目录

[一、 实习内容分析 3](#_Toc52104483)

[1. 项目介绍 3](#_Toc52104484)

[2. 需求分析 3](#_Toc52104485)

[（1） 功能性需求 3](#_Toc52104486)

[（2） 性能需求 5](#_Toc52104487)

[3. 系统设计 5](#_Toc52104488)

[（1） 第0层设计 5](#_Toc52104489)

[（2） 第1层设计 6](#_Toc52104490)

[（3） 接口描述 7](#_Toc52104491)

[4. 项目总结 8](#_Toc52104492)

[（1） 总体描述 8](#_Toc52104493)

[（2） 完成情况 8](#_Toc52104494)

[二、专题内容分析 9](#_Toc52104495)

[1.前后端的数据交互方法-Ajax 9](#_Toc52104496)

[（1） Ajax简介 9](#_Toc52104497)

[（2） 优缺点 9](#_Toc52104498)

[（3） 开发模式 11](#_Toc52104499)

[2.Tensorflow 12](#_Toc52104501)

[（1） 背景 12](#_Toc52104502)

[（2） 语言支持 12](#_Toc52104503)

[（3） 组件 13](#_Toc52104504)

[3.VGG神经网络模型 14](#_Toc52104509)

[（1） 背景 14](#_Toc52104510)

[（2） 特点 15](#_Toc52104511)

[三、实习中的收获 1](#_Toc52104512)6

[四、对实习工作的改进建议 1](#_Toc52104513)6

# 实习内容分析

## 项目介绍

由于手机拍照功能的完善，当前每个人都有大量的相片产生，整理相片是一件极其繁琐的事情，自动相册工具，通过图像识别，基于深度学习的神经网络算法，能够自动进行图像寓意提取，自动整理照片，生成成套相册，方便用户的管理和使用。

目前大部分手机相册都拥有自动分类的功能。以小米的系统相册为例，主要拥有三种不同的分类。一类是人物，即自动识别不同的人脸归类为不同的相册，由用户自己输入所识别的人脸的名字。一类是地点，推测是通过读取照片拍摄时所包含的位置信息，以此作为分类的依据。第三类则是类别，将照片分为文档、名片、幻灯片、服饰、建筑、风景等众多类别。

因此我们的项目目标是在上传图片后相册可以自动分类，将分类后的图片分门别类地放入各种不同的分类相册中，同时，在查看原相册和分类相册时可以查看图片的详细信息，其中就会显示该图片的标签和一些其他信息，这时可以对图片的标签信息进行修改。

## 需求分析

### 功能性需求

所谓功能性需求是指开发人员必须在产品中实现的软件功能，用户利用这些功能来完成任务，满足业务需求。

前台功能

|  |  |  |
| --- | --- | --- |
| **功能名称** | **子功能** | **描述** |
| “全文搜索”模块 |  |  |
| 用户中心 | 登录信息 | 查看自己的登录信息 |
| 图片查看 | 查看分类好的图片信息 |
| 图片修改 | 对所选择的图片信息进行修改 |
| 分类查看 | 对照片分类进行查看 |

后台功能

|  |  |  |
| --- | --- | --- |
| **功能名称** | **子功能** | **描述** |
| 用户管理 | 用户管理 | 对用户信息进行管理 |
| 权限管理 | 对用户权限进行设置 |
| 权限组管理 | 对不同类型用户的增删改查操作 |
| 标签管理 | 标签添加 | 添加分类标签 |
| 标签修改 | 修改分类标签 |
| 标签查看 | 查看分类标签 |
| 标签删除 | 删除分类标签 |
| 模型管理 | 模型添加 | 添加模型标签 |
| 模型修改 | 修改模型标签 |
| 模型查看 | 查看模型标签 |
| 模型删除 | 删除模型标签 |
| 数据管理 | 查看反馈标签数据 | 对用户的标签数据进行统计 |
| 显示反馈标签数据 | 利用图表形式显示标签数据 |

