 6 | 设置访问权限 | 选择访问权限并保存设置 | 正确访问图册 |
| 7 | 图片分类 | 根据标签分类，自动分类 | 成功分类图片 |
| 8 | 图片搜索 | 输入关键字搜索，基于内容搜索 | 成功找到目标图片 |
| 9 | 图片推荐 | 根据用户社交行为推荐图片或图册 | 成功推荐 |
| 10 | 点赞 | 给其他用户的图册点赞 | 成功点赞并通知对方 |
| 11 | 收藏图册 | 收藏喜欢的图册 | 成功收藏 |
| 12 | 评论 | 对图册进行评论 | 成功评论 |
| 13 | 退出登录 | 点击退出系统链接 | 成功退出系统 |

（2）性能测试

性能测试的内容主要包括时间性能和空间性能，为了获得更准确的性能数据，这里使用自动化测试，性能测试的工具为Pingdom网站性能测试软件。把本系统的域名输入到该软件的入口，点击开始测试，等待几分钟，该软件为本系统生成了请求主页文件的性能测试的数据结果，数据见表5-1请求时间表和表5-2请求文件表。通过数据可知，请求一个页面所花费的总时间是200.5ms，请求的文件总数为2.3MB。

表5‑2 请求时间表

|  |  |
| --- | --- |
| 操作 | 花费时间 |
| DNS | 0.0ms |
| SSL | 0.0ms |
| Connect | 97.5ms |
| Send | 0.1ms |
| Wait | 101.9ms |
| Receive | 1.1ms |
| Blocked | 0.0ms |
| Total | 200.5ms |

表5‑3 请求文件表

|  |  |  |
| --- | --- | --- |
| 内容类型 | 数据大小 | 百分比 |
| Image | 2.2MB | 96.42% |
| Script | 40.1KB | 1.76% |
| CSS | 19.7KB | 0.87% |
| Font | 18.0KB | 0.79% |
| HTML | 3.5KB | 0.15% |
| Total | 2.3MB | 100.00% |

（3）前端兼容性测试

因为用户使用的浏览器有多种选择，所以需要对各个浏览器进行前端兼容性测试，这里使用手动测试方式，测试平台主要是用户最常用的浏览器。结果见表5-4前端兼容性测试表。

表 5‑4前端兼容测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 浏览器名称 | 版本号 | 结果 |
| 1 | Google | 73.0.3683.75 | 正确显示UI组件 |
| 2 | Firefox | 65.0.2 | 正确显示UI组件 |
| 3 | Internet Explorer | 11.765.17134.0 | 正确显示UI组件 |
| 4 | Microsoft Edge | 42.17134.1.0 | 正确显示UI组件 |
| 5 | Opera | 54.0.2929.0 | 正确显示UI组件 |

## 性能分析

通过观察上节性能测试的数据结果可知，由于本系统的数据主体是图片类型，请求文件当中，图片数据占了几乎整个文件大小，所以图片数据更要占用资源，加载也比较耗时。

结 束 语

本课题从用户对有效管理图片的需求出发，运用云端技术和人工智能打造了一款图片云端管理系统。该系统旨在为用户提供一个安全的云端来存储大量图片，并提供一些实用的功能帮助用户科学有效的管理零散而又杂乱的图片，使用户方便快捷地使用图片。

本系统的功能特色有自动图片分类、以图搜图和图片推荐，相对应的，技术特色有卷积神经网络算法、基于内容的图片搜索和图片网络爬虫。为了实现图片分类功能，本项目采用TensorFlow深度学习框架，利用卷积神经网络训练分类模型，并将模型投入生产当中。经过大量数据的训练，分类模型在测试中可以达到90%的准确率，在给用户的图片自动分类时达到较好的效果。在训练模型的过程中，遇到的难题主要是硬件能力不足导致运算要耗费大量时间，以及训练数据含有较多杂质，这是因为样本数据是通过网络爬虫从百度图片收集的。运算能力不足的问题通过分布式技术可以得到一定缓解，而训练集含有杂质的解决办法是通过人工筛选、剔除代表性较差的图片。

对我来说，项目中最具挑战力的是运用深度学习实现系统的图片自动分类功能，在这过程中遇到很多棘手的问题，但也让我对图像处理和深度学习有了更高的理解。总的来说，经过本次项目的研究，我的分析问题和解决问题的能力得到了很大的提高。

