

分类号_____

学校代码_____

学号_____

密级_____

华中科技大学

硕士学位论文

基于迁移学习与深度学习的汽车 细粒度识别系统的设计及实现

学位申请人：万建伟

学 科 专 业：信息与通信工程

指 导 教 师：袁巍

答 辩 日 期：2017/05/25

**A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree for the Master of Engineering**

**Design and Implementation of Car Fine -grained
Identification System Based on Transfer Learning and
Deep Learning**

Candidate: Wan Jianwei

Major: Information and Communication Engineering

Supervisor: Yuan Wei

Huazhong University of Science & Technology

Wuhan, 430074, P. R. China,

May, 2017

独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的科研成果。尽我所知，除文中已经标明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期： 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权华中科技大学可以将本学位论文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论文。

本论文属于 ☐ 保密， 在_____年解密后适用本授权书。
☐ 不保密。

（请在以上方框内打“√”）

学位论文作者签名：

日期： 年 月 日

指导教师签名：

日期： 年 月 日

摘要

作为智能交通系统的重要组成部分，车型识别系统对于改善交通管理，建设智能城市具有重要意义。相较于传统车型识别系统主要适用于区分货车、客车、汽车等车型大类，汽车细粒度识别系统旨在同一车型大类下，至少识别出目标车型所属厂家，品牌和上市时间等信息，是构建新型车型识别系统的重要基础，同时也具有极大的市场应用前景。

考虑到直接采用深度学习来实现对汽车车型的细粒度识别存在着模型训练成本高昂，训练过程严重依赖大数据支持等问题，本文提出在深度学习的基础之上引入迁移学习方法，通过去除 Inception V3 模型的最后一层后再加上一个全连接层和 Softmax 分类器来构造汽车细粒度识别模型。并经过采集，分类，标记等步骤构建了一个包含“上海大众”旗下所有车型图像的小型数据集来充当训练数据集（共分 27 类，约 30000 张，车图包含各种车型视角和背景环境）。实验结果表明，该模型不但能在较小数据集上实现对任意视角车图的细粒度识别（最优准确度约 80%，增加每类车型的图像数量，模型识别准确率会相应有所提高），同时还能极大地降低训练成本。为进一步提高模型的识别精度，本文还对训练数据集图像进行了 Bounding box 处理，模型识别准确率取得了显著改善。

最后，针对于传统车型识别系统安装及维护困难，缺少市场应用等问题，本文将已训练好的汽车细粒度识别模型安装至 iOS 设备中，并围绕汽车细粒度识别功能开发了相关 App 及服务器程序，成功搭建了一个基于移动设备的多功能汽车细粒度识别系统。与传统车型识别系统相比，该系统更贴近普通用户，市场应用价值大，对深度学习的移动化及商业化应用也具有重要意义。

关键词：汽车细粒度识别 迁移学习 深度学习 分类器

Abstract

As an important part of intelligent transportation system, vehicle identification system is of great significance to improve traffic management and build intelligent city. Compared with the traditional research of vehicle identification which is mainly used to distinguish between trucks, buses, cars and other types of vehicle, the car fine-grained identification aims to identify information such as the manufacturer, brand and time-to-market of the target under the same model category, it is an important foundation to build a new type of vehicle identification system and has a great market prospects.

Considering the direct use of deep learning to achieve the fine-grained identification for cars may have a high cost of training and the training process seriously rely on the support of big data, this thesis propose to introduce a method called transfer learning on the basis of deep learning, and construct the car fine-grained identification model by re-adding a fully connected layer and Softmax classifier after removing the last layer of the Inception V3 model. To verify the effect of the model, this thesis build a small data set (a total of 27 categories, about 30,000 images which taken in a variety of perspective and background environment) which contains images of all kinds of cars of Shanghai Volkswagen for training through the collection, classification, marking and other steps. The experimental results show that this model can not only achieve the fine-grained identification of cars from any angle on a small data set (The best accuracy is about 80%, if increasing the number of images per type, the identification accuracy of model will be improved accordingly), but also greatly reduces the costs for training. In order to further improve the identification accuracy of the model, this thesis processes the training data set with a kind of technique called bounding box, and the identification accuracy of model is significantly improved.

Finally, for the problem of traditional vehicle identification system which installed

and maintained difficulty and lack of market application, this thesis installs the fine-grained identification model for cars which has trained to the iOS device and develops the App and server program around the function of fine-grained identification to build a multifunctional fine-grained identification system for cars which based on mobile device successfully. Compared with the traditional vehicle identification system, this system is more close to the average user and the market application is more extensive. In addition, it is of great significance for the mobile and commercial applications of deep learning.

Key words: Fine-grained identification for cars Transfer learning Deep learning
Classifier

目 录

摘 要.....	I
Abstract.....	II
1 绪论.....	1
1.1 研究背景及意义.....	1
1.2 研究现状.....	3
1.3 本文工作安排.....	5
2 汽车细粒度识别的相关理论介绍.....	7
2.1 CNN 基本原理和设计原则.....	7
2.2 Softmax 与 SVM 相关原理及区别.....	11
2.3 迁移学习原理及现状.....	15
2.4 本章小结.....	18
3 汽车细粒度识别模型的设计及验证.....	19
3.1 源任务模型选择.....	19
3.2 Inception V3 模型结构介绍.....	20
3.3 汽车细粒度识别模型设计.....	24
3.4 车型数据集采集.....	27
3.5 实验设计与分析.....	30
3.6 本章总结.....	38

4 汽车细粒度识别系统设计与实现	39
4.1 系统需求分析	39
4.2 开发环境介绍	40
4.3 主要功能设计	40
4.4 系统交互流程	41
4.5 数据库设计	42
4.6 通信协议设计	44
4.7 车型识别模块设计	45
4.8 汽车细粒度识别功能展示及测试	47
4.9 其余模块功能展示及测试	49
4.10 系统性能测试	52
4.11 本章小结	57
5 结论	58
5.1 论文的主要贡献	58
5.2 进一步工作建议	59
致 谢	60
参考文献	61

1 绪论

1.1 研究背景及意义

作为车型识别系统的重要基础，车型识别技术一直以来都得到了有关方面的持续研究与关注。就目前来看，当前的车型识别系统主要适用于区分如汽车、货车、卡车等相互间外在差异明显的车型，属于粗粒度识别范畴。近年来，随着中国城镇化的快速发展，越来越多的家庭或个人热衷于购买汽车作为代步工具，各大汽车制造商也为满足用户需求相竞推出各种车型，并不断进行迭代更新，导致大量车身外观相似，风格设计相近的汽车涌入城市街道（如图-1.1），不但给交管部门的管理带来了巨大的压力，也对当前车型识别研究提出了新的挑战。

一方面，大量的汽车设计和汽车类型使得汽车成为了一个丰富的对象类，依靠传统的车型识别技术无法进行有效区分或识别（如图-1.3 和图-1.4 所示车型，二者无论是在整体外观还是局部细节上都高度相似）。为了弥补当前车型识别系统的不足，人们逐渐将关注点放在了另一个全新的研究方向上——汽车细粒度识别^[1]。不同于车型粗粒度识别，汽车细粒度识别首先为汽车类别的标注提供了一个独特的层次结构^[2]（从上到下分别为汽车厂家，汽车品牌 and 上市年份，如图-1.2 所示），其主要目的是为了在任意车型视角下区分同一汽车厂家，同一汽车品牌中不同上市年份的汽车车型^[1]。另一方面，汽车市场的蓬勃发展造就了一大批的汽车爱好者和使用者，他们热衷于了解或使用不同的车型，对汽车知识有着强烈的兴趣。这使得汽车细粒度识别技术有着极大的市场需求及应用前景。因此，在某种程度上来说，汽车细粒度识别技术不但对于弥补当前车型识别系统的不足具有重要意义，若能加以有效应用，还能起到服务广大车友，发展汽车经济的作用，具有极大的商用价值。

自 2012 年以来，深度学习中的卷积神经网络模型依靠其特征自提取及特征具有更强的语义性等特性正被越来越多的运用到计算机视觉领域^{[3][4]}。然而，受限于其对大数据集的严重依赖，卷积神经网络虽然在数据集丰富的粗粒度图像分类任务中获

得了很好的效果，但对于子类繁多并缺少足够图像数据的细粒度图像分类任务，直接运用的效果并不理想^[5]。这对于将深度学习引入汽车细粒度识别任务是一种限制，同时也极大地限制了深度学习在其它领域的应用。所幸，迁移学习的出现为把深度学习引入汽车细粒度识别领域带来了机遇，通过把在一个大数据领域已经训练好的模型，应用到另一个小数据领域来实现模型的个性化定制，迁移学习不但打破了深度学习“逢模型必大数据”的局限，还能缩减模型的构建及训练成本^[6]。考虑到当下绝大多数领域无法收集到足够数据的事实，迁移学习为深度学习的大规模应用打下了坚实基础，具有非常广泛的应用前景（比如在医学图像分类，稀有语种研究等小数据领域）。同时在商业应用上，中小企业也可以在大数据条件不足的情况下，通过使用市场上购买的通用运算模型及自身收集的小数据，间接运用人工智能技术。



图-1.1：城市交通拥挤图

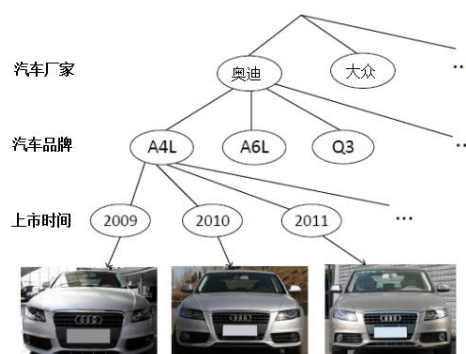


图-1.2：汽车类别标记的层次结构^[1]



图-1.3：上海大众-Polo-2016 款



图-1.4：上海大众-Polo-2013 款

1.2 研究现状

目前来看，国内外对于车型的识别研究主要基于图像处理技术。较早的车型识别均属于车型粗粒度识别，即研究目的主要集中在识别车型大类，如识别汽车，客车，货车，或是识别大型车，中型车，小型车等。传统车型识别的主要方法有：

一、基于模板匹配的车型识别方法

模板匹配属于模式识别，当对待识别的对象归纳出一定的模板后，便可使用该模板来匹配识别其他对象。采用模板匹配方法来进行车型识别的相关研究有许多，如 Zhang^[7]等人通过提出一个新颖的进化计算框架来估计 12 个形状参数和 3 个车型姿势参数，以用于评估车辆模型投影与图像数据之间的拟合度，之后根据拟合度来识别车型，Hsiao^[8]等人根据二维训练图像来构建三维空间曲线，之后使用基于视图的 3D 对准技术将 3D 曲线匹配到 2D 图像曲线以确定汽车车型等。此外，Lin^[9]等人根据 3D 对象表征比传统的 2D 方法包含有更多信息这一结论，通过采用联合优化 3D 模型拟合和细粒度分类成功实现了对车型的识别。虽然基于模板匹配的车型识别方法具有识别过程简单，识别速度快等优点，但其使用易受车型图像的影响，不适用于目标车辆的图像发生旋转，扭曲或不完整等情况。另外，采用该方法也难以区分外部形状高度相似的车型。

二、基于人工特征提取的车型识别方法

该方法整个识别过程可分为三步：采集车辆图像，提取车辆特征，进行车辆识别分类。简单来说，就是先通过收集车型图片构建一个车型样本库，然后通过一定的特征提取方法提取车型样本特征，最后通过训练分类器（SVM，BP 神经网络）来实现车型的识别分类。其中，提取车辆特征是采用人工特征提取方法进行车型识别的核心工作内容，相关研究有李澄非，陈新^[10]提出的融合局部二值模式（LBP）和 Hu 矩特征的车型识别算法，张红兵等^[11]提出的将车前脸图像的梯度方向直方图特征作为识别初始特征，采用线性判别分析算法进行特征提取的车型识别算法，王小龙^[12]提出的通过采用一种投票提升算法，将 24 种识别率不高的 Gabor 特征组合起来进行

车型识别，以提高识别率的方法等。由于车辆特征提取的好坏直接影响最终的车型分类结果，基于人工特征提取的车型识别方法存在较大的不确定性和人力成本。

三、基于深度学习的车型识别方法

近年来，深度学习中的卷积神经网络(CNN)因其特征自提取而无需研究者人工设计车型特征提取策略的特性，正被逐渐运用到车型识别当中。如西南交通大学邓柳^[13]提出的基于深度卷积神经网络来实现车型识别的方法。同时，由于 CNN 还可以保留领域的联系和空间的局部特点，与常见的全连接深度结构相比，CNN 处理实际尺寸的高维图像也毫无难度，因为 CNN 是基于共享卷积核的结构^[14]。然而，若要取得足够好的分类效果，CNN 结构的层数要设计的足够深^[15]。这直接导致模型的参数量巨大，训练过程严重依赖于大数据的支持且训练成本高昂。事实上，对于像汽车细粒度识别等领域而言，是难以采集到足够数据的。

相对于传统车型识别方法，汽车细粒度识别属于一个较新的研究方向，它本质上属于细粒度视觉分类的一个分支。所谓细粒度视觉分类，既是对粗粒度的大类别进行更加细致的子类划分，往往只能通过子类间的部分局部特征进行区分，识别难度更大。近年来，细粒度视觉分类逐渐获得了越来越多的关注。在 2016 年的 CVPR 会议上，就录用了不少关于细粒度视觉分类的文章。从特征的构建到网络架构的改建再到训练手段的创新，都出现了许多新颖的设计方案。例如，来自罗格斯大学的 Han Zhang 等人^[16]通过将研究重点放在图像的富有语义局部，提出了“小语义局部”的生成方法，并证明了在细粒度视觉分类问题上充分利用更多的“小语义局部”可以更好的提高分类结果。另外，针对于细粒度分类数据集数量少，类内差异大，类间差异小等情况，来自 NEC 实验室的 Feng Zhou 等人在本次 CVPR 会议上提出了一系列的解决方法^{[17][18][19]}。

由于汽车可在不同视角下呈现出较大的外观或局部差异，使得汽车细粒度识别与一般的细粒度识别任务有所不同^[1]。就目前所知，将细粒度视觉分类带入汽车识别领域的代表性论文有 Linjie Yang , PingLuo 等人发表在 CVPR 2015 上的 A Large-Scale Car Dataset for Fine-Grained Categorization and Verification^[1]。该文中，作

者首先构建了一个涵盖不同汽车视角及汽车部位的大型汽车数据图集，然后利用 CNN 模型进行车型特征提取，再采用联合贝叶斯网络^[20]作为分类器进行训练，最后实现了在层次结构上识别车型的厂家，品牌和上市年份等信息，但其所需车型数据图集较大，模型训练成本较高。另外，吴志伟^[21]根据实验结果证明了深度卷积神经网络用于车型细粒度识别上的可行性，但其构建的汽车数据集图像主要来源于架设在路边的监控摄像头，背景环境较为单一，且只包含车脸，无法在任意车型视角去进行识别。

1.3 本文工作安排

本文主要以汽车细粒度识别为研究目的，在分析总结了常用车型识别方法优缺点的基础之上，提出了通过“迁移学习+深度学习”来构造汽车细粒度识别模型的方法，从而在较低数据成本和计算成本下实现了对汽车车型的细粒度识别。为了让汽车细粒度识别技术真正服务于广大用户，促进汽车经济，同时也为了解决传统车型识别系统安装及维护困难，识别角度单一，缺少市场应用等问题，本文将训练好的汽车细粒度识别模型安装至移动设备中，并基于汽车细粒度识别功能，开发了一个完善的多功能汽车细粒度识别系统。具体的研究内容及章节安排如下所示：

第一章为绪论，主要介绍了汽车细粒度识别领域的相关研究背景和意义。此外，还介绍了与车型识别有关的国内外研究现状，为后文的详细论述奠定了基础。

第二章主要介绍了汽车细粒度识别所涉及的相关理论。2.1 节详细阐述了卷积神经网络（CNN）的原理及其在图像识别领域使用的优缺点，并给出了深度卷积神经网络的设计原则。2.2 节主要介绍了深度学习领域常用分类器 SVM 及 Softmax 回归的原理及区别，并阐述了它们各自适用的应用场景。2.3 节则详细介绍了迁移学习的相关原理和发展前景，得出了利用迁移学习可将大数据领域学习到的知识迁移到小数据领域的结论，从而为本文提出利用“迁移学习+深度学习”进行汽车细粒度识别模型的个性化定制提供了理论基础。