### 性能需求

系统的性能需求主要有：

|  |  |  |
| --- | --- | --- |
|  | 省局 | 分局 |
| 数据录入更新 | <5秒 | <5秒 |
| 简单数据计算处理 | <5秒 | <5秒 |
| 复杂数据计算处理 | <5秒 | <5秒 |
| 简单数据的查询（不包括图形、视频等） | <5秒 | <5秒 |
| 复杂数据的查询（不包括图形、视频等） | <5秒 | <5秒 |
| 其他图形视频查询 | <10秒 | <15秒 |

（一）响应性能：要求一般操作响应时间<5秒，复杂操作响应时间<20秒；

（二）用户接入数：最大用户接入数量为300；WEB并发使用用户数量为100；

（三）数据存储时间：要求数据库用户设置详细信息在线长期保存，系统数据详细信息要求在服务器中长期保存。

## 系统设计

### 第0层设计

**软件系统上下文**：用户使用自动相册系统，管理员管理自动相册系统，数据库和自动相册系统之间进行数据交互，浏览器将表单提交到自动相册系统中进行处理。

**设计方案**：采用了django做前端界面，pytorch和tensorflow在后台对上传的照片进行处理，将照片分入不同的类别。

**设计约束**：由于采用BS架构，所以在硬件上无特别约束，只是在模型训练上，电脑配置越高越好。

### 第1层设计

主要分为三个模块，分别是图像识别模块、图像分类模块、用户信息模块。图像分类模块基于核心的图像识别模块，图像识别模块对图像进行识别之后，图像分类模块获取识别后的信息之后，将不同标签的图像放入不同的相册中，用户通过用户登录模块登录之后，可以查看当前登录用户的相册。

#### 图像识别模块

**简介**： 图像识别模块是分类相册的核心功能，主要有景、物、人三类。

**功能列表**：识别场景、识别动物、识别不同的人，从单张照片抠下不同的人。

#### 图像分类模块

**简介**：图像分类模块是对图片进行操作的一个模块。

**功能列表**：图像按标签分类放在不同文件夹下。图像上传功能，上传时可选择单张图片上传或者文件夹上传，同时在上传时即会自动分类。选择功能，可以点击照片来选择当前照片，这里提供了全选和取消全选的功能。保存功能，可以将自己上传的照片有选择地保存到相册中。照片转移功能，可以在分类完成之后，对相册自动分类不满意的时候，手动将照片调到应该在的分类中。新建子文件夹功能，可以在分类相册中建立子文件夹，按照自己的喜好对相册进行分类。编辑模式，在相册中开启编辑模式，可以多选照片再进行相应的转移、删除操作。删除功能，对选中的照片进行删除。

#### 用户信息模块

**简介**：用户信息模块主要是对用户信息进行管理。

**功能列表**：注册功能，注册一个用户，存入数据库中。登录功能，输入用户名和密码，进行验证，通过即可成功登录。用户资料修改功能，可以对用户的资料进行修改。

### 接口描述

#### 图像识别模块的接口描述

**名称**：classifyImage

**说明**：对图片进行识别分类。

**定义**：classifyImage（request），Post型，参数request包含了用户信息和图片。

#### 图片分类模块的接口描述

**名称**：auto\_classified\_storage

**说明**：对上传的图片进行保存。

**定义**：auto\_classified\_storage(userName, typeName, image)，Post型，参数userName是当前用户的id，typeName是图片的分类，image是图片。

**名称**：read\_directory

**说明**：上传文件夹。

**定义**：read\_directory(directory\_name)，Post型，参数directory\_name是文件夹路径。

**名称**：get\_total\_img\_urls

**说明**：获取当前用户曾经上传过的图片。

**定义**：get\_total\_img\_urls(username)，Get型，参数username是当前用户的id。

**名称**：get\_specific\_urls

**说明**：获取某分类文件下的所有图片,包括根目录下与子分类文件夹下的图片。

**定义**：get\_specific\_urls(userName, typeName)，Get型，参数userName是当前用户的id, typeName是图片的分类。

## 项目总结

### 总体描述

该项目是自动相册，所谓自动相册就是可以自动对上传的照片进行分类的相册。我们组自己训练了深度学习的模型，对上传的图片进行图像识别，图像识别后，将每张图片放入其对应的分类中，其中，人的相册和其他相册有些不同，人的相册中对不同的人也进行了识别和分类，在人的相册中将不同的人放入不同的子相册。除此之外，我们组还做了新建子相册，图片的转移等功能，在前端界面美化上下了很大的功夫，力求给使用者一个最好的使用体验。