参考文献

1. 中国互联网络信息中心. 中国互联网络发展状况统计报告[R]. 中国互联网络信息中心, 2019.
2. Lingxi Xie, Richang Hong, Bo Zhang, and Qi Tian. Image Classification and Retrieval are ONE[A]. Proceedings of the 5th International Conference on Multimedia Retrieval (ICMR '15) [C], ACM, New York, USA, 2015:3-10.
3. Min-Hsuan Tsai, Shen-Fu Tsai, and Thomas S. Huang. Hierarchical image feature extraction and classification[A]. Proceedings of the 18th ACM international conference on Multimedia (MM '10)[C]. ACM, New York, USA, 2010:1007-1010.
4. Zhan Qin, Jingbo Yan, Kui Ren, Chang Wen Chen, and Cong Wang. Towards Efficient Privacy-preserving Image Feature Extraction in Cloud Computing [A]. Proceedings of the 22nd ACM International Conference on Multimedia (MM '14) [C], ACM, New York, USA, 2014:497-506.
5. Almarzooqi Ahmed and Tomoko Ozeki. Food Image Recognition by Using Bag-of-SURF Features and HOG Features [A]. Proceedings of the 3rd International Conference on Human-Agent Interaction (HAI '15) [C], ACM, New York, USA, 2015:179-180.
6. Wang Lei, Huawei Chen, and Yixuan Wu. Compressing Deep Convolutional Networks Using K-means Based on Weights Distribution [A]. Proceedings of the 2nd International Conference on Intelligent Information Processing (IIP'17) [C]. ACM, New York, USA, Article 10, 2017: 147-152.
7. 姜少波, 甘彤, 商国军. 基于深度学习的以图搜图架构及在公安图侦中的应用[J]. 电子技术与软件工程, 2018, (09):62-63.
8. 连锲. 高校图片管理系统设计[J]. 中国管理信息化, 2018, 21(06):146-147.
9. 朱沛贤. 基于卷积神经网络的单目图像深度估计研究[D]. 南京邮电大学[硕士论文], 2018.
10. 吴淑慧. 基于问题框架方法的软件非功能性需求分析的研究[D]. 安徽大学[硕士论文], 2018.
11. 江媛媛.“以图搜图”技术在照片档案管理中的应用研究[J]. 档案与建设, 2018, (06):38-41.
12. 李凯曼, 陈亮. 云计算中Web服务的服务质量评估模型研究[J]. 机电工程技术, 2019, 48(03):64-67.
13. 姜亚东. 卷积神经网络的研究与应用[D]. 电子科技大学[硕士论文], 2018.
14. Django Software Foundation. Django Documentation [EB]. https://docs.django project.com/zh-hans/2.1/, 2019-02-04.
15. 杨杰, 陈晓云. 图像分类方法比较研究[J]. 微计算机应用, 2007, (06):627-632.
16. 汪达舟. 面向跨模态商品搜索的深度卷积与特征融合算法研究[D]. 浙江大学[硕士论文], 2018.
17. 王子豪. 基于网络爬虫的信息采集技术研究[D]. 西北师范大学[硕士论文], 2018.
18. 郑泽宇, 梁博文, 顾思宇. TensorFlow:实战Google深度学习框架[M]. 北京:电子工业出版社, 1988:15-18.
19. Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science, 2014:1409-1556.
20. Tensorflow. Tensorflow Documentation [EB]. https://tensorflow.google.cn/ versions /api\_docs/python/tf, 2019-3-14.

致　　谢

开发完成本项目，并撰写好本论文，这些成果都离不开指导老师的悉心教导和同学们的热情帮助。

在此，我向我的指导老师，鄢田云副教授，表达由衷的感谢，感谢您这半年来不厌其烦地帮助我解惑，解决项目中的难题，感谢您不辞辛苦地给我检查论文并提出存在的问题和宝贵的建议。

我还要感谢我的学习委员唐皓月，在我撰写论文的过程中给予了我很多有用的帮助，帮我解决了相关论文排版问题。

最后向在百忙之中评审本文的各位专家、老师表示衷心的感谢！

**作者简介：**

|  |  |  |  |
| --- | --- | --- | --- |
| 姓 名： | 李莘 | 性别： | 男 |
| 出生年月： | 1996年10月10日 | 民族： | 汉 |
| E-mail： | 1696662197@qq.com |  |  |

声　　明

本论文的工作是2018年10月至2019年6月在成都信息工程大学计算机学院完成的。文中除了特别加以标注地方外，不包含他人已经发表或撰写过的研究成果，也不包含为获得成都信息工程大学或其他教学机构的学位或证书而使用过的材料。

关于学位论文使用权和研究成果知识产权的说明：

本人完全了解成都信息工程大学有关保管使用学位论文的规定，其中包括：

（1）学校有权保管并向有关部门递交学位论文的原件与复印件。

（2）学校可以采用影印、缩印或其他复制方式保存学位论文。

（3）学校可以学术交流为目的复制、赠送和交换学位论文。

（4）学校可允许学位论文被查阅或借阅。

（5）学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

除非另有科研合同和其他法律文书的制约，本论文的科研成果属于成都信息工程大学。

特此声明！

作者签名：

2019年05月25日