第三章的主要目的是为了验证采用“迁移学习+深度学习”方法构造的汽车细粒

度识别模型可以以较小数据集和计算成本来完成对汽车车型的有效识别，同时也为之后汽车细粒度识别系统的设计提供已训练好的模型和开发指导。其中，第 3.1 节通过分析比较确定了汽车细粒度识别模型用来进行迁移学习的源任务模型。第 3.2 节则简单介绍了源任务模型 Inception V3 的模型结构。第 3.3 节主要介绍了汽车细粒度识别模型的设计过程及网络结构。第 3.4 节则介绍了车型训练数据集的采集，标记以及分类等步骤，并以“上海大众”为例，构建了一个包含有“上海大众”旗下所有车型品牌的车型数据集，用来作为汽车细粒度识别模型的训练数据集。最后，第 3.5 节通过设计三个实验，一方面验证了第 3.3 节构造的汽车细粒度识别模型适用于汽车细粒度分类任务（在以第 3.4 节中构造的“以车尾为主视角”的车型数据集上采用 bounding box 处理后进行模型训练得到了约 80% 的准确率），另一方面还得出以下结论：1、随着数据集中每类图像数量的增多，汽车细粒度识别模型的总体准确度会相应增加；2、以不同视角的车型图片构建车型数据集，训练得到的模型准确度不同；3、采用 bounding box 消除背景环境的干扰，可以更快，更准确的学习目标车型的特征，从而显著提高分类准确度。这些结论将会用于指导后续汽车细粒度识别系统的设计与开发。

第四章是本文的重点。首先，本章介绍了汽车细粒度识别系统的详细设计过程：包括系统需求分析，开发环境介绍，主要功能介绍，系统交互流程介绍，数据库设计介绍，通信协议设计介绍等。在系统的功能设计中，为提高识别结果的准确率，相关设计参考了第三章得到的实验结论。其次，本章第 4.8 节重点展示了汽车细粒度识别模块的界面设计及功能测试结果，第 4.9 节则展示了系统的其余功能模块及功能测试结果，测试结果均满足设计要求。最后，本章第 4.10 节主要介绍了系统的性能测试结果，包括服务器的压力测试和客户端的内存测试，稳定性测试等。性能测试结果表明，该系统符合实际使用的条件（iOS 客户端代码已上传至 Github：https://github.com/wanjianwei/car_category）。

第五章首先对论文进行了总结，阐述了本文对车型识别领域的主要贡献。其次讨论了汽车细粒度识别系统的主要改进方向，并给出了下一步工作的建议。

2 汽车细粒度识别的相关理论介绍

本章首先介绍了图像识别领域中卷积神经网络（CNN）的基本原理和设计原则，并分析了其使用现状及优缺点。其次介绍了图像分类领域中常用分类器（Softmax 回归分类器和 SVM 分类器）的基本原理及使用区别。最后介绍了迁移学习相关原理，分析了其对促进深度学习应用及发展的影响，为后文针对汽车细粒度识别任务提出解决方法提供了理论依据。

2.1 CNN 基本原理和设计原则

卷积神经网络是一种用于处理具有网格状拓扑结构数据的专用神经网络，简称为 CNN。从命名上看，该网络采用了卷积数学运算，并至少在其一个层中使用卷积代替一般的矩阵乘法。实际上，典型的卷积神经网络层组件主要由卷积和池化操作组成。自从 2012 年 Alex Krizhevsky 等人通过采用卷积神经网络将图像分类误差降到 15%，从而赢得当年的 ImageNet 竞赛冠军以来，CNN 因其特征自提取而无需研究者人工设计特征提取策略的特性逐渐被人们所熟知，并被广泛应用到计算机视觉领域。

2.1.1 CNN 的网络结构

卷积神经网络主要由卷积层，池化层和全连接层构成，针对不同的任务，网络结构设计也不尽相同。现以常用于手写数字识别任务（MNIST）的网络结构 LeNet 为例，来大致介绍 CNN 的构成及操作过程。其中 LeNet 网络结构如图-2.1 所示，图像特征提取过程为：首先输入 32×32 的图片，后经由 5×5 的过滤器（卷积层），得到 28×28 的特征图，再经由 2×2 的池化操作（池化层），得到 14×14 的特征图，之后再经由 5×5 的过滤器（卷积层），得到 10×10 的特征图，完成后再经过 2×2 的池化操作（池化层），特征图 2D 尺寸变为 5×5 ，最后既是进入全连接层。各层的主要功能和详细介绍如下所示：

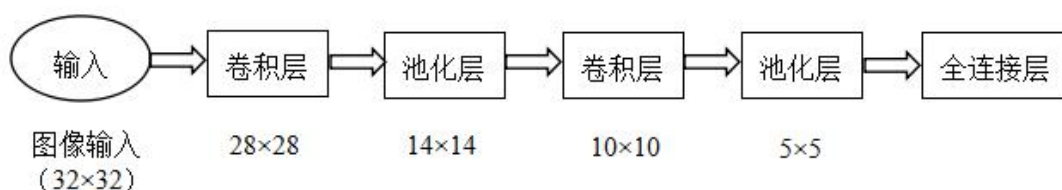


图-2.1: LeNet 网络结构

(1) 卷积层：卷积层的主要功能是用来提取图像特征，卷积层的层数越多，图像特征提取能力就越强。从构成上看，其由多个滤波器叠加组成，每个滤波器即为带有一组固定权重的神经元，也可看做是一组滤波矩阵，矩阵大小设置不同，特征提取效果就不同，具体可根据实际情况来确定。利用神经网络的参数共享特性，可将同一卷积层中不同滤波器的权重设置为相同的，以有效减少网络结构的总体权重数量。图像数据输入卷积层，实际就是将图像数据与滤波矩阵做卷积计算，卷积计算公式为： $s(t) = (x * \omega)(t) = \sum_{a=-\infty}^{\infty} x(a)\omega(t-a)$ ，运算过程见图-2.2^[44]：

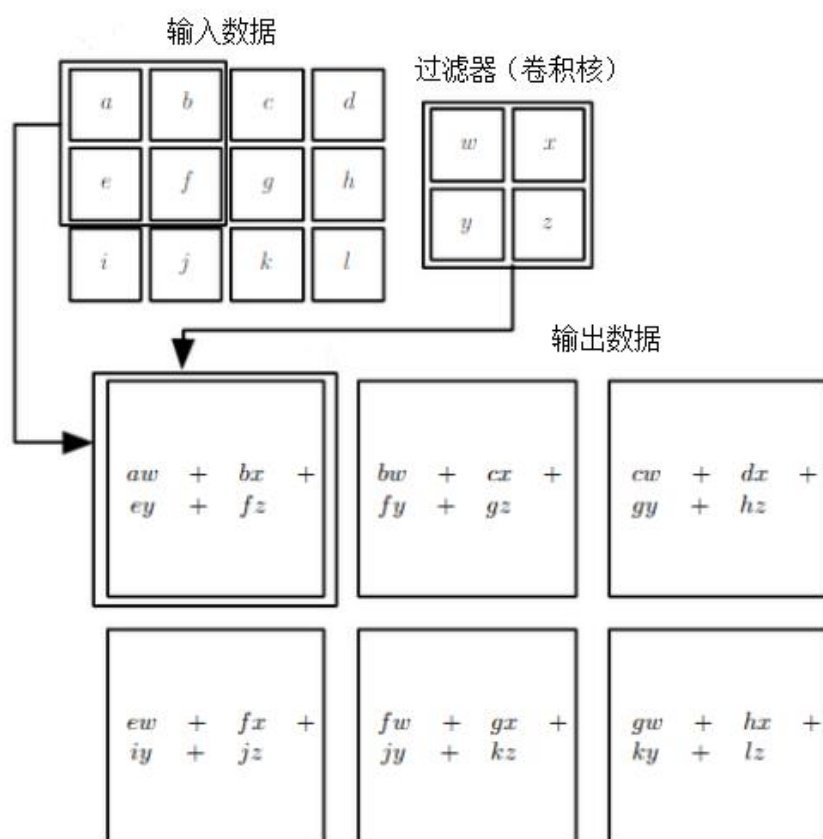


图-2.2: 卷积计算过程^[44]

(2) 池化层：池化层往往跟在卷积层后面，主要目的就是利用图像特征的局部共同性原理，将位置相邻的图像特征进行聚合统计。例如，经过某卷积层后，特征图尺寸变为 20×20 ，此时若设定池化区域为 10×10 ，那么经池化操作后将会得到一个 2×2 的特征图（如图-2.3 所示）。因为池化概括了整个邻域上的响应，有效降低了特征图的维度，所以网络结构的总体参数数量是减少了的，这对提高网络的计算效率，降低过拟合风险具有极大的帮助。

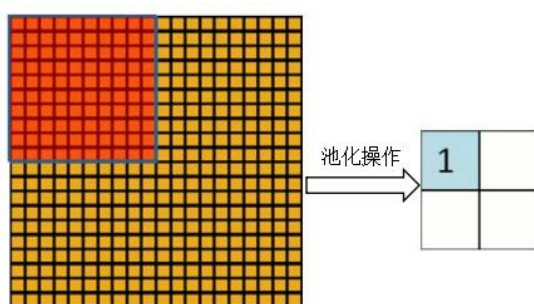


图-2.3：池化操作

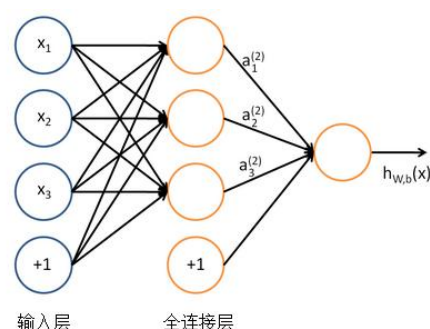


图-2.4：全连接层

(3) 全连接层：在 CNN 网络结构中（如图-2.1），全连接层通常出现在最后几层。全连接层对前一层是全连接的形态（如图-2.4 所示），其主要功能是进行分类或回归，既将样本从特征空间映射到标签。由于全连接层参数较为冗余，当前较新的一些网络模型已不再使用全连接层，而是通过全局平均池化来进行代替。但是近期有研究表明：不含全连接层的网络进行微调（迁移学习中的一种技术，有关迁移学习可见 2.3 节）后的结果要明显比含有全连接层的网络要差。

自从 2012 年的 ImageNet 竞赛冠军被 Krizhevsky 等人获得后，他们团队提出的 AlexNet 神经网络模型^[4]已被成功应用在其他更大型的计算机视觉任务（如：对象检测，人体姿态估计，视频分类，对象跟踪等）当中。这给从事计算机视觉领域研究的学者指明了一个新的研究方向，既专注于找到更高性能的卷积神经网络。2014 年，VGGNet 和 GoogleNet 同时赢得 2014 年 ILSVRC 挑战赛则表明：深度卷积模型（DCNN）中的架构改进可以提高大多数其他严重依赖高质量视觉特征的计算机视觉任务的性能。近三年来看，得益于结构设计更复杂，参数量更多的深度卷积神经网络

络 (DCNN)，图像识别和对象检测的质量确实已经取得了极大的进步。但是，这同时也使得 CNN 的发展受到了不小的限制。以同时赢得 14 年 ILSVRC 挑战赛的 GoogleNet 模型和 VGG 模型为例，前者一共拥有 500 万个参数，而后者网络结构中所拥有的参数是前者的 36 倍。如此大量的参数意味着完成模型训练需要提供丰富的数据集和耗费巨大的计算成本。而实际上，绝大多数领域是无法收集到足够数据的，对于普通的研究者或实验中心来说，也无力承担高昂的计算成本。这让 CNN 的应用在目前为止还仅仅停留在大数据领域（如 ImageNet，NLP 等）。

2.1.2 CNN 模型架构设计原则^[22]

从理论上来说，卷积神经网络层数越多，越能提取高纬度特征，其识别/分类效果就越好。但这并不意味着 CNN 模型架构的设计可以简单的通过增加层数来获得高识别准确度。因为层数越多，参数就越多，而且还会导致模型结构更复杂，由此导致的后果就是需要收集更多的数据和消耗更多的计算成本。因此，设计一个性能优异的 CNN 模型是要进行多方面考虑的。通过分析近年来的一些优秀 CNN 模型，除了要视具体任务而定这一基本原则外，以下还总结得出了其它一些常用设计原则。

（1）在架构设计中包含多个分支来增加路径。历年的 ILSVRC 挑战赛诞生了许多优秀的 CNN 模型，通过分析可以发现，通过增加卷积神经网络架构中的路径数量是最近的一个发展趋势。这一点可从 Alexnet 到 Inception 再到 ResNets 的架构演变中体现出来。除此之外，如 Larsson 等人^[23]和 Ioannou 等人^[24]最新提出的 CNN 架构设计中也体现了这一原则。

（2）增加对称性，并保持架构设计尽可能简单。Springenberg 等人^[25]曾通过实验发现：在模型架构中采用更少的类型单元以保持整体架构的整洁性在相同任务下往往可以得到更好的实验结果。如今，该结论逐渐被其他研究者应用到 CNN 网络设计中，比如 Larsson 等人设计的 FractalNet^[23]就通过增加结构对称来保持整体架构的简洁。

（3）归一化层输入。Ioffe 等人^[26]和 Tim Salimans 等人^[27]都曾通过实验发现，

对层输入进行归一化可以有效改进训练过程和最终准确性，但对其中的原因并未给出详细解释。一般可这样理解，归一化使所有输入层的样本处在更平等的位置，这有利于反向传播的训练。

2.2 Softmax 与 SVM 相关原理及区别

在视觉分类领域中，多分类问题往往是主流，例如手写数字分类任务（MNIST），需要识别 0~9 共 10 个数字图像。通常，人们利用 CNN 模型与多分类器相结合的方式来完成图像多分类任务（如图-2.5 所示）。其中，CNN 模型主要用来从训练数据中学习并提取图像的特征信息，之后将提取到的特征信息通过全连接层输入到多分类器中，而多分类器则主要用来完成图像的最终分类。目前，在深度学习领域常用的多分类器有 Softmax 回归和 SVM（支持向量机），二者在基本原理，分类效果和适用场景上都不相同。正确理解二者之间的区别，对于处理多分类任务具有重要意义。



图-2.5：图像分类过程

2.2.1 Softmax 回归基本原理

Softmax 回归是 logistic 回归模型在多分类问题上的推广，主要用于处理多分类问题。当 Softmax 回归处理的输入数据集总类别数为 2 时，Softmax 回归既退化为 Logistic 回归。所以，若要了解 Softmax 回归的实现原理，需要首先回顾下 logistic 的实现原理。

在 Logistic 回归中，假设训练数据集由 m 个已标记的样本组成：

$\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$ ，其中输入特征 $x^{(i)} \in R^{n+1}$ ，类标记 $y^{(i)} \in \{0, 1\}$ 。根据定义，Logistic 的假设函数数学形式为：

$$h_{\theta}(x) = \frac{1}{1 + \exp(-\theta^T x)},$$

代价函数的数学形式为：

$$J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^m y^{(i)} \log h_{\theta}(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_{\theta}(x^{(i)})) \right]$$

Logistic 回归的主要工作就是通过训练参数 θ 以使代价函数 $J(\theta)$ 最小化。而在 Softmax 回归中，由于针对的是多分类问题，同样在给定输入数据集 $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$ 情况下，类标记 $y^{(i)}$ 的取值范围为 $\{1, 2, \dots, k\}$ ，其中 k 为总类别数，比如说输入数据集总共有 20 种类别，那么 $k = 20$ 。此时，若利用假设函数 $h_{\theta}(x)$ 来估算任意输入 x 在数据集每个类别 j 上的概率值 p ，其中：

$$p(y^{(i)} = j | x^{(i)}; \theta) = \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^k e^{\theta_l^T x^{(i)}}},$$

那么得到的应该是一个 k 维的向量（向量元素均为概率值，归一化后其和等于 1），参考 logistic 回归的假设函数定义，可推算出 Softmax 回归的假设函数数学形式为：

$$h_{\theta}(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1 | x^{(i)}; \theta) \\ p(y^{(i)} = 2 | x^{(i)}; \theta) \\ \vdots \\ p(y^{(i)} = k | x^{(i)}; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^k e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

对应 logistic 回归，Softmax 回归的模型参数为 $\{\theta_1, \theta_2, \dots, \theta_k\}$ ，其中 $\frac{1}{\sum_{j=1}^k e^{\theta_j^T x^{(i)}}}$ 是对

概率分布的归一化操作。Softmax 回归的代价函数数学形式则为：

$$j(\theta) = -\frac{1}{m} \left[\sum_{i=1}^m \sum_{j=1}^k 1\{y^{(i)} = j\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^k e^{\theta_l^T x^{(i)}}} \right],$$