### 完成情况

总体功能基本完成，用户模块和图片识别模块均已完成，图片分类模块中搜索功能没有完成，标签统计没有完成。总体代码大概四五千行，代码缺陷率约为3个/千行。

# 二、专题内容分析

## 1.前后端的数据交互方法-Ajax

我们组对于提交表单等操作采用的是AJAX来交互。

### Ajax简介

Ajax是一种在无需重新加载整个网页的情况下，能够更新部分网页的技术，AJAX = Asynchronous JavaScript and XML.AJAX 是一种用于创建快速动态网页的技术。AJAX 通过在后台与服务器进行少量数据交换，使网页实现异步更新。这意味着可以在不重载整个页面的情况下，对网页的某些部分进行更新。

传统的网页（不使用 AJAX）如果需要更新内容，必须重载整个页面。

有很多使用 AJAX 的应用程序案例：Google Maps、Gmail、Youtube 和 Facebook。AJAX全称为“Asynchronous JavaScript and XML”（异步JavaScript和XML），是一种创建交互式网页应用的网页开发技术。它使用：

使用XHTML+CSS来标准化呈现；

使用XML和XSLT进行数据交换及相关操作；

使用XMLHttpRequest对象与Web服务器进行异步数据通信；

使用Javascript操作Document Object Model进行动态显示及交互；

使用JavaScript绑定和处理所有数据。

### 优缺点

(1).AJAX的优点

<1>.无刷新更新数据。

AJAX最大优点就是能在不刷新整个页面的前提下与服务器通信维护数据。这使得Web应用程序更为迅捷地响应用户交互，并避免了在网络上发送那些没有改变的信息，减少用户等待时间，带来非常好的用户体验。

<2>.异步与服务器通信。

AJAX使用异步方式与服务器通信，不需要打断用户的操作，具有更加迅速的响应能力。优化了Browser和Server之间的沟通，减少不必要的数据传输、时间及降低网络上数据流量。

<3>.前端和后端负载平衡。

AJAX可以把以前一些服务器负担的工作转嫁到客户端，利用客户端闲置的能力来处理，减轻服务器和带宽的负担，节约空间和宽带租用成本。并且减轻服务器的负担，AJAX的原则是“按需取数据”，可以最大程度的减少冗余请求和响应对服务器造成的负担，提升站点性能。

<4>.基于标准被广泛支持。

AJAX 基于标准化的并被广泛支持的技术，不需要下载浏览器插件或者小程序，但需要客户允许JavaScript在浏览器上执行。随着Ajax的成熟，一些简化 Ajax使用方法的程序库也相继问世。同样，也出现了另一种辅助程序设计的技术，为那些不支持JavaScript的用户提供替代功能。

<5>.界面与应用分离。

Ajax使WEB中的界面与应用分离（也可以说是数据与呈现分离），有利于分工合作、减少非技术人员对页面的修改造成的WEB应用程序错误、提高效率、也更加适用于现在的发布系统。

(2).AJAX的缺点

<1>.AJAX干掉了Back和History功能，即对浏览器机制的破坏。

在动态更新页面的情况下，用户无法回到前一个页面状态，因为浏览器仅能记忆历史记录中的静态页面。一个被完整读入的页面与一个已经被动态修改过的页面之间的差别非常微妙；用户通常会希望单击后退按钮能够取消他们的前一次操作，但是在Ajax应用程序中，这将无法实现。

后 退按钮是一个标准的web站点的重要功能，但是它没法和js进行很好的合作。这是Ajax所带来的一个比较严重的问题，因为用户往往是希望能够通过后退来 取消前一次操作的。那么对于这个问题有没有办法？答案是肯定的，用过Gmail的知道，Gmail下面采用的Ajax技术解决了这个问题，在Gmail下 面是可以后退的，但是，它也并不能改变Ajax的机制，它只是采用的一个比较笨但是有效的办法，即用户单击后退按钮访问历史记录时，通过创建或使用一个隐 藏的IFRAME来重现页面上的变更。（例如，当用户在Google Maps中单击后退时，它在一个隐藏的IFRAME中进行搜索，然后将搜索结果反映到Ajax元素上，以便将应用程序状态恢复到当时的状态。）

