**基于YOLOv2目标检测器的行人检测 赵惠 指导教师 郑艳 厦门大学**

2029320381452802095400

**本 科 毕 业 论 文 （设 计）**

**（主修专业）**

**基于YOLOv2目标检测器的行人检测**

**Pedestrian Detection Based on YOLOv2 Object Detection**

姓 名：赵惠

学 号：24320132202527

学　　 院：软件学院

专 业：软件工程

年 级：2013级

校内指导教师： 郑 艳 讲师

校外指导教师： 周励琨 研发部经理

**二〇一七 年 五 月**

**厦门大学本科学位论文诚信承诺书**

本人呈交的学位论文是在导师指导下独立完成的研究成果。本人在论文写作中参考其他个人或集体已经发表的研究成果，均在文中以适当方式明确标明，并符合相关法律规范及《厦门大学本科毕业论文（设计）规范》。

该学位论文为（ ）课题（组）的研究成果，获得（ ）课题（组）经费或实验室的资助，在（ ）实验室完成（请在以上括号内填写课题或课题组负责人或实验室名称，未有此项声明内容的，可以不作特别声明）。

另外，本人承诺辅修专业毕业论文（设计）（如有）的内容与主修专业不存在相同与相近情况。

学生声明（签名）：

年 月 日

致 谢

凤凰花开的季节，到了属于我的毕业季。四年的大学时光，转瞬即逝，留下的是一个心底永远的回忆。大一加入体育部，和体育部小伙伴们团结协作组织活动，一起作为西边烤场的常客；大二宿舍，教室，饭堂三点一线，潜心学习，有了满意的成绩；大三加入贝壳工作室，参与厦大迎新系统的开发，体会肩上的那份责任；大四广州实习，感受深度学习的魅力。这四年，紧张但充实，收获友谊，收获成长，少一分稚气，多一分稳重。感谢厦大，临别之际，向您表达最崇高的敬意！

值此论文完成之际，谨向所有关心和支持我的人们致以诚挚的谢意！

首先，我要感谢我的导师郑艳讲师。在毕业设计阶段给予我指导性的意见和建议，督促我认真完成毕业论文，对我的论文进行批阅。在此，谨向郑艳老师致以最诚挚的感谢！

其次，我要感谢广州市维安科技股份有限公司和研发部的周励琨经理，给予我实习的机会，初次涉猎深度学习，提供毕业设计用的硬件设备。同时感谢陈海玉同事在实习期间给我提供了住所，感谢王喆同事在实习期间给予我电脑硬件设备方面的帮助，感谢郭倩同时在实习期间帮助我学习深度学习相关知识。

再者，我要感谢厦门大学软件学院全体老师的辛勤栽培。特别感谢辅导员李绍玉老师，教学秘书林伟强老师，教授我计算机网络课程的林坤辉老师，和助教王哲远老师。我要感谢我的舍友，体育部的成员，贝壳工作室的成员，大学四年，一起成长，感谢你们的陪伴。

最后，我要感谢我的家人，感谢一直以来的鼓励与支持。在未来的日子里，我会更加努力学习和工作，不辜负你们的期望。

**摘 要**

行人检测作为自动驾驶、智能监控、智能机器人等许多应用的底层基础，近年来，已经引起了超过一般目标检测的特别关注。随着卷积神经网络（Convolutional Neural Networks, CNNs）的在计算机视觉领域的不断发展，目标检测领域在较高精度下，已经可以达到实时的检测性能。比如，以回归的方法处理检测问题的YOLOv2[1]，是先进的、实时的目标检测器。在一个Titan X上每秒可检测40帧分辨率为544\*544的图片。行人一般具有比较小的尺度，行人各种姿势和不同遮挡水平对行人检测来说都是巨大的挑战。为了保证精度，代表行人检测先进技术的几个行人检测器，RPN+BF[2]，SAF R-CNN[3]，CompACT-Deep[4]检测一帧图片的时间约为0.5秒。目前最精确的行人检测器F-DNN[5]检测一帧图片的时间也需要0.11秒，约每秒检测9帧，这还远远达不到监控视频每秒至少检测25帧的实时性要求。