如果将 Logistic 回归的代价函数作如下变换：

$$\begin{aligned} J(\theta) &= -\frac{1}{m} \left[\sum_{i=1}^m y^{(i)} \log h_{\theta}(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_{\theta}(x^{(i)})) \right] \\ &= -\frac{1}{m} \left[\sum_{i=1}^m \sum_{j=0}^1 1\{y^{(i)} = j\} \log p(y^{(i)} = j | x^{(i)}; \theta) \right], \end{aligned}$$

对比后可发现，Softmax 回归的代价函数其实就是 Logistic 代价函数的多维推广。

一般来说，通过最小化代价函数 $j(\theta)$ ，就能得到一个可用的 Softmax 回归分类器。但事实上，目前并没有有效的闭式解法来解决 Softmax 代价函数 $j(\theta)$ 的最小化问题。为了尽可能的使 $j(\theta)$ 最小化，通常的做法是先对 $j(\theta)$ 求导，以获得其偏导数公式，如下所示：

$$\nabla_{\theta_j} j(\theta) = -\frac{1}{m} \sum_{i=1}^m [x^{(i)} (1\{y^{(i)} = j\} - p(y^{(i)} = j | x^{(i)}; \theta))]$$

然后将其偏导数带入到梯度下降法或 L-BFGS 等迭代优化算法中来间接实现代价函数 $j(\theta)$ 的最小化。

2.2.2 Softmax 回归与 SVM 区别

首先，Softmax 回归与 SVM 最直接的差别体现在二者对损失函数（有些时候也称为代价函数）定义不同。同样以输入数据集 $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$ 为例，其中 $x^{(m)}$ 表示输入向量。Softmax 回归的代价函数在 2.2.1 节中已推导出，而 SVM 的损失函数公式如下所示：

$$L = \frac{1}{m} \sum_i \sum_{j \neq y^{(i)}} [\max(0, f(x^{(i)}; W)_j - f(x^{(i)}; W)_{y^{(i)}} + \Delta)] + \lambda \sum_k \sum_l W_{k,l}^2;$$

其中 $f(x^{(i)}; W) = Wx^{(i)}$, 表示得分函数, 是个一维向量, $f(x^{(i)}; W)_j$ 既表示向量中第 j 个元素。 W 表示权重矩阵, λ 是一个超参数, Δ 表示一个固定的边界值, 通常被设置为 1.0; 以任意一个输入数据点为例, Softmax 回归分类器和 SVM 分类器的直观差异如图-2.6^[44]所示。

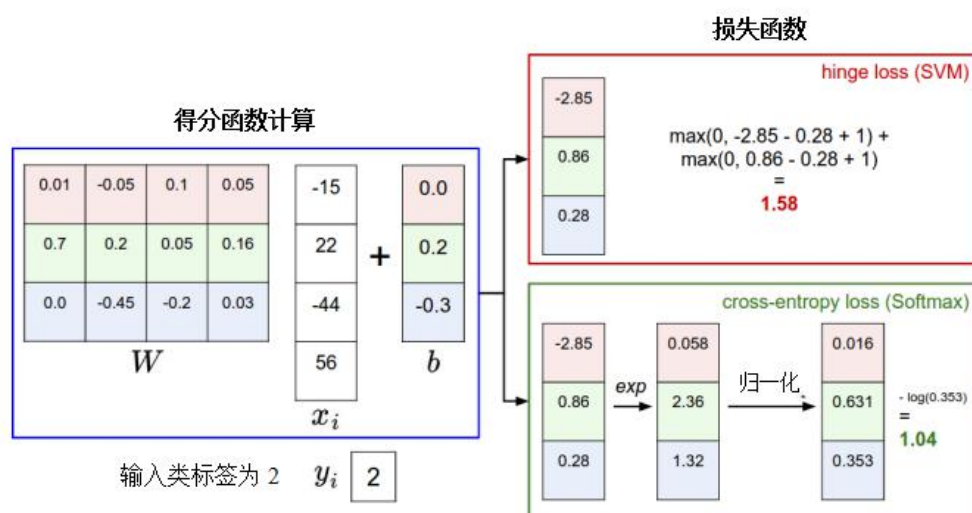


图-2.6: SVM 与 Softmax 回归的直观差异^[44]

其次, 针对任意输入向量, SVM 得分函数计算得到的是所有标签的分数, 结果未进行校准, 难于解释其含义。而 Softmax 分类器得到的则是输入相对于所有标签的概率。比如说, 任意给定输入图像 A, 其中 $A \in \{\text{“猫”}, \text{“狗”}, \text{“猪”}\}$, 经 SVM 分类器后, 得到 1×3 矩阵为 $[12.5, 0.6, -23.0]$, 其中各矩阵元素分别对应“猫”, “狗”, “猪”的分数。而经过 Softmax 分类器后, 假设输出结果为 $[0.9, 0.09, 0.01]$, 那么各矩阵元素代表的是输入图像对应“猫”, “狗”, “猪”的概率。显然, Softmax 分类器的结果更易于理解和解释。另外, 如果输入的图像 A 为“猫”, 那么 SVM 计算得到的损失将为 0, 因为类别“猫”的得分 (12.5) 相对于类别“狗” (0.6) 和“猪” (-23.0) 的得分远多于 Δ , 其中 $\Delta = 1.0$ 。即使最后的得分为 $[9, 1, 1]$, 经 SVM 分类器计算得到的损失仍然为 0。也就是说, SVM 并不关心每个类别的具体分数, 只需满足条件就行。但 Softmax 则

不同，针对 $[9,1,1]$ 计算得到的损失值要高于 $[12.5,0.6,-23.0]$ 。换句话说，Softmax 分类器更看重它所产生的分数，相对越正确的类可能具有越高的概率，并且损失值会更小。

在实际使用中，Softmax 分类器通常拿来与 SVM 分类器进行比较，其实两者的性能差异是很小的，用户完全可以根据实际任务去选择采用哪种分类器。一般来说，CNN 与 Softmax 分类器结合使用的情况更常见。

2.3 迁移学习原理及现状

数据挖掘和机器学习技术已经在很多工程领域取得了显著成功，包括像分类，回归和聚类等。然而，只有当训练和测试数据从相同的特征空间或相同的分布中绘制，大多数的机器学习方法才能取得良好的效果。不幸的是，在许多现实应用中，重新收集所需的训练数据或是重新建立模型的成本代价是及其高昂的。此外，近年来深度学习逐渐成为人工智能领域研究的主流技术，但受限于其对大数据的严重依赖，目前深度学习的应用还仅仅局限于某些大数据领域（如图像识别，语音识别，NLP 等）。如何降低模型的构建成本或计算成本，并使那些在大数据领域取得优异表现的模型依然可以适用于其它小数据领域，这已成为人工智能领域急需解决的问题。在这样一种背景下，迁移学习逐渐走入了人们的视线。顾名思义，迁移学习的主要功能就是利用原始领域或学习任务中学到的知识，来加强目标预测函数的学习^[45]（如图-2.7 所示），其优点可简单概括为在训练开始时学习曲线具有更高的起点，更陡的斜率和更高的渐近性能（如图-2.8 所示）。

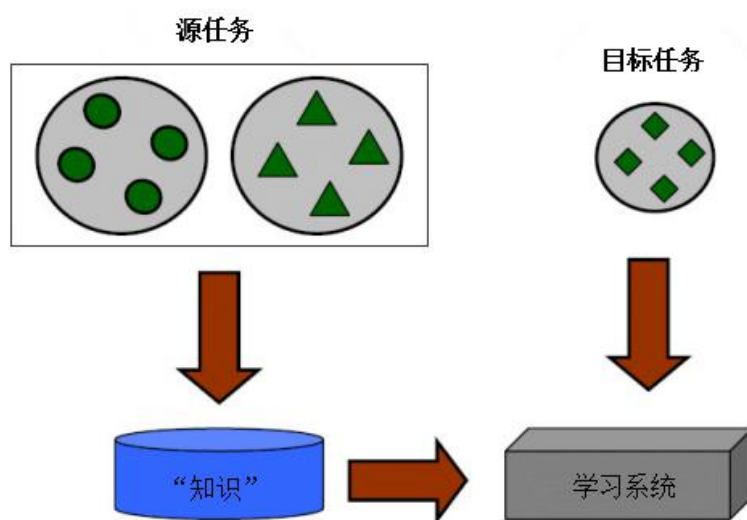


图-2.7: 迁移学习原理

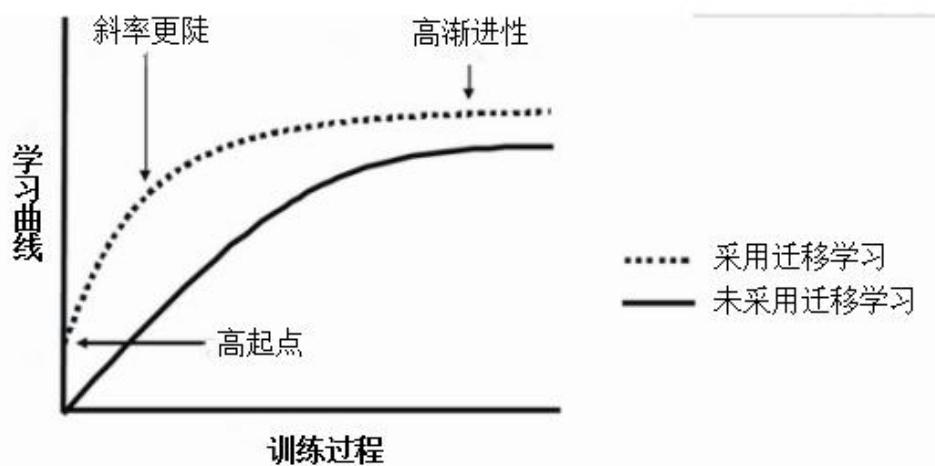


图-2.8: 迁移学习训练过程

在迁移学习中，研究的问题主要有 3 个：1、需要迁移什么，2、如何进行迁移，3、何种情况下进行迁移。其中问题 1 主要是为了弄清哪部分知识是可以进行跨域或跨任务迁移的，因为有些知识可能是某任务或领域所独有的，而有些知识是目标域与源域共有的。若是源域和目标域共有的知识，则可用来改善目标域的学习过程。问题 2 主要是研究采用何种方式进行迁移。关于问题 3，人们更多的是去研究在何种

情况下，不能进行迁移学习。因为若是错误地进行迁移学习，可能会带来一些难以想象的负面效果。比如说，如果目标域与源域之间并无任何联系的话，强制将源域的知识迁移到目标域非但不能改善目标域的学习训练，还会使训练结果变得更差。目前为止，有关迁移学习的大部分研究都集中在问题 1 与问题 2 上。总体上来看，针对于“需要迁移什么”和“如何进行迁移”，迁移学习的研究主要可分为以下 4 个方面^[28]：

(1) 实例-迁移(instance-transfer)：前提是源域中的部分数据可以通过重新加权后继续用于目标域的学习训练中。目前，该领域的主要研究有 Dai 等人^[29]基于 *AdaBoost* 算法提出的 *TrAdaBoost* 算法。该算法通过不断迭代地重新加权源域数据以减少“不良”源数据的影响，同时来鼓励“良好”源数据为目标域的训练或学习贡献更多。对于每一轮迭代，*TrAdaBoost* 都在加权源和目标数据上训练基本分类器，错误只会在目标数据上计算。

(2) 特征-表示-迁移(Feature-representation-transfer)：简单来说就是找到一个“好的”特征表示，以减少源域和目标域之间的差异以及分类和回归模型的误差。对于不同类型的源域数据，找到“好”特征表示的策略是不同的。如果源域中有许多标记数据可用，那么可以采用有监督型学习方法来构建特征表示，Argyriou 等人^[30]认为这与多任务学习领域的常用特征学习相似，并提出一种用于多任务学习的稀疏特征学习方法，可以通过求解最优化问题来学习共同特征，求解公式如下所示：

$$\arg \min \sum_{t \in \{T, S\}} \sum_{i=1}^{n_t} L(y_t, \langle a_t, U^T x_t \rangle) + \gamma \|A\|_{2,1}^2$$

其中 $A = [a_S, a_T] \in R^{d \times 2}$ 是个参数矩阵， U 是个 $d \times d$ 矩阵。如果源域中没有标记的数据可用，则可采用无监督型学习方法来构造特征表示，相关研究有 Raina 等人^[32]提出的采用稀疏编码来学习更高级别的转移学习特征。

(3) 参数-迁移(Parameter-transfer)：该方法假设源域和目标域模型之间存在共享参数，通过将需迁移的知识编码成共享参数来实现跨域知识传递。相关研究主

要有 Lawrence 等人^[31]提出的 *MT-IVM* 算法, 该算法基于高斯过程 (GP), 通过共享同一 GP 之前的多个任务来学习高斯过程的参数。

(4) 关系-知识-迁移 (Relational-knowledge-transfer): 该方法主要是通过构建源域和目标域之间关系知识的映射来处理相关领域间 (如网络数据和社交网络数据) 的迁移学习, 与之前方法不同的是, 该方法假设从每个域获取的数据是具有联系的, 并将这种关系迁移到目标域中。相关研究主要有 Mihalkova 等人^[33]提出的 *TAMAR* 算法, 该算法成功将马可夫逻辑网络 (MLN) 的相关知识迁移到了其他相关领域中。

目前, 迁移学习正吸引着越来越多的研究者加入到此研究中来, 尤其是与深度学习结合后, 迁移学习的优势更加明显。传统深度学习受限于对大数据的依赖, 其应用领域一直十分有限, 与迁移学习结合后, 不但能将深度学习应用到绝大多数的小数据领域, 更重要的是, 这将赋予普通中小企业或高校打破人工智能技术垄断的能力: 即使无法在自身研究领域收集到足够数据, 也可通过购买通用运算模型, 运用深度学习技术。这对推动深度学习的工业化发展具有重要意义。

2.4 本章小结

本章主要对汽车细粒度识别任务所涉及到的理论进行了介绍与分析。虽然卷积神经网络 (CNN) 在图像分类领域引起了巨大变革, 但受限于对大数据的严重依赖, 其并不适合直接应用于小数据领域。然而相关理论及实验表明, 迁移学习可根据原始领域或任务中学到的知识, 来加强目标领域的训练和学习, 从而极大的节省数据及计算成本, 这就为本文提出利用“迁移学习+深度学习”来构造汽车细粒度识别模型提供了理论基础。在分类器的选择上, Softmax 回归分类器与 SVM 分类器在性能上十分相似, 但 Softmax 回归分类器得到的输出是输入相对于每个类别的概率, 结果更易于解释, 与 CNN 结合使用也更为常见。

3 汽车细粒度识别模型的设计及验证

汽车细粒度识别技术是构建汽车细粒度识别系统的核心。通过上一章的分析可知，在计算机视觉领域，相较于传统图像识别方法，采用卷积神经网络（CNN）与常用分类器（SVM 或 Softmax）进行组合来实现图像分类的方法往往可得到更高的识别精度。然而，受限于对大数据的严重依赖和高昂的计算成本，深度卷积神经网络在各领域的大规模应用仍然无法实现，因为当前绝大多数的领域是无法采集到足够的数据来进行深度学习训练的，其中就包括汽车细粒度分类领域。另外，进行深度学习训练的高昂计算成本也让许多研究者或实验中心无法承受。以一台普通电脑为例，若要在 ImageNet^[34]数据集上完成对 GoogleNet 模型^[3]的训练，可能需要几天甚至几周的时间。为在较小数据集及计算成本下实现对汽车车型的细粒度识别，本文在深度学习的基础上，引入了迁移学习，提出了一种基于迁移学习与深度学习的汽车细粒度识别模型构造方法，并通过构建一个小型的车型数据集来进行实验验证，获得的实验结论将直接用于指导后续汽车细粒度识别系统的设计。

3.1 源任务模型选择

汽车细粒度分类任务本质上属于图像分类领域。由 2.1 节可知，深度卷积神经网络（DCNN）已成为该领域的主流技术。根据 2.3 节的介绍，若目标任务与源任务之间具有相关性，则可利用迁移学习将原始领域或任务中学到的知识用于加强目标领域的训练和学习。基于此，表-3.1 列出了近年来图像分类领域常用的几个 DCNN 模型（都已在 ILSVRC-2012-CLS 图像分类数据集上进行过训练），总体代表了当前图像分类领域的最优水平。由于同属图像分类领域，从中选择一个作为汽车细粒度识别模型的源任务模型是再合适不过了。