但是，虽然说这个问题是可以解决的，但是它所带来的开发成本是非常高的，并与Ajax框架所要求的快速开发是相背离的。这是Ajax所带来的一个非常严重的问题。

一 个相关的观点认为，使用动态页面更新使得用户难于将某个特定的状态保存到收藏夹中。该问题的解决方案也已出现，大部分都使用URL片断标识符（通常被称为 锚点，即URL中#后面的部分）来保持跟踪，允许用户回到指定的某个应用程序状态。（许多浏览器允许JavaScript动态更新锚点，这使得Ajax应 用程序能够在更新显示内容的同时更新锚点。）这些解决方案也同时解决了许多关于不支持后退按钮的争论。

<2>.AJAX的安全问题。

AJAX 技术给用户带来很好的用户体验的同时也对IT企业带来了新的安全威胁，Ajax技术就如同对企业数据建立了一个直接通道。这使得开发者在不经意间会暴露比 以前更多的数据和服务器逻辑。Ajax的逻辑可以对客户端的安全扫描技术隐藏起来，允许黑客从远端服务器上建立新的攻击。还有Ajax也难以避免一些已知 的安全弱点，诸如跨站点脚步攻击、SQL注入攻击和基于Credentials的安全漏洞等等。

<3>.对搜索引擎支持较弱。

对搜索引擎的支持比较弱。如果使用不当，AJAX会增大网络数据的流量，从而降低整个系统的性能。

<4>.破坏程序的异常处理机制。

至 少从目前看来，像Ajax.dll，Ajaxpro.dll这些Ajax框架是会破坏程序的异常机制的。关于这个问题，曾在开发过程中遇到过，但是查了一 下网上几乎没有相关的介绍。后来做了一次试验，分别采用Ajax和传统的form提交的模式来删除一条数据……给我们的调试带来了很大的困难。

<5>.违背URL和资源定位的初衷。

例如，我给你一个URL地址，如果采用了Ajax技术，也许你在该URL地址下面看到的和我在这个URL地址下看到的内容是不同的。这个和资源定位的初衷是相背离的。

<6>.AJAX不能很好支持移动设备。

一些手持设备（如手机、PDA等）现在还不能很好的支持Ajax，比如说我们在手机的浏览器上打开采用Ajax技术的网站时，它目前是不支持的。

<7>.客户端过肥，太多客户端代码造成开发上的成本。

编写复杂、容易出错 ；冗余代码比较多（层层包含js文件是AJAX的通病，再加上以往的很多服务端代码现在放到了客户端）；破坏了Web的原有标准。

### 与传统的web应用比较

传统的Web应用交互由用户触发 一个HTTP请求到服务器,服务器对其进行处理后再返回一个新的HTHL页到客户端, 每当服务器处理客户端提交的请求时,客户都只能空闲等待,并且哪怕只是一次很小的交互、只需从服务器端得到很简单的一个数据,都要返回一个完整的HTML 页,而用户每次都要浪费时间和带宽去重新读取整个页面。这个做法浪费了许多带宽，由于每次应用的交互都需要向服务器发送请求，应用的响应时间就依赖于服务 器的响应时间。这导致了用户界面的响应比本地应用慢得多。

与此不同，AJAX应用可以仅向服务器发送并取回必需的数据，它使用SOAP或其它一些 基于XML的Web Service接口，并在客户端采用JavaScript处理来自服务器的响应。因为在服务器和浏览器之间交换的数据大量减少，结果我们就能看到响应更快 的应用。同时很多的处理工作可以在发出请求的客户端机器上完成，所以Web服务器的处理时间也减少了。