本论文旨在修改YOLOv2目标检测器，将他专门用于检测行人，在保证检测速度的同时，提高行人检测的精度。

**关键词：**行人检测；卷积神经网络；YOLOv2目标检测器

**Abstract**

Pedestrian detection, as a basis foundation of real-world applications such as automatic driving, intelligent surveillance and intelligent robot, has attracted special attention beyond general object detection. With the continuous development of convolution neural network in the field of computer vision, object detection has been able to achieve real-time detection performance with high accuracy. For example, YOLOv2[1] detector, which solve detection problem as regression problem, is a state-of-the-art, real-time object detection system. On a Titan X it processes 544\*544 images at 40 frames per second (FPS). Small scales, difference postures and occlusion levels are huge challenges for pedestrain detection. In order to ensure accuracy, the state-of -the-art pedestrain detectors, such as RPN+BF[2], SAF R-CNN[3] and CompACT-Deep[4] detector, detect an image in about 0.5s. The most accuracy pedestrain detector F-DNN[5], need 0.11s for detecting an image, is about 9 FPS, which is far less than the monitors video to detect at least 25 FPS real-time requirements.

This paper aims to modify the YOLOv2 object detector, which is specially designed for pedestrian detection, and can improve the accuracy of pedestrian detection while ensuring the detection speed.

**Key words：**Pedestrian Detection; Convolution Neural Networks; YOLOv2 Object Detector

**目 录**

[第一章 绪论 1](#_Toc480989251)

[1.1 引言 1](#_Toc480989252)

[1.2 行人检测研究现状 2](#_Toc480989253)

[1.3 研究内容 4](#_Toc480989254)

[1.4 论文组织结构 4](#_Toc480989255)

[第二章 相关知识概述 5](#_Toc480989256)

[2.1 人工神经网络 5](#_Toc480989257)

[2.1.1 产生背景 5](#_Toc480989258)

[2.1.2 人工神经网络结构 6](#_Toc480989259)

[2.1.3 人工神经网络使用方法 8](#_Toc480989260)

[2.2 卷积神经网络 10](#_Toc480989261)

[2.2.1 基础介绍 10](#_Toc480989262)

[2.2.2 网络结构 10](#_Toc480989263)

[2.2.3 主要特点 13](#_Toc480989264)

[2.3 Quartz作业调度框架 15](#_Toc480989265)

[2.4 HTTPS 16](#_Toc480989266)

[2.5 GZIP 19](#_Toc480989267)

[2.6 本章小结 23](#_Toc480989268)

[第三章 系统设计与实现 24](#_Toc480989269)

[3.1 系统架构 24](#_Toc480989270)

[3.2 业务流程 25](#_Toc480989271)

[3.3 系统功能模块 27](#_Toc480989272)

[**3.3.1 前端数据录入** 27](#_Toc480989273)

[**3.3.2 后台数据处理** 32](#_Toc480989274)

[**3.3.3 休息室旅客管理** 35](#_Toc480989275)

[**3.3.4 统计报表** 37](#_Toc480989276)

[3.4 网页优化 39](#_Toc480989277)

[3.4.1 **HTTPS** 39](#_Toc480989278)

[3.4.2 **GZIP压缩** 41](#_Toc480989279)

[3.5 本章小结 42](#_Toc480989280)

[第四章 系统测试与运行结果 43](#_Toc480989281)

[4.1 测试环境 43](#_Toc480989282)

[4.2 部署过程 44](#_Toc480989283)

[4.3 运行结果 45](#_Toc480989284)

[4.3.1 旅客管理模块 45](#_Toc480989285)

[4.3.2 休息室管理模块 48](#_Toc480989286)

[4.3.3 统计报表模块 50](#_Toc480989287)

[4.4 本章小结 52](#_Toc480989288)

[第五章 总结与展望 53](#_Toc480989289)

[5.1 论文总结 53](#_Toc480989290)

[5.2 工作展望 54](#_Toc480989291)

[参考文献 55](#_Toc480989292)

# 第一章 绪论

## 引言

目标检测的任务，是在图片或视频中检测出目标，标明目标的类别和框出目标的位置，目标可以是多个类别。行人检测是一类特殊的目标检测，只在图片或视频中检测出行人，框出行人的位置，近年来受到了超越一般目标检测外的特别关注。行人检测是许多应用的重要组件。

* 1. 车辆辅助驾驶与自动驾驶

随着科技的发展，人们生活水平不断提高，越来越多的人都有了自己的代步工具-汽车。随着车辆的增多，交通事故也成为了人们时刻需要警惕的风险。据世界卫生组织统计，2013年世界各地道路交通死亡人数为125万人，而且所有道路交通死亡中半数最无保护的人，即骑摩托车者、行人和骑自行车者。一名成年行人如果被时速低于50公里/小时的汽车碰撞，死亡几率在20%以下，但如果被时速为80公里/小时的汽车碰撞，则死亡风险几乎为60%。这说明控制车速可以有效降低交通事故的死亡风险。如果在车辆行驶的过程中实时检测出处于危险区域的行人，使司机有足够的时间采取措施，或是车辆自动采取措施，就能降低发生交通事故风险或者死亡风险，所以行人检测是车辆辅助驾驶与自动驾驶的关键组件之一。目前的车辆辅助驾驶和无人驾驶是学术界和工业界共同关注的热点，但目前行人检测的精度和速度都还有待提高。