表-3.1：图像分类领域常用 DCNN 模型比较

模型	大小（单位:M）	Top-1 准确率	Top-5 准确率
Inception V1	26.6	69.8	89.6
Inception V2	44.9	73.9	91.8
Inception V3	108.8	78.0	93.9
Inception V4	184.4	80.2	95.2
Inception-ResNet-v2	237.2	80.4	95.3
ResNet 50	102.5	75.2	92.2
ResNet 101	178.7	76.4	92.9
ResNet 152	241.6	76.8	93.2
VGG 16	553.4	71.5	89.8
VGG 19	574.7	71.1	89.8

理论上来说，当神经网络拥有越多的参数，其函数近似功能越强，效果越好。但分类模型所具有的参数多少也决定了其模型尺寸大小，虽然增加的模型大小和计算成本往往直接转化为大多数任务的质量增益（前提是提供足够的标记数据用于训练），但是计算效率和低参数量仍然是用于各种使用情况的有利因素，尤其是对于要将深度学习模型迁移到移动设备上来说。需要同时考虑到移动设备的内存大小限制，计算能力限制以及该模型最终所能达到的识别精度（该模型最终识别精度受限于其迁移学习模型的精度）。由上表比较来看，Inception V3 分类模型相对体积较小（108.8M），Top-5 准确率又较高（93.9%），考虑到之后要将汽车细粒度识别模型迁移到移动端设备，以将其真正服务于广大车友，并探索深度学习的商业化应用，选择 Inception V3 作为汽车细粒度识别任务的源任务模型最为合适。

3.2 Inception V3 模型结构介绍

Inception V3 网络结构衍生于 GoogleNet 模型。2014 年，VGGNet 和 GoogleNet 同时赢得了 2014 年 ILSVRC 挑战赛，从而使 GoogleNet 模型为人们所熟知。虽然，

VGGNet 与 GoogleNet 在最终的识别精度上相差不大，但二者的模型结构并不相同。VGGNet 模型网络结构简单，计算成本却很大，评估该网络模型需要大量的计算操作。而 GoogleNet 的网络结构却能在严格的内存限制和计算预算约束下完成训练。例如，GoogleNet 仅仅使用了 500 万个参数，相对于使用了 6000 万个参数的 AlexNet 模型结构，减少了 12 倍，而 VGGNet 使用的参数却比 AlexNet 还要多出 3 倍，这意味着 GoogleNet 的网络结构更适合于那些需要合理处理大量数据的大数据场景或者是那些内存及计算能力有限的场景，比如说在移动设备中。传统上来说，获取高性能的卷积神经网络的一般方法是增加模型的深度或每层的宽度，但这同时会带来如下问题：1、参数太多，在训练数据集较小的情况下容易产生过拟合。2、网络层数越多，计算成本越大，模型优化越难完成。为了解决这些问题，GoogleNet 在网络结构设计上进行了一些优化和创新，并总结了以下几个网络设计原则：

1、避免表示瓶颈，尤其是在网络的初始部分；也就是说，在到达用于当前任务的最终特征图之前，特征图大小应该从输入到输出缓慢变小。

2、高维表示在网络内更容易本地处理。通过增加卷积网络中每个瓦片的激活，可获取更多的解离特征，由此产生的网络将训练的更快。

3、空间聚合可以在较低维度上进行嵌入而不产生任何损失。

4、通过平衡每级滤波器的数量和深度可以使网络达到最佳性能。一般而言，增加网络的宽度和深度是有助于得到更高质量的网络。然而，如果二者能够并行的增加，那么就能达到对于恒定量计算的最佳改进。

随着时间发展，googlenet 相继衍生出 Inception V1~V4 等版本。后者都是在前者网络结构基础上进行改进得来。其网络结构中的几个显著特点如下所示：

首先，GoogleNet 网络结构的一个显著特点是没有较大的卷积核，而是通过适当的因式分解得到更多较小卷积核，如：5×5 用两个 3×3 卷积核代替（如图-3.1）。3×3 可用一个 1×3 和一个 3×1 卷积核代替（如图-3.2）这样做可极大降低参数数量和计算成本。

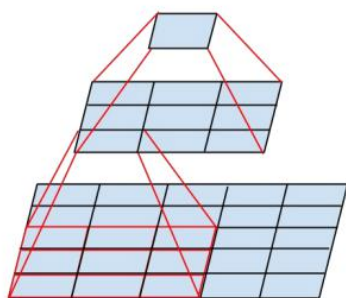


图-3.1

5×5 用两个 3×3 卷积核代替^[3]

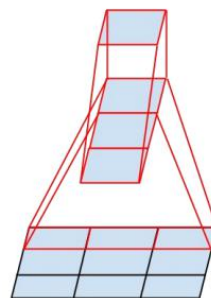


图-3.2

3×3 用 1×3 和 3×1 卷积核代替^[3]

其次，传统方法习惯于采用池化操作来减少特征图的网格大小，为了避免产生代表瓶颈，在运用最大或平均池之前，网络过滤器的激活维度被扩展了，例如，假设开始是一个具有 k 个滤波器的 $d \times d$ 网格，如果我们想要获得具有 $2k$ 个滤波器的 $\frac{d}{2} \times \frac{d}{2}$ 网格，我们首先需要计算一个具有 $2k$ 个步长为 1 的卷积过滤器，然后再应用池化步骤。这意味着总体计算成本的增加。为此，GoogleNet 提出了一种方法（通过使用两个并行的步长为 2 的模块：P 和 C，见图-3.3），不但减少了特征图的网格大小，同时还减少了计算成本，甚至是进一步消除代表瓶颈。

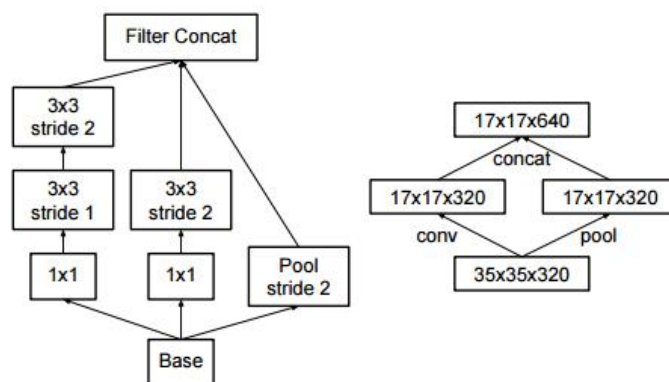


图-3.3：使用两个并行且步长为 2 的模块来减少网格大小和计算成本^[3]

另外，引入了辅助分类器的概念，利用辅助分类器来充当正则化器。这是由如下事实支持的：如果侧分支被批量归一化或是具有丢弃层，那么网络的主分类器将执行的更好。

GoogleNet 的网络布局结构较为复杂，其中的衍生版本 Inception-V2 的网络结构简略介绍如表-3.2 所示：

表-3.2: Inception V2 网络结构^[3]

Type	Patch size/stride	Input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
Conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	如图-3.4 所示 (每个 5×5 卷积核都被替换为两个 3×3 卷积核)	35×35×288
5×Inception	如图-3.5 所示 (N×N 卷积分解后的模块，该架构中 N=7)	17×17×768
2×Inception	如图-3.6 所示 (扩展了滤波器组输出的 Inception 模块，这种架构用于最粗糙的 8×8 网格，以促进高维表示)	8×8×1280
pool	8×8	8×8×2048
linear	logits	1×1×2048
softmax	classifier	1×1×1000

上述所述网络结构中，已将传统的 7×7 卷积因子分解为基于 3 个 3×3 的卷积。对于网络结构中的 Inception 部分，共有 3 个 35×35 的传统 Inception 模块（如图-3.4 所示），每个模块有 288 个滤波器。使用网格缩减技术将其减少为具有 768 个滤波器的 17×17 网格。这之后是 5 个如图-3.5 所示的因子分解的 Inception 模块实例。在最粗的 8×8 级，有 2 个如图-3.6 所示的 Inception 模块，每个瓦片的连接输出滤波器组大小为 2048。

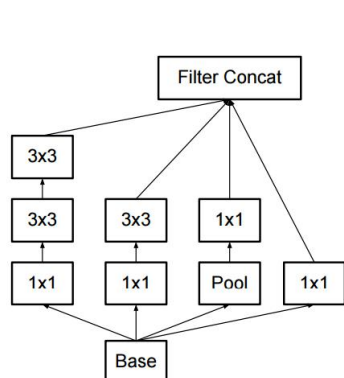


图-3.4: $3 \times \text{Inception}^{[3]}$

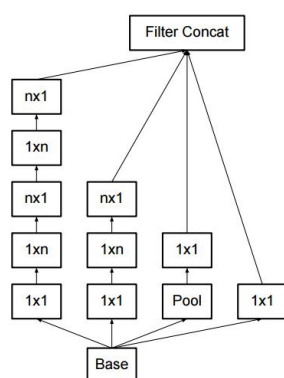


图-3.5: $5 \times \text{Inception}^{[3]}$

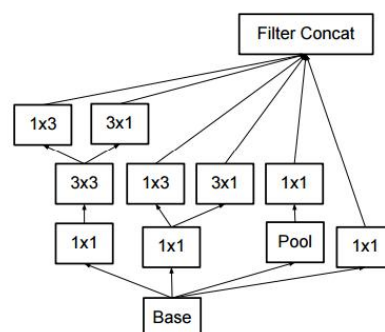


图-3.6: $2 \times \text{Inception}^{[3]}$

3.3 汽车细粒度识别模型设计

由 3.1 节所述可知，在考虑移动设备的内存大小限制，计算能力限制以及该模型最终所能达到的识别精度等情况下，选择已在 ImageNet 数据集上进行过训练的 Inception V3 模型来作为汽车细粒度识别模型的源任务模型是最为合适的。从 2.3 节可知，Softmax 回归分类器与 SVM 分类器的性能差异是很小的。但不同于 SVM 分类器，Softmax 回归分类器可得到任一输入图像相对于各个类别的概率值，这对于扩展汽车细粒度识别模型的实际应用有帮助。比如说，针对任意待识别车型图像，除了识别目标车型的信息外，还希望得到与目标车型最为相似的其它车型有哪些，若采用 Softmax 回归分类器，则可根据输入图像相对于各个类别的概率值来进行排序选取。鉴于此，本文采用 Softmax 回归分类器来作为汽车细粒度识别模型的分器。同样，从 2.1 节叙述可知：不含全连接层的网络进行微调（迁移学习中的一种技术）后的结果要明显比含有全连接层的网络要差。综上所述，本文将汽车细粒度识别模

型的网络结构设计为如图-3.7 所示，主要结构设计为：去除 Inception V3 模型的最后一层，并重新加上一个全连接层和 Softmax 分类器。

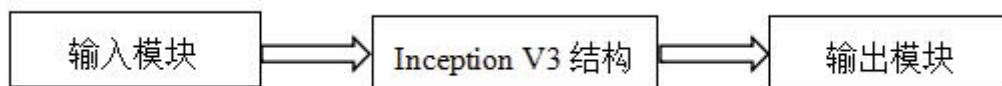


图-3.7：汽车细粒度识别模型网络结构简述

其中，Inception V3 模型在 3.2 节已经介绍过，其网络结构与 Inception V2 的模型结构（表-3.2 所示）类似，该部分主要是利用 Inception V3 模型在 ImageNet 数据集上学到的知识来改善汽车细粒度识别模型的训练与学习过程。输入模块则如图-3.8 所示，其功能主要是用来对任意输入图像进行解码及格式处理。输出模块则如图-3.9 所示，主要负责最终的分类识别及结果校验。图-3.8 与图-3.9 均获取于 Tensorboard，Tensorboard 是 Tensorflow 的图形化、可视化工具，可用于显示 Tensorflow 中由 tensor 和 flow 构成的静态图，以及训练过程中精度、偏差等分析的动态图。图-3.10 则给出了汽车细粒度识别模型输出模块的主要定义代码，编程语言为 Python。

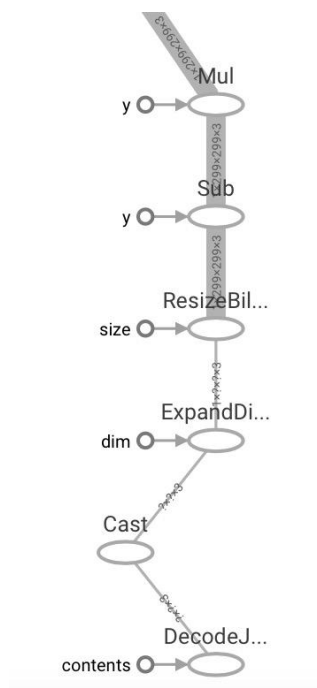


图-3.8：汽车细粒度识别模型的输入模块

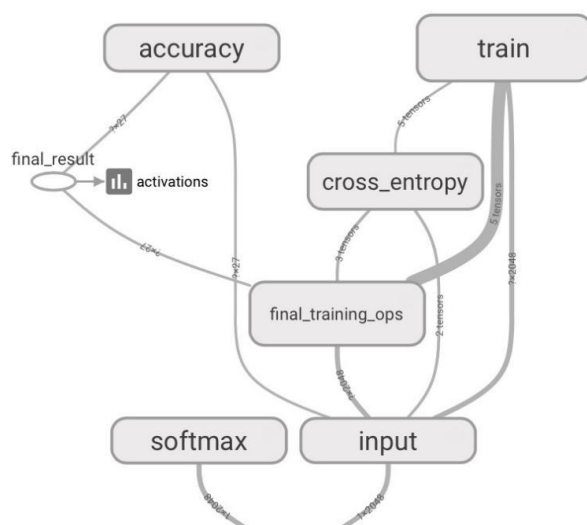


图-3.9: 汽车细粒度识别模型的输出模块

```
# 添加一个softmax回归分类器和全连接层以供模型训练。
def add_final_training_ops(class_count, final_tensor_name, bottleneck_tensor):
    """
    输入参数:
        class_count: 输入数据集的类别总数;
        final_tensor_name: 产生结果的新的最终节点的名称字符串;
        bottleneck_tensor: 主CNN的graph输出
    """
    with tf.name_scope('input'):
        bottleneck_input = tf.placeholder_with_default(bottleneck_tensor, shape=[None,
            BOTTLENECK_TENSOR_SIZE], name='BottleneckInputPlaceholder')
        ground_truth_input = tf.placeholder(tf.float32, [None, class_count],
            name='GroundTruthInput')

        layer_name = 'final_training_ops'
        with tf.name_scope(layer_name):
            with tf.name_scope('weights'):
                layer_weights = tf.Variable(tf.truncated_normal(
                    [BOTTLENECK_TENSOR_SIZE, class_count],
                    stddev=0.001), name='final_weights')
                variable_summaries(layer_weights)
            with tf.name_scope('biases'):
                layer_biases = tf.Variable(tf.zeros([class_count]), name='final_biases')
                variable_summaries(layer_biases)
            with tf.name_scope('Wx_plus_b'):
                logits = tf.matmul(bottleneck_input, layer_weights) + layer_biases
                tf.summary.histogram('pre_activations', logits)

        final_tensor = tf.nn.softmax(logits, name=final_tensor_name)
        tf.summary.histogram('activations', final_tensor)

        with tf.name_scope('cross_entropy'):
            cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits,
                ground_truth_input)

            with tf.name_scope('total'):
                cross_entropy_mean = tf.reduce_mean(cross_entropy)
            tf.summary.scalar('cross_entropy', cross_entropy_mean)

        with tf.name_scope('train'):
            train_step = tf.train.GradientDescentOptimizer(FLAGS.learning_rate)
                .minimize(cross_entropy_mean)

    return (train_step, cross_entropy_mean, bottleneck_input,
        ground_truth_input, final_tensor)
```

图-3.10: 模型输出模块主要定义代码

3.4 车型数据集采集

根据中国近几年汽车市场的销售情况来看，上海大众旗下的汽车品牌在中国的销量常年领先，用户数量较多。从其生产的各类车型来看，子品牌较多，各车型品牌在外观风格上的差异较小，适合用来研究细粒度分类问题。本文即以“上海大众”旗下所有车型品牌为例，分析并构建一个包含有“上海大众”旗下所有品牌车型图像的车型数据集，用来作为车型细粒度识别模型的训练数据集。现对该数据集作如下简要介绍：

1、车型图片主要采集于瓜子二手车、人人二手车等二手车网站或汽车之家、搜狐汽车等汽车网站。

2、针对于传统车型识别方法只采集汽车某一视角或某一部位或某一环境场景下的图片，本文在车型数据集的采集过程中涵盖了各种环境场景和汽车 360 度全视角的图片，如图-3.11 所示。