## 2.Tensorflow

### 背景

谷歌大脑自2011年成立起开展了面向科学研究和谷歌产品开发的大规模深度学习应用研究，其早期工作即是TensorFlow的前身DistBelief。DistBelief的功能是构建各尺度下的神经网络分布式学习和交互系统，也被称为“第一代机器学习系统”。DistBelief在谷歌和Alphabet旗下其它公司的产品开发中被改进和广泛使用。2015年11月，在DistBelief的基础上，谷歌大脑完成了对“第二代机器学习系统”TensorFlow的开发并对代码开源。相比于前作，TensorFlow在性能上有显著改进、构架灵活性和可移植性也得到增强。此后TensorFlow快速发展，截至稳定API版本1.12，已拥有包含各类开发和研究项目的完整生态系统。在2018年4月的TensorFlow开发者峰会中，有21个TensorFlow有关主题得到展示。

### 语言支持

TensorFlow支持多种客户端语言下的安装和运行。截至版本1.12.0，绑定完成并支持版本兼容运行的语言为C和Python，其它（试验性）绑定完成的语言为JavaScript、C++、Java、Go和Swift，依然处于开发阶段的包括C#、Haskell、Julia、Ruby、Rust和Scala。

我们组使用的是Python所以着重讲Python中的TensorFlow。TensorFlow提供Python语言下的四个不同版本：CPU版本（tensorflow）、包含GPU加速的版本（tensorflow-gpu），以及它们的每日编译版本（tf-nightly、tf-nightly-gpu）。TensorFlow的Python版本支持Ubuntu 16.04、Windows 7、macOS 10.12.6 Sierra、Raspbian 9.0及对应的更高版本，其中macOS版不包含GPU加速。安装Python版TensorFlow可以使用模块管理工具pip/pip3或anaconda并在终端直接运行。此外Python版TensorFlow也可以使用Docker安装。

### 组件

#### 核心组件

分布式TensorFlow的核心组件包括：分发中心、执行器、内核应用和最底端的设备层/网络层。

分发中心从输入的数据流图中剪取子图，将其划分为操作片段并启动执行器。分发中心处理数据流图时会进行预设定的操作优化，包括公共子表达式消去、常量折叠等。

执行器负责图操作在进程和设备中的运行、收发其它执行器的结果。分布式TensorFlow拥有参数器以汇总和更新其它执行器返回的模型参数。执行器在调度本地设备时会选择进行并行计算和GPU加速。

内核应用负责单一的图操作，包括数学计算、数组操作、控制流和状态管理操作。内核应用使用Eigen执行张量的并行计算、cuDNN库等执行GPU加速、gemmlowp执行低数值精度计算，此外用户可以在内核应用中注册额外的内核以提升基础操作，例如激励函数和其梯度计算的运行效率。

单进程版本的TensorFlow没有分发中心和执行器，而是使用特殊的会话应用联系本地设备。TensorFlow的C语言API是核心组件和用户代码的分界，其它组件/API均通过C语言API与核心组件进行交互。

#### 低阶API

**张量**：张量是TensorFlow的核心数据单位，在本质上是一个任意维的数组。可用的张量类型包括常数、变量、张量占位符和稀疏张量。张量的秩是它的维数，而它的形状是一个整数元组，指定了数组中每个维度的长度。张量按NumPy数组的方式进行切片和重构。张量有23种数据类型，包括4类浮点实数、2类浮点复数、13类整数、逻辑、字符串和两个特殊类型，数据类型之间可以互相转换。TensorFlow中的张量是数据流图中的单位，可以不具有值，但在图构建完毕后可以获取其中任意张量的值，该过程被称为“评估”。TensorFlow无法直接评估在函数内部或控制流结构内部定义的张量。如果张量取决于队列中的值，那么只有在某个项加入队列后才能评估。

**变量**：变量是可以通过操作改变取值的特殊张量。变量必须先初始化后才可使用，低阶API中定义的变量必须明确初始化，高阶API例如Keras会自动对变量进行初始化。TensorFlow可以在tf.Session开始时一次性初始化所有变量，对自行初始化变量，在tf.Variable上运行的tf.get\_variable可以在定义变量的同时指定初始化器。在对变量进行共享时，可以直接引用tf.Variables，也可以使用 tf.variable\_scope进行封装。