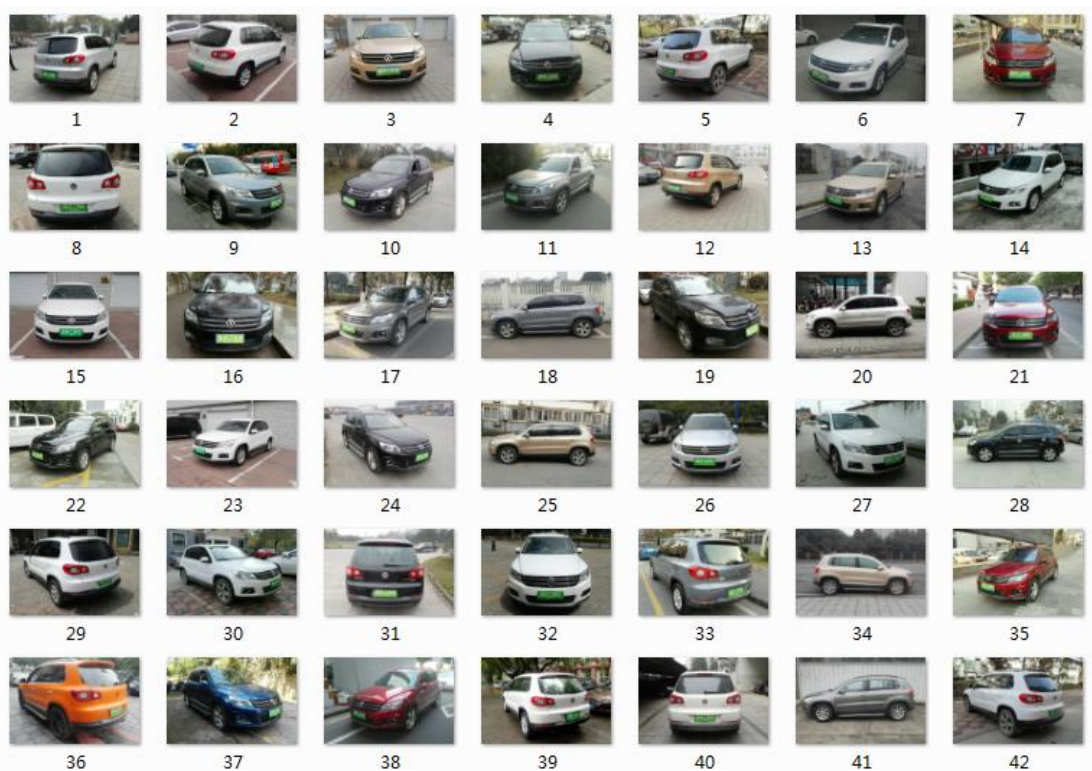


图-3.11：车型数据集的采集过程中涵盖了各种环境场景和汽车视角

3、为研究车型不同部位所包含的特征对车型细粒度分类的影响，从普通用户的拍照习惯出发，在原数据集的基础上将车型数据集又分开标记为“以车头视角为主”的数据集和“以车尾视角为主”的数据集。如图-3.12 与图-3.13 所示：



图-3.12：“以车头视角为主”数据集图片



图-3.13：“以车尾视角为主”数据集图片

4、已完成对“上海大众”旗下所有车型图片的分析、采集及分类标记（约 30000 张图片，共 27 类）。汽车细粒度分类的层次结构如图-3.14 所示，数据集属性标记格式为：厂商_品牌_上市年份_汽车类型。如：上海大众_朗境_2014_紧凑型车或上海大众_桑塔纳志俊_2008 2010_中型车。

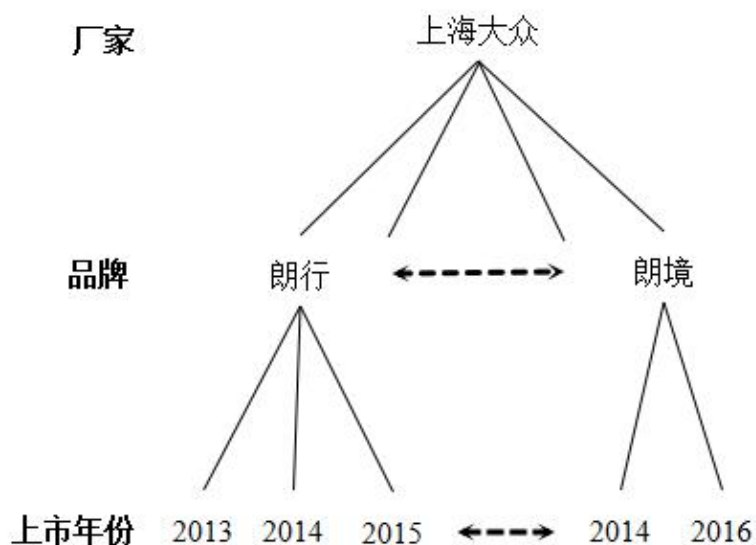


图-3.14：车型标记层次结构

华中科技大学硕士学位论文

注：2008 款和 2010 款的桑塔纳志俊被划分为同一类别，原因在于，2008 款的桑塔纳志俊和 2010 款的桑塔纳志俊无论是在整体外观或是局部细节上都极其类似。也就是说，汽车厂家并未在二者外观上做什么修改。类似的分类标记同样存在于上海大众的其它一些汽车品牌上，数据集详细的分类标记可参看表-3.3：

表-3.3：车型数据集分类标记详情

序号	属性标记
1	上海大众_途安 L_2016_MPV
2	上海大众_途安_2011 2013 2015_MPV
3	上海大众_途安_2008_MPV
4	上海大众_途观_2013 2015 2016_SUV
5	上海大众_途观_2010 2012_SUV
6	上海大众_桑塔纳志俊_2008 2010_中型车
7	上海大众_桑塔纳尚纳_2013 2015_紧凑型车
8	上海大众_桑塔纳浩纳_2015_紧凑型车
9	上海大众_Polo 劲取_2011_紧凑型车
10	上海大众_Polo 劲取_2009_紧凑型车
11	上海大众_Polo 劲情_2007 2009_小型车
12	上海大众_PoloGTI_2012_小型车
13	上海大众_Polo_2014 2016_小型车
14	上海大众_Polo_2011 2013_小型车
15	上海大众_辉昂_2016_高档车
16	上海大众_帕萨特领驭_2009_中型车
17	上海大众_帕萨特_2016_中型车
18	上海大众_帕萨特_2011 2013 2014 2015_中型车
19	上海大众_LAVIDA 朗逸_2015_小型车
20	上海大众_LAVIDA 朗逸_2013 2014_小型车

21	上海大众_LAVIDA 朗逸_2008 2011_小型车
22	上海大众_LAVIDA 朗行_2015_小型车
23	上海大众_LAVIDA 朗行_2013_小型车
24	上海大众_LAVIDA 朗境_2014_小型车
25	上海大众_凌度_2015_小型车
26	上海大众_CrossPolo_2012 2014 2016_紧凑型车
27	上海大众_CrossPolo_2007 2009_紧凑型车

3.5 实验设计与分析

本文的最终目的是要利用“深度学习+迁移学习”来实现汽车细粒度的识别，并设计实现一个基于移动端的汽车细粒度识别系统，以将汽车细粒度识别技术服务大众，同时探索深度学习的商业化应用。设计以下相关实验，既是为了验证第 3.3 节设计的汽车细粒度识别模型的有效性，同时也是为之后汽车细粒度识别系统的设计提供理论指导。首先，配置实验环境如下所示：

实验数据集（由 Polo 系列和朗逸系列构成）：上海大众_LAVIDA 朗逸_2015_小型车、上海大众_LAVIDA 朗逸_2013 2014_小型车、上海大众_LAVIDA 朗逸_2008 2011_小型车、上海大众_Polo_2014 2016_小型车、上海大众_Polo_2011 2013_小型车、上海大众_Polo 劲取_2011_紧凑型车、上海大众_Polo 劲取_2009_紧凑型车、上海大众_Polo 劲情_2007 2009_小型车，并称其为 car_dataset_1；

实验平台：Mac mini（内存：8GB，处理器：2.6GHz Intel Core i5）；

软件工具：Tensorflow（训练过程未采用 GPU 加速）；

训练参数设置：迭代步数为 8000 步，学习速率为 0.01，训练集（Train Set）占比 80%，测试集（Test Set）占比 10%；验证集（Validation Set）占比为 10%；

整个实验共分为三个步骤，具体的实验设计及过程如下所示：

实验一：汽车细粒度识别模型可否在较低数据成本和计算成本下完成对汽车车型的细粒度识别。

设计本实验的目的是为了验证第 3.3 节构造的汽车细粒度识别模型是否能在较低数据成本和计算成本下完成对汽车车型的有效细粒度识别，并探究训练图像数量对模型最终准确率的影响。实验共分 8 组，每组实验的车型数据集单类数量不同，并以每类 50 张递增。其对汽车细粒度识别模型准确率的影响如表-3.4 所示：

表-3.4：实验一实验结果

实验序号	每类数量	最终准确率	accuracy_1 准确度曲线图 (红色表示训练，绿色表示测试)
1	50 张	72.4%	
2	100 张	77.0%	
3	150 张	78.0%	

4	200 张	80.1%	<p>Line graph titled 'accuracy_1' showing accuracy vs. iterations for 200 images. The x-axis ranges from 0.000 to 8.000k iterations. The y-axis ranges from 0.200 to 1.000. A red line (training accuracy) starts at ~0.200 and rises to ~0.95. A blue line (validation accuracy) starts at ~0.200 and rises to ~0.85. Both lines show some initial fluctuation before stabilizing.</p>
5	250 张	81.8%	<p>Line graph titled 'accuracy_1' showing accuracy vs. iterations for 250 images. The x-axis ranges from 0.000 to 8.000k iterations. The y-axis ranges from 0.100 to 0.900. A red line (training accuracy) starts at ~0.100 and rises to ~0.85. A blue line (validation accuracy) starts at ~0.100 and rises to ~0.80. Both lines show some initial fluctuation before stabilizing.</p>
6	300 张	81.4%	<p>Line graph titled 'accuracy_1' showing accuracy vs. iterations for 300 images. The x-axis ranges from 0.000 to 8.000k iterations. The y-axis ranges from 0.100 to 0.900. A red line (training accuracy) starts at ~0.100 and rises to ~0.85. A blue line (validation accuracy) starts at ~0.100 and rises to ~0.80. Both lines show some initial fluctuation before stabilizing.</p>
7	350 张	80.9%	<p>Line graph titled 'accuracy_1' showing accuracy vs. iterations for 350 images. The x-axis ranges from 0.000 to 8.000k iterations. The y-axis ranges from 0.100 to 0.900. A red line (training accuracy) starts at ~0.100 and rises to ~0.85. A blue line (validation accuracy) starts at ~0.100 and rises to ~0.80. Both lines show some initial fluctuation before stabilizing.</p>
8	400 张	83.7%	<p>Line graph titled 'accuracy_1' showing accuracy vs. iterations for 400 images. The x-axis ranges from 0.000 to 8.000k iterations. The y-axis ranges from 0.100 to 0.900. A red line (training accuracy) starts at ~0.100 and rises to ~0.85. A blue line (validation accuracy) starts at ~0.100 and rises to ~0.85. Both lines show some initial fluctuation before stabilizing.</p>

根据表-3.4 所示实验结果，绘制图-3.15 如下所示：

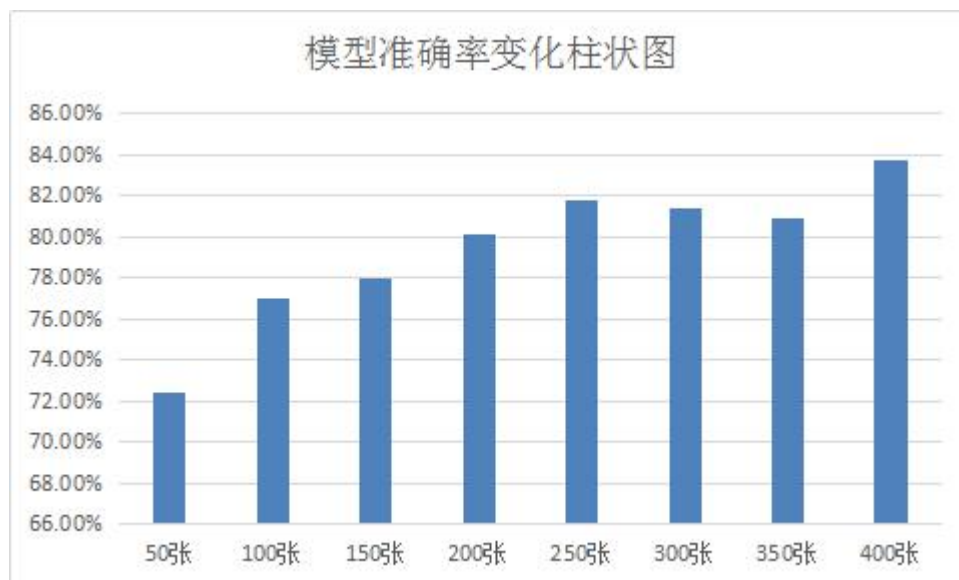


图-3.15：汽车细粒度识别模型准确率变化柱状图

根据图-3.15 及表-3.4 所示结果分析来看，可得出如下结论：

1、随着训练数据集中每类图像数量的增多，汽车细粒度识别模型的总体准确度会相应增加，过拟合现象会逐渐减弱（所谓过拟合现象，直观来说既是训练数据识别效果很好，而测试数据识别效果较差，如表-3.4 中的 `accuracy_1` 曲线图所示，当代表训练数据的红色准确率曲线越优于代表测试数据的绿色准确率曲线时，则表明过拟合现象越严重）。当训练数据集的每类图像数量为 400 张左右时，过拟合现象并不明显，且准确率为 83.7%，完全适用于对汽车进行细粒度识别。

2、从图-3.15 中可知，模型的准确率并非随着每类数量的增加而总体提高，而是在一定范围内出现了抖动。经分析，这主要与车型图像的采集质量有关（采集不同的车型图片来构建训练数据集所获得的准确度不同）。对于像场景处于黑夜中，车身有遮挡，车型图像中存在其它车型，以及一些二手的改装车，若被加入训练数据集，非但不能帮助提高最终模型的准确度，还会不利于模型对汽车特征的学习或提取，影响最终的准确率。

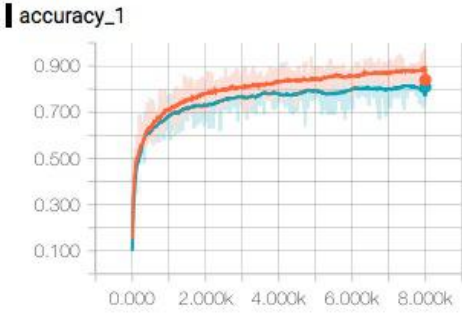
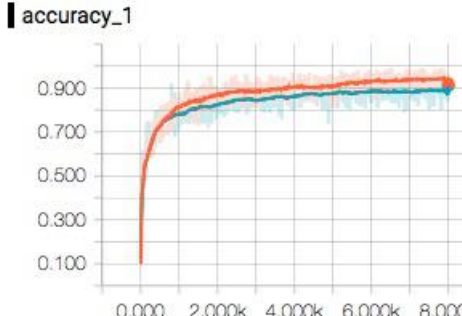
3、从模型的计算成本来看，当训练数据集中每类数量为 400 张左右时，即可在

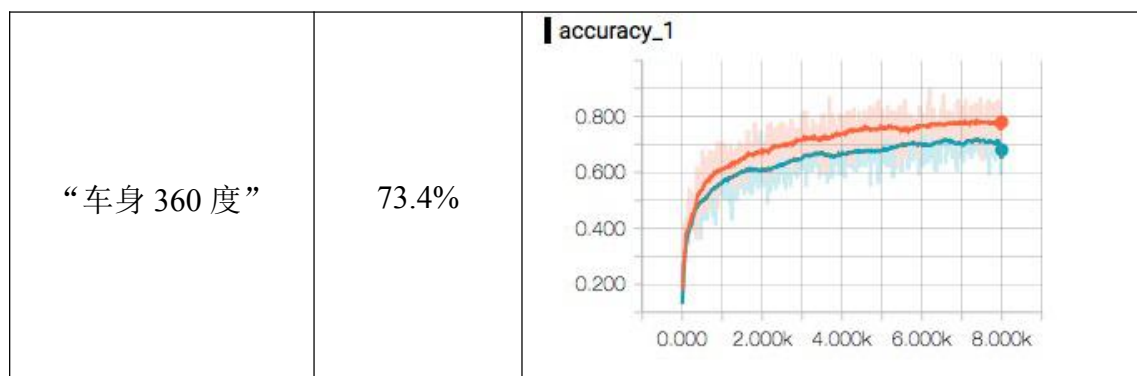
无明显过拟合情况下完成对汽车细粒度识别模型的训练，并且准确率为 83.7%，模型训练时间在上述实验平台上不超过 2 小时。同样情况下，如果直接重新训练其源任务模型 Inception V3 来进行车型识别，所需的单类图像数量将远多于 400 张，模型训练时间也将大大超过 2 小时。这表明，采用“迁移学习+深度学习”方法构造的汽车细粒度识别模型可以在较少数据成本和计算成本下实现对汽车车型的细粒度识别。

实验二：车型不同部位所包含的特征对汽车细粒度识别准确度的影响。

由于汽车可在不同视角下呈现出较大的外观或局部差异，本次实验主要探究车型不同部位所包含的特征对汽车细粒度识别准确度的影响。实验仍然以 car_dataset_1 为实验数据集，将 car_dataset_1 又拆分标记为“以车头视角为主”的数据集（如图-3.12 所示）和“以车尾视角为主”的数据集（如图-3.13 所示）。其中，为保证实验结果具有可比性，“以车头为主视角”的数据集与“以车尾为主视角”的数据集单类图像数大体相等，实验结果如表-3.5 所示：

表-3.5：实验二实验结果

数据集	最终准确率	accuracy_1 曲线图 ((红色表示训练，绿色表示测试))
“以车头为主视角”	80.8%	
“以车尾为主视角”	84.2%	



根据上述实验结果可以发现，以不同视角的车型图片构建车型数据集，训练得到的模型准确度不同；相较而言，车尾部分更容易用来区分。主要原因在于，相对于车前半部分，上海大众旗下车型的车尾部分修改幅度更大，更易于分类。这一实验结果将对之后的汽车细粒度识别系统设计提供参考。

实验三：对车型数据集图片事先采用 Bounding box 技术进行处理是否对汽车细粒度分类模型的准确度有影响；

所谓 bounding box（包围盒，如图-3.16 所示），其原理就是通过一定的方法，将待识别的物体从目标图像中截取出来，再输入到卷积神经网络中进行特征提取。在实验一中，由于车型数据集图片采集于各大二手车平台网站，车型背景环境各式各样，直接将背景复杂的车型图片作为训练数据集，单单通过增加数据集数量来提高模型最终识别准确率，效果并不显著。除去迁移学习模型 Inception V3 的卷积神经网络结构限制外，另一个主要原因就是车型背景环境复杂，影响了卷积神经网络对车型特征的正确提取。若采用 bounding box，消除背景环境的干扰，是否会对最终分类准确率有帮助？相关实验结果如表-3.6 所示：

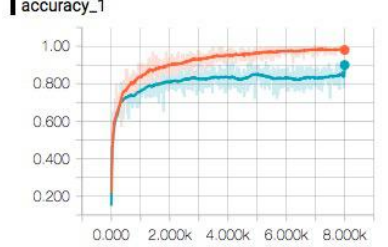
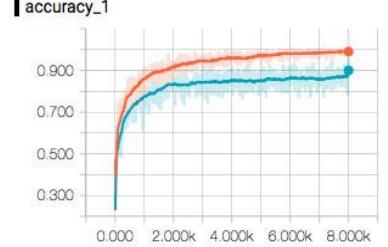
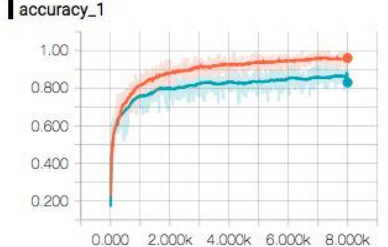
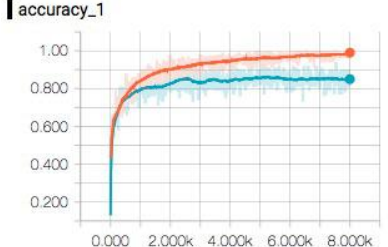


图-3.16: Bounding box 原理

注：本次实验重新构造了实验数据集，主要包含如下类别：上海大众_CrossPolo 2007 2009_紧凑型车、上海大众_CrossPolo_2012 2014 2016_紧凑型车、上海大众_Polo_2011 2013_紧凑型车、上海大众_Polo_2014 2016_紧凑型车、上海大众_PoloGTI_2012_紧凑型车、上海大众_Polo 劲取_2009_小型车、上海大众_Polo 劲取_2011_小型车。实验环境与实验一、实验二相同。

表-3.6: 实验三实验结果（红色曲线表示训练数据，绿色曲线表示测试数据）

实验 序号	单类图像 数量	最终准确度 (未使用 bounding box)	最终准确度 (使用 bounding box)
1	100	<p>准确率：79.7%</p>	<p>准确率：90.1%</p>

2	200	<p>准确率：85.7%</p> 	<p>准确率 92.1%</p> 
3	300	<p>准确率：86.0%</p> 	<p>准确率：92.7%</p> 

从表-3.6 可以看出，采用 Bounding box 对训练数据集图像进行处理后，汽车细粒度识别模型的准确率得到了显著改善。同时，随着训练数据集单类图像数量的增加，过拟合现象逐渐减弱（与实验一类似）。分析上述结果可以发现，通过对车型图像进行 Bounding box 处理能够消除背景环境的干扰，使模型更快，更准确的学习设定车型的特征，从而显著提高分类准确度。该结论对于后续汽车细粒度识别系统的设计具有重要指导意义。

最后，本文在以“车尾为主视角”的整个“上海大众”车型数据集上进行了试验（共 27 类，每类数量 200~700 张），采用 bounding box 处理后可以得到约 80%左右的识别准确率。根据实验一结果可知，随着车型数据集每类图像数量的增加和图像质量的改善，模型最终分类准确率会相应提高。这表明，在深度学习的基础上引入迁移学习来构造汽车细粒度识别模型是可以在较小数据集和计算成本上实现对汽车车型的细粒度识别。

3.6 本章总结

本章主要介绍了汽车细粒度识别模型的设计过程，网络结构以及训练数据集的准备和相关实验的设计等。其中，第 3.3 节主要介绍了汽车细粒度识别模型的设计过程和网络结构。第 3.4 节介绍了训练数据集的采集与标记分类，并构建了一个包含有“上海大众”旗下所有车型品牌的车型数据集（共分 27 类，约 30000 张图片）。第 3.5 节设计了三个实验，验证了采用“迁移学习+深度学习”方法构造的汽车细粒度识别模型可在较小数据成本和计算成本下完成对汽车车型的细粒度识别，并得出以下结论：1、随着数据集中每类图像数量的增多，车型识别模型的总体准确度会相应增加；2、以不同视角的车型图片构建车型数据集，训练得到的模型准确度不同；3、采用 bounding box 消除背景环境的干扰，可以更快，更准确的学习设定车型的特征，从而显著提高分类准确度。该实验所得结论将用于指导后续汽车细粒度识别系统的设计与开发。

4 汽车细粒度识别系统设计与实现

通过上一章的实验可知，基于“迁移学习+深度学习”来构建汽车细粒度识别模型的可在降低训练成本和数据成本下完成模型训练。本章将依据上一章实验得到的结论及识别模型，针对于传统车型识别系统安装及维护困难，识别角度单一，缺少市场应用等问题，设计一个多功能汽车细粒度识别系统，以探索汽车细粒度识别技术及深度学习的商业化和移动化应用。

4.1 系统需求分析

截止 2016 年末，我国汽车驾驶登记人数已超过了 3.1 亿。庞大的汽车用户群体极大的繁荣了汽车市场，也促使各大汽车厂商相竞推出各种各样的汽车车型。面对众多类似款式的汽车，绝大多数的汽车用户或爱好者是难以依据车型外观去了解该车型所属品牌及其详细参数信息的，这为汽车细粒度识别技术的市场应用提供了广阔前景。若能让用户只需通过手机拍照即可识别出感兴趣的车型，并得到该车型的详细参数信息，评价信息和购买信息等，一方面将极大的方便广大对汽车感兴趣的汽车用户，另一方面也可促进汽车市场经济的发展。

另外，深度学习的发展虽然已对计算机科学产生了深远影响，但对普通用户来说，它仍然是个陌生的概念。由于训练成本高昂，严重依赖于大数据支持，使得深度学习的大多数应用均局限于大型数据中心或实验室中，真正应用于市场的并不多。很多时候，用户都是通过本地设备将数据发送给远程数据中心，通过架设在那里的大型计算机来进行处理并返回结果。网络传输可能造成的数据延迟与丢失暂且不说，关键是还需消耗网络流量，用户体验并不好。近年来，硬件设备的发展使得移动设备的计算能力得到了极大的提升，普通手机的 CPU 现在也可以进行每秒数千万次运算，这为把深度学习模型移植到手机上打下了良好基础。通过手机传感器（麦克风，相机等）获取输入数据，手机 CPU 进行深度学习计算，并与应用程序相结合，深度学习能够衍生出许多有意义的应用，从而真正地走入普通用户的生活当中。

基于此，本论文将围绕汽车细粒度识别功能开发相关 App 及服务器程序，以搭建一个基于移动端的多功能汽车细粒度识别系统，来探索汽车细粒度识别技术和深度学习的商业化及移动化应用。所谓基于移动端，即指将汽车车型的识别过程放在移动端本地完成，这意味着需要把之前训练好的汽车细粒度识别模型安装到 iOS 或 Android 设备上，并克服手机存储器及计算能力的各种资源限制，利用移动设备的 CPU 进行深度学习计算。所幸，随着谷歌深度学习框架 Tensorflow（最初由 Google 大脑小组的研究员和工程师开发出来，是一个采用数据流图，用于数值计算的开源软件库，支持在多个平台展开计算，例如台式计算机的一个或多个 CPU 或 GPU，服务器，移动设备等）的开源，将深度学习移植到移动端设备已不再变得困难。

4.2 开发环境介绍

本次汽车细粒度识别系统的开发平台介绍如表-4.1 所示。其中，服务器搭建在阿里云平台上。服务器端的架构方式是 LAMP（Linux+Apache+Mysql+PHP），之所以采用这种架构方式，首先是因为这四个软件都是免费或开源软件，不需要花费额外的成本就可以快速地搭建一个稳定、高效的服务器系统。另外，这种架构模式也是目前 Internet 上流行的网站架构方式。移动端选择在 iOS 设备上开发，主要是因为 iOS 设备的硬件条件，如存储器及 CPU 计算能力等都优于一般 Android 机型。

表-4.1：开发环境介绍

	平台或系统	编程语言
服务器	阿里云+Linux+Apache	PHP
数据库	Mysql	SQL
移动端	iOS	Objective—C

4.3 主要功能设计

为了能让汽车细粒度识别技术真正走入普通用户的生活，并以一定的交互场景

来完成汽车的细粒度识别，本文从用户的实际需求出发，围绕汽车细粒度识别功能，设计了如下主要功能（表-4.2 所示），为使整个系统的功能更完善，相关功能设计还参考了上一章的实验结论。

表-4.2：主要功能表

功能名称	功能简介
车型信息	为用户展示各品牌的车型参数及图片
车型评价	为用户展示其他用户对该车型的评价
车型识别	用户利用手机拍摄车型图片并在本地识别该车型参数，并向用户展示相似车型参数数据。
登录注册	登录注册模块，其中注册采用短信验证码方式。
个人中心	用于显示及修改用户的个人信息。
车图上传	收集用户上传车图，丰富系统训练图集，提高后续分类模型的精度（根据前一章实验一结论：车型数据集数量增多，车型识别最终准确度也会相应提高）

4.4 系统交互流程

依据系统的主要功能，汽车细粒度识别系统的整体交互流程图如图-4.1 所示：

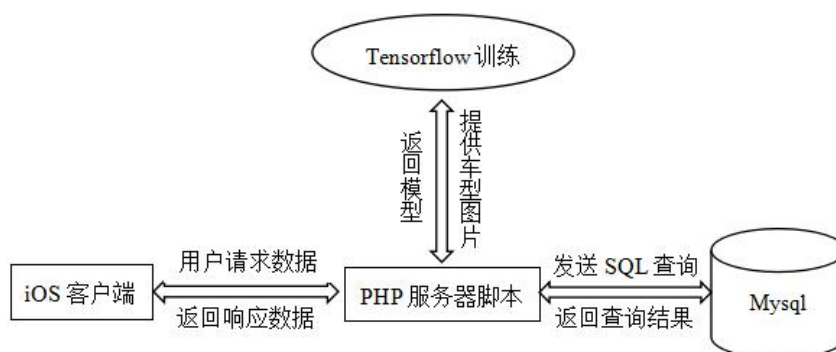


图-4.1：系统流程图

从上述系统流程图中可看出，iOS 客户端向 PHP 服务器发送请求，PHP 服务器即将存储在 Mysql 中的相关数据取出，返回给 iOS 客户端界面展示。之前在系统主要功能设计中已提到，在该系统中，为提高模型的识别准确度，根据第三章实验一的结论，设计有一个“车图上传”功能。该功能的主要目的是收集用户主动上传的车型图片。为实现这一功能，在服务器平台（阿里云）上安装有 Tensorflow 软件，PHP 脚本会根据用户上传的车型图片标记信息，创建专门的类别目录，用来存放用户上传的车型图片。待车型数据集增加了一定数量后，便启动 Tensorflow 重新进行训练，获得最新的模型文件（pb 文件，里面存储有模型结构参数和各滤波器参数），在用户下次打开 App 时，提示用户在下载更新。

4.5 数据库设计

参考系统功能设计需求，并从实际出发，本系统数据库只需存储 4 种类型信息，分别是：用户信息、车型品牌信息、车型评价信息、车型厂家信息。与之对应，共设计 4 张表，依次进行存储。各数据表详细设计信息如表-4.3 所示：

表-4.3：数据库设计

数据表	说明	字段设计
用户信息表	用于存储用户的个人信息,其中 userId 为主键。	1、用户 Id (userId) 2、注册手机 (telephone) 3、注册账号 (username) 4、注册密码 (password) 5、注册时间 (registerTime) 6、性别 (gender) 7、头像 (portrait) 8、个性签名 (sign)

华 中 科 技 大 学 硕 士 学 位 论 文

车型品牌信息表	用于存储品牌车型（如：朗逸，途观等）的参数信息；参数信息来源于搜狐汽车网。其中 car_brand 为主键，manufacturerId 为外键	1、车型品牌（car_brand） 2、所属厂商（manufacturerId） 3、车型报价（car_price） 4、车型结构（car_architecture） 5、车型级别（car_level） 6、油耗(car_MPG) 7、保修(car_warranty) 8、二手车售价（car_secondHand） 9、汽车排量（car_displacement） 10、车型样图（car_demoImg） 11、车型图片（car_images） 12、优点（car_advantage） 13、缺点(car_drawback) 14、参数详情（car_parameters）
车型评价表	用于存储用户对某品牌车型的评价；其中 commentId 为主键，car_brand 和 comment_userId 为外键。	1、评价 Id（commentId） 2、车型品牌（car_brand） 3、评价用户(comment_userId) 4、评价内容（content） 5、评价时间（comment_time） 6、点赞数（praiseNumbers） 7、评论附图（comment_images）
车型厂家信息表	用来存储汽车制造商的详细信息如：上汽大众，华晨宝马；其中 manufacturerId 为主键。	1、汽车厂商 Id（manufacturerId） 2、汽车厂商 Logo（car_logo） 3、汽车厂商名（car_manufacturer） 4、汽车经销商（car_dealership）

4.6 通信协议设计

由于客户端 App 需要与服务器进行通信来实现系统的其它功能，如获取额外的车型或用户信息等，所以需要设计一套完善的通信协议来规范二者之间的通信过程。从服务器角度来看，其主要实现了如下几个功能（如表-4.4 所示），针对每个交互功能，本系统都设计了对应的通信协议，以“获取某汽车厂商下所有车型品牌的基本信息”功能为例，其样式如表-4.5 所示。