**数据流图和会话**：TensorFlow在数据流编程下运行，具体地，使用数据流图表示计算指令间的依赖关系，随后依据图创建会话并运行图的各个部分。tf.Graph包含了图结构与图集合两类相关信息，其中图结构包含图的节点和边缘对象，表示各个操作组合在一起的方式，但不规定它们的使用方式，类似于汇编代码；图集合是在tf.Graph中存储元数据集合的通用机制，即对象列表与键的关联。例如当用户创建变量时，系统将其加入变量集合，并在后续操作中使用变量集合作为默认参数。构建tf.Graph时将节点和边缘对象加入图中不会触发计算，图构建完成后将计算部分分流给tf.Session实现计算。tf.Session拥有物理资源，通常与Python的with代码块中使用，在离开代码块后释放资源。在不使用with代码块的情况下创建tf.Session，应在完成会话时明确调用tf.Session.close结束进程。调用Session.run创建的中间张量会在调用结束时或结束之前释放。tf.Session.run是运行节点对象和评估张量的主要方式，tf.Session.run需要指定fetch并提供供给数据字典，用户也可以指定其它选项以监督会话的运行。

**保存和恢复**：TensorFlow的低阶API可以保存模型和学习得到的变量，对其进行恢复后可以无需初始化直接使用。对张量的保存和恢复使用tf.train.Saver。TensorFlow保存的模型使用SavedModel文件包，该文件包含是一种独立于语言且可恢复的序列化格式，使较高级别的系统和工具可以创建、使用和转换 TensorFlow模型为SavedModel。tf.saved\_model API可以直接与SavedModel进行交互，tf.saved\_model.simple\_save用于保存模型，tf.saved\_model.loader.load用于导入模型。

#### 高阶API

**Estimators**：Estimators是TensorFlow自带的高阶神经网络API。Estimators封装了神经网络的训练、评估、预测、导出等操作。Estimators的特点是具有完整的可移植性，即同一个模型可以在各类终端、服务中运行并使用GPU或TPU加速而无需重新编码。Estimators模型提供分布式训练循环，包括构建图、初始化变量、加载数据、处理异常、创建检查点并从故障中恢复、保存TensorBoard的摘要等。

**Keras**：Keras是一个支持TensorFlow、Thenao和Microsoft-CNTK的第三方高阶神经网络API。Keras以TensorFlow的Python API为基础提供了神经网络、尤其是深度网络的构筑模块，并将神经网络开发、训练、测试的各项操作进行封装以提升可扩展性和简化使用难度。在TensorFlow下可以直接导出Keras模块使用。

**Eager Execution**：Eager Execution是基于TensorFlow Python API的命令式编程环境，帮助用户跳过数据流编程的图操作，直接获取结果，便于TensorFlow的入门学习和模型调试，在机器学习应用中可以用于快速迭代小模型和小型数据集。Eager Execution环境只能在程序的开始，即导入tensorflow模块时启用 。

**Data**：tf.data是TensorFlow中进行数据管理的高阶API。在图像处理问题中，tf.data可以对输入图像进行组合或叠加随机扰动，增大神经网络的训练收益；在文字处理问题中，tf.data负责字符提取和嵌入，后者将文字转化为高维向量，是进行机器学习的重要步骤。tf.data包含两个类：tf.data.Dataset和tf.data.Iterator，Dataset自身是一系列由张量构成的组元，并包含缓存、交错读取、预读取、洗牌、投影、重复等数据预处理方法、Iterator类似于Python的循环器，是从Dataset中提取组元的有效方式。tf.data支持从NumPy数组和TFRecord中导入数据，在字符数据处理时时，tf.data.TextLineDataset可以直接输入ASCII编码文件。

## 3.VGG神经网络模型

### 背景

2014年，牛津大学计算机视觉组（Visual Geometry Group）和Google DeepMind公司的研究员一起研发出了新的深度卷积神经网络：VGGNet，并取得了ILSVRC2014比赛分类项目的第二名和定位项目的第一名。

VGGNet探索了卷积神经网络的深度与其性能之间的关系，成功地构筑了16~19层深的卷积神经网络，证明了增加网络的深度能够在一定程度上影响网络最终的性能，使错误率大幅下降，同时拓展性又很强，迁移到其它图片数据上的泛化性也非常好。到目前为止，VGG仍然被用来提取图像特征。