表-4.4：服务器功能接口

功能	接口
用户登录	login.php
获取注册验证码	getRandomNumber.php
用户注册	register.php
获取某汽车厂商下所有车型品牌的基本信息	getCarInfo.php
获取某一车型的详细信息	getCarDetailInfo.php
查看车型评论	checkUserComment.php
点赞车型评论	praiseForComment.php
发布车型评论	publishComment.php
查看用户个人信息	checkMyInfo.php
修改用户个人信息	changeMyInfo.php
上传用户头像	uploadPortrait.php
上传车型图像	uploadCarImages.php
获取所有汽车厂商或某厂商下所有车型品牌	chooseManufacturerOrBrand.php
用户退出	logout.php

表-4.5：通信协议示例

功能说明	功能：请求获取某汽车厂家下所有车型品牌基本信息 接口地址：getCarInfo.php	
request	Direction	移动客户端——>web Server
	传输协议	http post
	数据格式	JSON
	请求参数示例	{“car_manufacturer”:”上海大众”}
	返回数据示例	成功： {“state”:true,”returnInfo”:[{“car_brand”:”朗逸”,“car_demoImg”:”123.jpg”,“car_price”:”12~20”,“car_IMG”:”5.7-6.9L/100km”,“car_level”:”紧凑车型”},{“state”:false,”message”:”获取车型信息失败”}]}
	操作数据表	车型品牌信息表（carBrandInfoTable）

4.7 车型识别模块设计

在该系统中，汽车细粒度识别模块是主要核心功能，其主要通过安装在移动端本地的汽车细粒度识别模型来完成目标车型的识别，无需与服务器进行通信，待识别车型图像则主要是通过自定义相机拍照来获取，设计界面如图-4.2 所示：

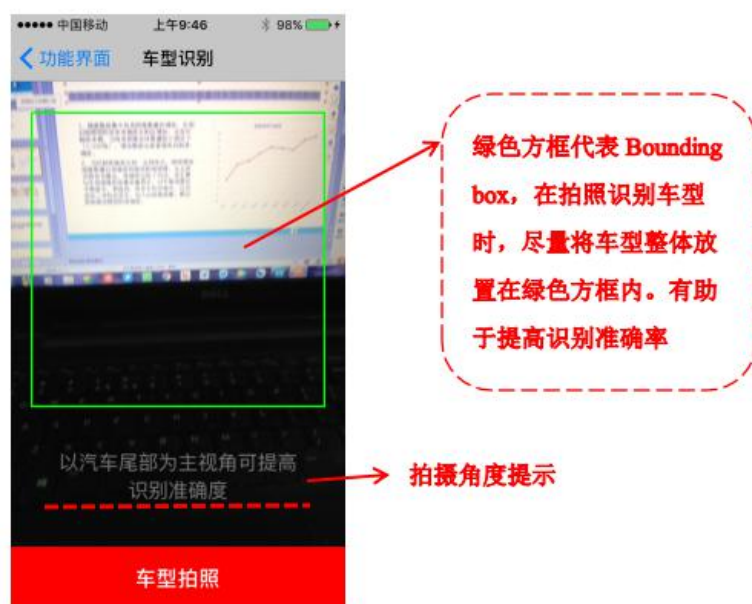


图-4.2: 汽车细粒度识别模块

在上图自定义相机界面中，添加了一个绿色方框，用来提示用户尽量调整相机焦距或距离以将待识别汽车车型放置在绿色方框内之后再点击“车型拍照”按钮。当用户点击“车型拍照”按钮后，App 会将绿色方框内的图像截取出来，作为待识别的输入图像。之后 App 调用本地安装的汽车细粒度识别模型对其进行识别，并返回最终识别结果。该功能的设计主要是参考了第三章实验三的结论：**对原始图像采用 Bounding box 技术以去除背景环境的干扰，可显著提高识别准确度**。实际上，在“车图上传”功能上同样采用了类似的设计（在相机界面上添加一个代表 Bounding box 的绿色方框），目的是为了使系统收集到的车型图像均采用 Bounding box 处理过。

在图-4.2 中，绿色方框下有灰色的提示文字，用于提示用户拍摄车型的什么部位可以更容易识别。该功能设计主要参考了第三章实验二的结论：**以不同视角的车型图片构建车型数据集，训练得到的模型最终识别准确度不同**。例如“上海大众”旗下各车型品牌，经过实验表明，车尾部分相较于车前部分改动更大，更容易区分。那么，为了提高汽车细粒度识别模型的准确度，车型训练数据集主要会由“以车尾视角为主”的车型数据集构成。提示文字则会提示用户：以汽车尾部为主视角可提高识别准确度。

4.8 汽车细粒度识别功能展示及测试

汽车细粒度识别模块的 App 端界面设计如下图所示，现对其操作流程介绍如下：图-4.3 为汽车识别模块的拍照界面，按照之前“车型识别模块设计”的介绍与要求调整相机后，点击“车型拍照”按钮，即可得到图-4.4 所示界面，该界面主要展示拍摄得到的车型图像。如果用户对拍摄的车型图像不满意，可以点击右上角“重拍”按钮，回到图-4.3 所示界面。当用户点击“车型识别”按钮后，App 随即在本地调用汽车细粒度识别模型对车型图像进行识别。识别成功后跳转到识别结果展示界面（图-4.5），该界面不但展示已识别的目标车型的品牌及上市时间信息，还会同时展示与该目标车型外观相近的类似车型，并按照相关程度从高到低排列，给用户更多的选择。若对识别结果中所展示车型的详细信息感兴趣，可点击对应单元格，即可跳转到车型详细信息展示界面（图-4.6）。需要注意的是，车型的详细信息是向服务器请求获得的，需要在手机联网情况下才能完成，而车型的识别过程则是在本地设备中完成的，无需消耗网络流量。

由第三章实验可知，本系统所使用的汽车细粒度识别模型在以“车尾为主视角”的整个“上海大众”车型数据集上进行了试验（共 27 类，每类数量 200~700 张。其中，训练数据占比 80%，测试数据占比 10%，验证数据占比 10%），采用 bounding box 处理后可以得到约 80%左右的识别准确率。当利用 Tensorflow 框架将 PC 端上训练好的汽车细粒度识别模型迁移至 iphone 手机上时，考虑到手机内存的限制，Tensorflow 会对模型的结构做一些删改，以缩减模型体积，这会使得在手机端上的模型识别准确率差于 PC 端上的结果。目前，本文已使用该功能对上海大众旗下的部分车型进行了识别验证，效果良好。图-4.3~图-4.6 展示的既是使用该功能对上海大众旗下 2015 款朗行进行识别的过程，从识别结果看，符合预期设定。



图-4.3: 车型拍照

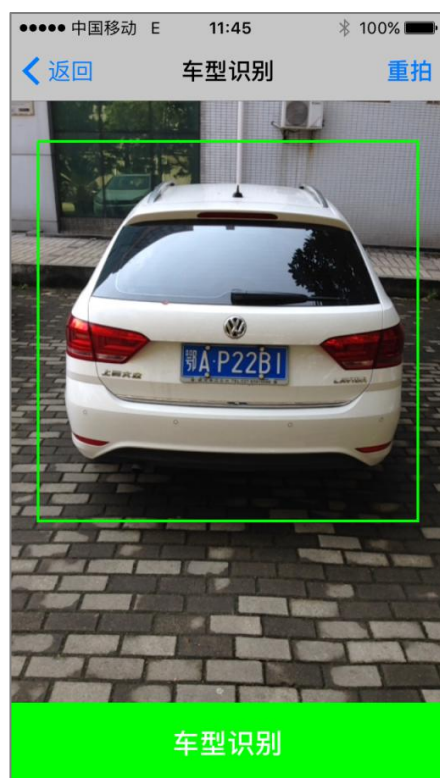


图-4.4: 车型识别



图-4.5: 识别结果展示



图-4.6: 车型详细信息

4.9 其余模块功能展示及测试

汽车细粒度识别系统的其余功能模块如图-4.7~图-4.16 所示，所列功能均进行过功能测试，测试结果符合设计要求。各功能模块简略介绍如下所示：

图-4.7 与图-4.8 为车型信息展示模块，其中图-4.7 展示的是某汽车厂商旗下所有车型品牌的简单信息，包括车型品牌名、车型售价、油耗和车型级别等。用户可通过点击该界面的导航栏来跳转到汽车厂商选择界面（如图-4.15），以选择展示不同汽车厂商下所有车型的品牌信息。图-4.8 展示的是车型品牌的详细信息，主要包括车型图片，车型参数，车型评价及经销商详情等。

图-4.9 与图-4.10 为注册登录模块，其中注册采用短信验证码形式。

图-4.11 为个人信息展示模块，可供用户修改性别，昵称，个性签名和密码。

图-4.12 为车型评价模块，用于展示某一车型品牌的所有用户评价，用户可针对某条评论进行点赞。

图-4.13 为车图上传模块，主要是用于收集用户上传的车图以丰富系统的训练数据集，提高后续模型的识别精度。服务器每隔一段时间则会启用 Tensorflow 重新训练汽车细粒度识别模型，并通知 iOS 客户端进行下载更新。

图-4.14 为评论发布模块，可供用户发布对某一车型品牌的评价。

图-4.15 为汽车厂商选择模块，图 4.16 则为用户操作指南模块。



图-4.7: APP 首页



图-4.8: 汽车型详细信息

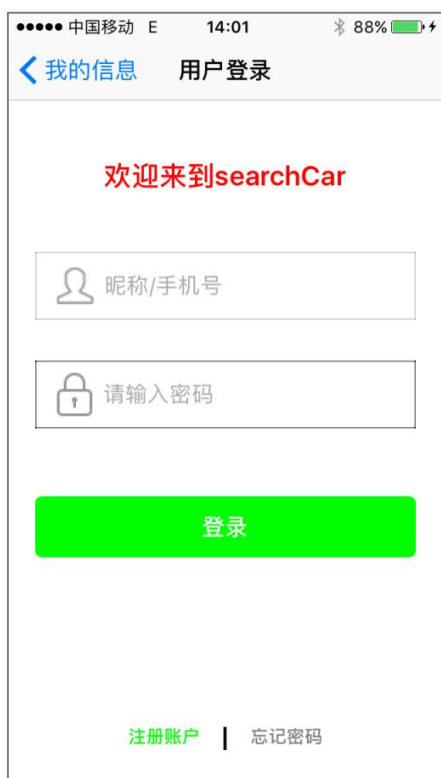


图-4.9: 登录界面



图-4.10: 注册界面



图-4.11：用户信息界面



图-4.12：车型评论界面



图-4.13：上传车图



图-4.14：发表车型评论



图-4.15: 汽车厂家选择



图-4.16: 用户指南

4.10 系统性能测试

在完成系统的开发工作后, 本文从实际使用的角度出发, 对汽车细粒度识别系统进行了功能测试和性能测试。其中, 汽车细粒度识别系统的“功能测试”部分在第 4.8 节和 4.9 节已经介绍过, 本章节主要介绍系统的性能测试结果。由于汽车细粒度识别系统主要由客户端和服务端两个部分组成, 所以本系统的性能测试主要包括客户端性能测试和服务端性能测试。

4.10.1 服务器性能测试

随着用户数量的持续增长, 系统服务端需要支撑的访问量会越来越大, 服务端的性能问题将会逐渐凸显出来。从实际使用的角度来说, 人们往往比较关心服务端的承载量, 因此在服务器的性能测试上, 更关注压力测试结果。在实际使用中, 汽车细粒度识别系统的服务端架构方式为 Linux+Apache+Mysql+PHP, API 共有 14 个

（如表-4.4 所示），从系统权限的角度来看，可分为无需登录即可请求的 API 和需要登录后才能请求的 API。本文从中挑选出两个业务访问量大，可能对系统造成性能瓶颈的接口来进行测试，分别为“获取某汽车厂商下所有车型品牌的基本信息”接口（getCarInfo.php，为 App 首页界面的请求 API，无需进行用户登录即可完成请求，首页界面设计如图 4.7 所示）和“登录”接口（login.php，登录界面的请求 API，界面设计如图-4.9 所示）。压力测试工具采用 Apache 服务器自带的 Apache ab，其特点是功能简单，但实用性强。测试结果分别如图-4.17 和图-4.18 所示，其中部分参数解释如表-4.6 所示。

表-4.6：主要参数解释

Concurrency Level	虚拟用户数，也可看做是并发请求数
Time taken for tests	所有请求处理完成所花费的时间
Complete requests	本次测试所完成的请求总数
Request per second	每秒处理请求数，该数值很大程度上体现了测试接口的性能情况。
Time per request	每个请求花费的时间，即平均事物响应时间

```
Server Software:      Apache/2.4.7
Server Hostname:      120.25.162.238
Server Port:          80

Document Path:        /searchCar/getCarInfo.php
Document Length:      95 bytes

Concurrency Level:     100
Time taken for tests:  0.301 seconds
Complete requests:     1000
Failed requests:        0
Write errors:           0
Total transferred:     296000 bytes
HTML transferred:      95000 bytes
Requests per second:   3318.43 [#/sec] (mean)
Time per request:      30.135 [ms] (mean)
Time per request:      0.301 [ms] (mean, across all concurrent requests)
Transfer rate:          959.23 [Kbytes/sec] received

Connection Times (ms)
              min    mean[+/-sd] median    max
Connect:        0      0   0.9      0      4
Processing:      5     28   4.2     29     34
Waiting:         4     28   4.3     29     34
Total:           8     29   3.6     29     36

Percentage of the requests served within a certain time (ms)
 50%      29
 66%      30
 75%      30
 80%      30
 90%      31
 95%      32
 98%      32
 99%      33
100%      36 (longest request)
```

图-4.17: getCarInfo.php 测试结果

```
Server Software:      Apache/2.4.7
Server Hostname:     120.25.162.238
Server Port:         80

Document Path:       /searchCar/login.php
Document Length:     76 bytes

Concurrency Level:    100
Time taken for tests: 0.343 seconds
Complete requests:    1000
Failed requests:      0
Write errors:         0
Total transferred:    472000 bytes
Total POSTed:         187000
HTML transferred:     76000 bytes
Requests per second:  2916.17 [#/sec] (mean)
Time per request:     34.291 [ms] (mean)
Time per request:     0.343 [ms] (mean, across all concurrent requests)
Transfer rate:        1344.17 [Kbytes/sec] received
                      532.54 kb/s sent
                      1876.72 kb/s total


Connection Times (ms)
              min    mean[+/-sd] median    max
Connect:      0      0    1.0      0      4
Processing:    3     32    6.1     32     43
Waiting:      3     32    6.2     32     42
Total:        7     32    5.5     32     43


Percentage of the requests served within a certain time (ms)
 50%      32
 66%      33
 75%      35
 80%      36
 90%      38
 95%      39
 98%      40
 99%      40
100%      43 (longest request)
```

图-4.18: login.php 测试结果

从图-4.17 中可知，getCarInfo.php 接口测试的并发请求数（Concurrency Level）设置为 100，其在 0.301 秒内（Time taken for tests）共完成了 1000 次请求（Complete requests），每秒处理请求数（Request per second）为 3318.43，平均响应时间（Time per request）为 0.301ms。从图-4.18 中可知，login.php 接口测试的并发请求数同样设置为 100，其在 0.343 秒内共完成了 1000 次请求，每秒处理请求数为 2916.17，平均响应时间为 0.343ms。同时，登录接口的测试结果中含有 Total posted，这是为了保证测试的有效性，本文在测试过程中提交了登录参数 accountName 和 password。由

于汽车细粒度识别系统目前来说只是一个中小型系统，对于一个中小型系统而言，每秒处理请求数无论是 3318 还是 2916，都可以很好的满足实际使用条件。

4.10.2 客户端性能测试

客户端与服务端在移动应用架构中的分工不同，因此在性能测试的标准和方法上也不相同。对于服务端来说，人们更在意其承载量与并发性，所以服务端的性能测试主要关注的是压力测试结果（如 4.10.1 节所述）。而对于客户端而言，由于直接面对的是普通用户，人们往往更关心其运行时的稳定性和流畅性，因此在客户端的性能测试中，稳定性测试和内存测试是主要的考察目标^[46]。在本系统中，客户端主要为 iOS App，对于稳定性测试，只需让应用尽可能长时间的运行，并观察应用是否异常闪退即可。而内存测试则可采用 xcode 自带的 instrument 工具来进行。本次测试所用设备为 iphone 4s，内存测试结果如图-4.19 所示。

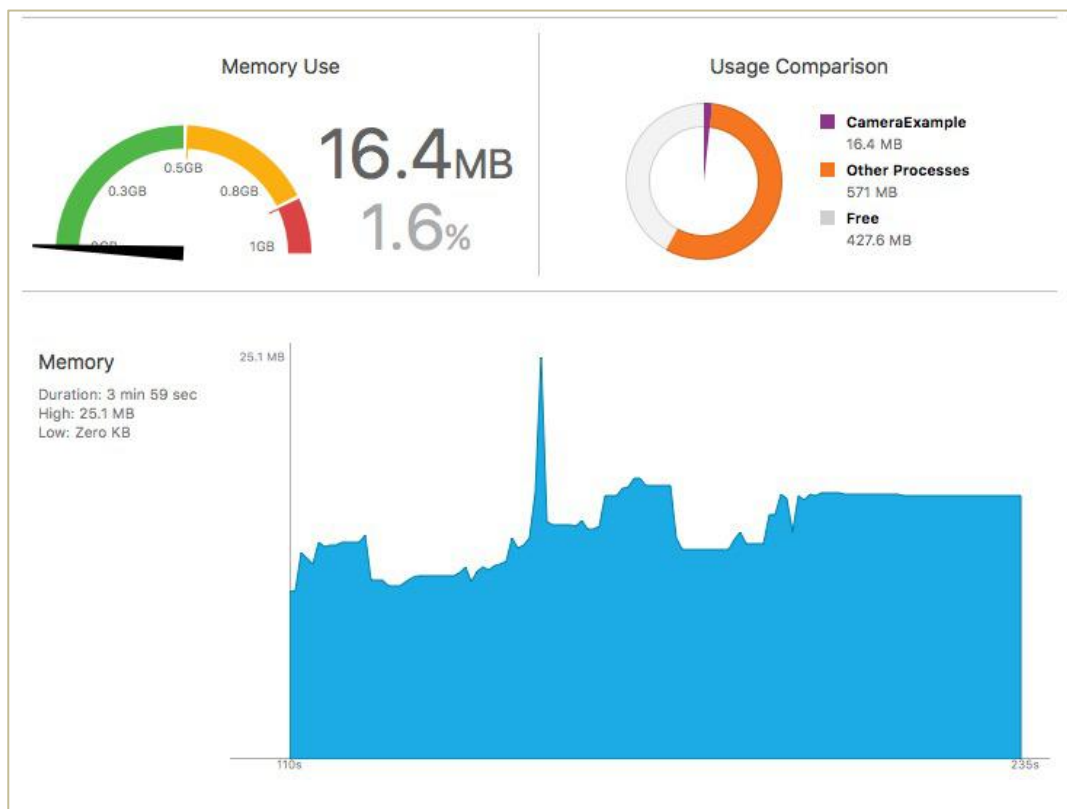


图-4.19：客户端性能测试结果

从图-4.19 中可以看出，iOS App 在运行过程中的内存占用一直维持在 16.4MB 左右，在 iPhone 4s 中的内存占用比为 1.6%，总体来说是比较低的。同时，在长时间地运行过程中未出现内存泄露，界面卡顿和 App 闪退等现象，说明 App 较稳定，满足实际使用的条件。

4.11 本章小结

本章主要介绍了汽车细粒度识别系统的详细设计过程以及功能测试和性能测试结果。其中，系统设计过程主要包括系统需求分析，开发环境介绍，主要功能介绍，系统交互流程，数据库设计以及通信协议设计等部分。同时，在设计过程中，参考第三章实验结论一：**随着数据集中每类图像数量的增多，车型识别模型的总体准确度会相应增加**，设计了“车图上传”功能，主要是用于收集用户上传的车图，丰富系统训练图集，提高后续分类模型的精度。参考实验结论二：**以不同视角的车型图片构建车型数据集，训练得到的模型最终识别准确度不同**，在车型识别模块的相机拍摄界面添加了灰色文字说明，用于提示用户拍摄车型的什么部位可以更容易识别车型。参考实验结论三：**对原始图像采用 Bounding box 技术以去除背景环境的干扰，可显著提高识别准确度**，在车型识别模块的相机拍摄界面添加了一个绿色透明方框，用来提示用户尽量调整相机焦距或距离以将打算识别的车型放置在绿色方框内之后再点击“车型拍照”按钮。功能测试部分则重点展示了汽车细粒度识别模块和系统其余功能模块的界面设计以及实验结果，结果符合预期的设计要求。系统性能测试部分则介绍并展示了汽车细粒度识别系统服务器端的压力测试结果以及客户端 App 的内存测试结果和稳定性测试结果。测试结果均表明，该系统具备实际使用的条件。

5 结论

本章节首先对论文进行了总结，阐述了本文对车型识别领域的主要贡献。其次讨论了汽车细粒度识别系统的主要改进方向，并给出了下一步工作的建议。

5.1 论文的主要贡献

汽车细粒度识别相较于传统的车型粗粒度识别，存在着类内差异小，类间差异大，缺少公共数据集等问题，但对于改善交通管理，建设智能城市具有重要意义，同时也拥有广阔的商业应用前景。本文既在此背景下，将主要研究目的定位于汽车细粒度识别。主要贡献有：

1、受限于无法采集到足够的车型图像数据，直接将深度学习模型应用于汽车细粒度识别领域，非但效果不理想，还使计算成本变得高昂。针对于此，本文提出采用迁移学习与深度学习相结合的方法来构建汽车细粒度识别模型（主要构造方法为：去除 Inception V3 模型的最后一层并重新加上一个全连接层和 softmax 回归分类器）。并经过采集，分类，标记等步骤构建了一个包含“上海大众”旗下所有车型图像的数据集（共分为 27 类，约 30000 张，包含各种汽车视角和背景环境）来充当汽车细粒度识别模型的训练数据集。实验结果表明：采用该方法不但能在较小数据集上实现车型的细粒度识别，同时还能极大地减少模型的训练成本。

2、为提高汽车细粒度识别模型准确率，本文对模型的训练图集及待识别输入图像均采用 bounding box 进行处理（包围盒，即将待识别的物体从目标图像中截取出来）以消除背景环境对车型特征提取的影响。实验结果表明，采用该方法可显著提高汽车细粒度识别模型的准确率。

3、中国汽车市场的蓬勃发展使得汽车细粒度识别技术具备广阔的商业应用前景。另一方面，当前深度学习的实际应用多局限于数据中心或少数公司，并未真正走入普通用户的生活当中。针对这些问题，本文以已训练好的汽车细粒度识别模型为例，通过 tensorflow 框架将其安装至移动设备中，并围绕汽车细粒度识别功能开发

了相关 App 与服务器程序，成功搭建了一个具备多功能的汽车细粒度识别系统，以探索汽车细粒度识别技术及深度学习的商业化及移动化运用。

5.2 进一步工作建议

本论文提出的基于迁移学习+深度学习的模型构造方案给车型细粒度识别研究带来了一些创新思路，并针对传统车型识别系统维护及安装困难，缺少市场应用等问题，设计及开发了一个基于移动端的多功能汽车细粒度识别系统。然而，当前工作仅处于初步阶段，后续仍需要进行深入的研究和拓展。

将来的工作可以考虑在以下方面展开研究：

(1) 本文通过采用 bounding box 技术从而显著提高了车型识别准确率，然而 Bounding box 截取的是车型整体图像，考虑到汽车细粒度识别中，车型之间的差异往往来源于一个或多个局部特征，充分利用更多的富有语义局部可以更好的提高分类结果，未来的工作可重点放在如何获取车型的富有语义局部上。

(2) 汽车细粒度识别系统功能较简单，只能满足用户基本需求，未来可持续扩展该系统功能，如添加车辆购买、分享等功能。同时，由于汽车细粒度识别模型目前体积较大，考虑到手机的内存限制，未来可进一步研究如何在不影响模型整体性能的情况下缩减模型的体积大小。

(3) 由于细粒度视觉领域缺少足够的数据，本文在开发汽车细粒度识别系统时为丰富模型训练图集，提高车型识别准确度，设计了“车图上传”功能以收集用户主动上传的车型图片。但对于用户上传的车图是否标注准确并无保证，错误标注的车图非但无法提高模型的准确率，相反还会使模型的准确度下降。所以下一步工作可更多的放在如何确保用户正确标注并上传车图以丰富车型数据集。

致谢

时间过得很快，转眼间三年的研究生学习生涯即将结束。如今站在这人生的路口回望过去，有辛酸，有喜悦，成功过，也失败过。但总体而言，研究生这三年还是让我学到了许多知识，结识了许多同学，增添了许多阅历。这都是我人生的财富，在论文的最后，在即将分别的时刻，我想向这三年里帮助过我成长的老师和同学表达由衷的感谢。

首先，我要感谢的是我的导师袁巍。袁老师治学严谨，学术渊博，为人却很谦逊。虽然我没有出众的才能，却庆幸得到了袁老师的肯定和接纳，为我打开了通往知识殿堂的大门。这三年里，袁老师时刻关心着我的学习和生活，在科研之路上给予了我许多建议和忠告，对我的学术成长有很大的帮助。

其次，我要感谢我所在实验室的所有同学。能够在同一实验室进行科研和学习是一种缘分。难忘那些为完成项目而日夜坚守的日子，虽然在团队合作中总会有分歧，但大家还能和谐相处，遇事相互分担，这种情谊让人感动。也感谢在这段时间里，他们对我的包容，理解和帮助。能有幸和他们一起拼搏，共同成长，我觉得非常快乐。

最后，我要感谢实验室的尤新革老师，曹汉强老师，徐端全老师，陈建文老师。虽然不是我的直接导师，与我研究方向也不一样，但每当我在学术上有疑问，他们总能热情详尽的为我解答疑惑，无私给予我帮助。我还要感谢我的父母，虽然他们无法在学术上给予我任何帮助，但他们却时刻牵挂，关心着我的生活，是我心中永远的精神港湾。

三年的时间也许只是沧海一瞬，但却永远记录在我的人生历程中。最后，再次对这三年里所有帮助过我的同学，老师表示由衷感谢！

参考文献

- [1] Yang L, Luo P, Chen C L, et al. A large-scale car dataset for fine-grained categorization and verification[J]. 2015:3973-3981.
- [2] Maji S, Rahtu E, Kannala J, et al. Fine-Grained Visual Classification of Aircraft[J]. HAL - INRIA, 2013.
- [3] Szegedy C, Vanhoucke V, Ioffe S, et al. Rethinking the Inception Architecture for Computer Vision[C]// Computer Vision and Pattern Recognition. 2016:2818-2826.
- [4] A. Krizhevsky, I. Sutskever, and G. E. Hinton..Imagenet classification with deep convolutional neural networks.In NIPS,pages 1097-1105,2012.
- [5] Cui Y, Zhou F, Lin Y, et al. Fine-Grained Categorization and Dataset Bootstrapping Using Deep Metric Learning with Humans in the Loop[C]// IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 2016:1153-1162.
- [6] Zhuang F Z, Luo P, He Q, et al. Survey on transfer learning research[J]. Journal of Software, 2015, 26:26-39.
- [7] Zhang Z, Tan T, Huang K, et al. Three-Dimensional Deformable-Model-Based Localization and Recognition of Road Vehicles[J]. IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society, 2012,21(1):1-13.
- [8] Hsiao E, Sinha S N, Ramnath K, et al. Car make and model recognition using 3D curve alignment[J]. 2014.
- [9] Lin Y L, Morariu V I, Hsu W, et al. Jointly Optimizing 3D Model Fitting and Fine-Grained Classification[M]// Computer Vision – ECCV 2014. Springer International Publishing, 2014.
- [10] 李澄非, 陈新华. 融合局部二值模式和 Hu 矩特征的车辆识别[J]. 激光与光电子学进展, 2016(10):200-205.

- [11]张红兵, 李海林, 黄晓婷等. 基于车前脸 HOG 特征的车型识别方法研究与实现[J]. 计算机仿真, 2015, 32(12):119-123.
- [12]王小龙. 基于视频图像的车型识别算法研究与实现[D]. 西安电子科技大学, 2014.
- [13]邓柳. 基于深度卷积神经网络的车型识别[D]. 西南交通大学, 2015.
- [14]Masci J, Meier U, Dan C, et al. Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction[J]. 2011.
- [15]Christian Szegedy, Wei Liu, Yangqing Jia, et al. Going deeper with convolutions.arXiv:1409.4842v1, 2014.
- [16]Zhang H, Xu T, Elhoseiny M, et al. SPDA-CNN: Unifying Semantic Part Detection and Abstraction for Fine-Grained Recognition[C]// IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2016:1143-1152.
- [17]Zhou F, Lin Y. Fine-grained Image Classification by Exploring Bipartite-Graph Labels:, US20160307072[P]. 2016.
- [18]Zhang X, Zhou F, Lin Y, et al. Embedding Label Structures for Fine-Grained Feature Representation[C]// IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 2015:1114-1123.
- [19]Cui Y, Zhou F, Lin Y, et al. Fine-Grained Categorization and Dataset Bootstrapping Using Deep Metric Learning with Humans in the Loop[C]// IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 2015:1153-1162.
- [20]Chen D, Cao X, Wang L, et al. Bayesian face revisited: a joint formulation[C]// European Conference on Computer Vision. 2012:566-579.
- [21]吴志伟. 基于深度卷积神经网络的车型识别方法[J/OL].机电工程技术, 2016 (Z2).<http://www.cnki.net/kcms/detail/44.1522.TH.20160808.1700.300.html>.
- [22]Smith L N, Topin N. Deep Convolutional Neural Network Design Patterns[J]. 2016.

- [23] Larsson G, Maire M, Shakhnarovich G. FractalNet: Ultra-Deep Neural Networks without Residuals[J]. 2016.
- [24] Ioannou Y, Robertson D, Zikic D, et al. Decision Forests, Convolutional Networks and the Models in-Between[J]. 2015.
- [25] Springenberg J T, Dosovitskiy A, Brox T, et al. Striving for Simplicity: The All Convolutional Net[J]. Eprint Arxiv, 2015.
- [26] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift[J]. Computer Science, 2015.
- [27] Salimans T, Kingma D P. Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks[J]. 2016.
- [28] Pan S J, Yang Q. A Survey on Transfer Learning[J]. IEEE Transactions on Knowledge & Data Engineering, 2010, 22(10):1345-1359.
- [29] Dai W, Yang Q, Xue G R, et al. Boosting for transfer learning[C]// International Conference on Machine Learning. ACM, 2007:193-200.
- [30] Argyriou A, Evgeniou T, Pontil M. Convex multi-task feature learning[J]. Machine Learning, 2008, 73(3):243-272.
- [31] Lawrence N D, Platt J C. Learning to learn with the informative vector machine[C]// International Conference on Machine Learning. ACM, 2004:65.
- [32] Lee H, Battle A, Raina R, et al. Efficient sparse coding algorithms[J]. Advances in neural information processing systems, 2007, 19: 801.
- [33] Mihalkova L, Huynh T, Mooney R J. Mapping and revising Markov logic networks for transfer learning[C]// National Conference on Artificial Intelligence. AAAI Press, 2007:608-614.
- [34] Russakovsky O, Deng J, Su H, et al. ImageNet Large Scale Visual Recognition Challenge[J]. International Journal of Computer Vision, 2015, 115(3):211-252.

- [35] Shimaoka S, Stenetorp P, Inui K, et al. Neural Architectures for Fine-grained Entity Type Classification[J]. 2016.
- [36] Wang Y, Choi J, Morariu V I, et al. Mining Discriminative Triplets of Patches for Fine-Grained Classification[J]. 2016:1163-1172.
- [37] Qian Q, Jin R, Zhu S, et al. Fine-Grained Visual Categorization via Multi-stage Metric Learning[J]. Computer Science, 2014:3716-3724.
- [38] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science, 2014.
- [39] Wang J, Song Y, Leung T, et al. Learning Fine-Grained Image Similarity with Deep Ranking[C]// IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2014:1386-1393.
- [40] Yu F, Seff A, Zhang Y, et al. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop[J]. Computer Science, 2015.
- [41] Zhang N, Donahue J, Girshick R, et al. Part-Based R-CNNs for Fine-Grained Category Detection[M]// Computer Vision – ECCV 2014. Springer International Publishing, 2014:834-849.
- [42] Angelova A, Zhu S, Lin Y. Image segmentation for large-scale subcategory flower recognition[C]// Applications of Computer Vision. IEEE, 2013:39-45.
- [43] Krause J, Jin H, Yang J, et al. Fine-grained recognition without part annotations[J]. 2015:5546-5555.
- [44] CS231n.Convolutional Neural Networks for Visual Recognition
[EB/OL].<http://cs231n.github.io/linear-classify/>
- [45] 李亚丽, 王敏, 李静. 迁移学习的研究现状[J]. 时代教育, 2014(9):222-222.
- [46] 王立峰. 基于 Android 平台的校园传媒一体化平台设计[D]. 湖南大学, 2015.