VGGNet可以看成是加深版本的AlexNet，都是由卷积层、全连接层两大部分构成。

### 特点

1. **结构简洁**

VGG由5层卷积层、3层全连接层、softmax输出层构成，层与层之间使用max-pooling（最大化池）分开，所有隐层的激活单元都采用ReLU函数。

1. **小卷积核和多卷积子层**

VGG使用多个较小卷积核（3x3）的卷积层代替一个卷积核较大的卷积层，一方面可以减少参数，另一方面相当于进行了更多的非线性映射，可以增加网络的拟合/表达能力。

小卷积核是VGG的一个重要特点，虽然VGG是在模仿AlexNet的网络结构，但没有采用AlexNet中比较大的卷积核尺寸（如7x7），而是通过降低卷积核的大小（3x3），增加卷积子层数来达到同样的性能（VGG：从1到4卷积子层，AlexNet：1子层）。

VGG的作者认为两个3x3的卷积堆叠获得的感受野大小，相当一个5x5的卷积；而3个3x3卷积的堆叠获取到的感受野相当于一个7x7的卷积。这样可以增加非线性映射，也能很好地减少参数。

1. **小池化核**

相比AlexNet的3x3的池化核，VGG全部采用2x2的池化核。

1. **通道数多**

VGG网络第一层的通道数为64，后面每层都进行了翻倍，最多到512个通道，通道数的增加，使得更多的信息可以被提取出来。

1. **层数更深，特征图更宽**

由于卷积核专注于扩大通道数、池化专注于缩小宽和高，使得模型架构上更深更宽的同时，控制了计算量的增加规模。

1. **全连接转卷积**

这也是VGG的一个特点，在网络测试阶段将训练阶段的三个全连接替换为三个卷积，使得测试得到的全卷积网络因为没有全连接的限制，因而可以接收任意宽或高为的输入，这在测试阶段很重要。

如本节第一个图所示，输入图像是224x224x3，如果后面三个层都是全连接，那么在测试阶段就只能将测试的图像全部都要缩放大小到224x224x3，才能符合后面全连接层的输入数量要求，这样就不便于测试工作的开展。

# 三、实习中的收获

1. 了解如何使用bootstrap进行前端页面的设计，后端采用springBoot+MyBatis进行基本列表的增删改查以及列表的分页条件查询。初步了解React native等移动端框架。
2. 使用bootstrap搭建完成前端主体页面使用js,jquery完成前端页面图片数据的获取以及组件之前的交互。使用ajax完成前后端数据的交互
3. 了解到了前端js里面的许多事件的触发事件，像onmousedown，onmousemove,onmouseup，ondragstart，ondragenter，ondragover等鼠标的移动拖拽事件。Window.getComputedStyle用来获取外联样式与内联样式总和计算之后style样式。设计媒体查询，适配不同设备访问页面的布局。

# 四、对实习工作的改进建议

1. 实现删除功能时，SQL语句的返回值类型错写成List，导致删除功能出Bug.

列表中嵌入后台数据时，经常把$和#弄混，导致代码出现问题。在导入包时，总是显示错了，后来发现时yml里面缺少依赖。经过小组成员一起探讨之后可以快速的找出代码里面的错误所在。另外，断点调试这种方法也帮助我解决了一些问题。

1. 在获取图片信息时，一开始采用前端input file框想获取图片路径，后来发现google等浏览器出于安全问题不支持获取本地文件路径。于是，我们选择图片图片的数据流，将图片的数据流传送到后端去。

在前后台交互发送ajax请求时，经常因为没弄清楚变量的数据类型而出错，后面通过debug找出了数据类型。

1. 有时候在获取一个Dom节点样式时候，无法直接通过document.style读取，因为大部分样式写在外联样式表中，无法直接在Dom节点中读取，这时候得通过window.getComupteStyle来获取改节点的样式。在PC端布局页面时尽量采用响应式布局，div还有img等标签最好不要规定成固定大小，否则在其他尺寸不同的设备访问的时候会导致整个排版错乱。