* 1. 智能监控

在安防领域，目前的监控大多还是由人工完成，这需要大量的人员，工作枯燥但又必须时刻保持警惕，稍有不慎，带来的损失不容小觑。设想一下，如果在监狱、金库、珠宝店等在特定时间避免行人出入的地方，计算机能实时获取监控视频，精确地检测出行人，记录行人出现的时间。这样就可以节约监控的人力成本，提高安防水平。一旦发生盗窃案件，越狱事件等，也能快速定位到发生的时间，有助于快速破案。

* 1. 视频结构化

视频结构化是当前的一个发展趋势，利用行人检测找出视频中的人，进一步对行人的肤色，着装，性别等进行判断，构造新型的结构化数据，存储到数据库里，方便事后取证。当然还可以结合机动车检测等，存储别的视频结构化信息，提供多种多样的服务。

* 1. 智能机器人

智能机器人也是近年来最热门的研究方向，大部分智能机器人的主要服务对象是人，所以检测出人是它的基本技能，在此基础上才能更好地为人服务。

行人检测的应用场景不局限于上述的几种，随着时间推移和科技发展，会有更多各种各样的需求，因此行人检测近年来受到了广泛的关注。

## 行人检测研究现状

目前目标检测有两个基本的框架：R-CNN系列的基于候选区域（Region Proposals）的目标检测框架，这种框架把检测问题当做分类问题解决，另一个是YOLO系列的目标检测框架，这种框架则把检测问题当做回归问题来解决。大多数的目标检测和行人检测研究都是基于把检测问题当做分类问题。

传统的目标检测利用穷举的策略进行区域选择，采用滑动窗口，设置不同的窗口大小和不同的长宽比对图像进行遍历，然后提取一些手工设计的特征如SIFT、HOG等，再给SVM、Adaboost等分类器进行分类。基于滑动窗口的区域选择方式选择出来的窗口太多且没有针对性，手工设计的特征对于多样性的变化不够健壮。于是出现了基于候选区域的一系列深度学习目标检测法。

文献[7]提出了候选区域的概念，先从图像中找出可能包含目标的位置，可以用Selective Search、Edge Boxes等方法实现。然后对每个候选区域利用卷积神经网络提取出关键特征，输入到SVM进行分类。这种方法称为R-CNN。候选区域的方式比滑动窗口的方式可以明显减少提取窗口的数量，而且保持较高的召回率。但是R-CNN检测速度太慢，假设对图像提取了2000个候选区域，之后每个候选区域就相当于是一张图像一样进行卷积网络提取特征和分类，那么检测实际上相当于对1张图像进行了2000次提取特征和分类的过程。这个问题在文献[8]中解决了。

文献[8]提出了SPP-NET网络，利用空间金字塔池化层（Spatial Pyramid Pooling Layer），对每个候选区域使用不同大小的金字塔映射，使不同尺寸的候选区域在进入全连接层时可以统一到固定的长度（卷积层的输入不限制输入图像尺寸，但全连接层限制图像的尺寸），从而使得一张图像只用提取一次卷积层特征，然后将候选区域在原图上的位置映射到卷积特征图上。SPP-NET因此大大加快了检测速度，但是R-CNN和SPP-NET还有一个共同的问题，训练分为多个阶段，要微调网络，然后训练分类器，再训练边框回归器。这个问题在文献[9]中进行了改善。

文献[9]提出了Fast R-CNN网络，它用感兴趣区域池化层（Region-of-Interest Pooling Layer）代替了SPP-NET中的金字塔映射，感兴趣区域池化层只需要对每个候选区域下采样到一个7\*7的特征图即可。Fast R-CNN用Softmax代替SVM分类，利用多任务损失函数将边框回归加入到了网络中，使得不考虑候选区域选取的过程，那么整个检测器的训练过程是端到端的。但是候选区域选取的过程本身就是一个耗时的过程，文献[10]对这部分做了改进。

文献[10]提出了Faster R-CNN网络，它采用了RPN（Region Proposal Network）进行候选区域的提取，再将提取到的候选区域给Fast R-CNN进行分类。RPN使用卷积神经网络直接产生候选区域，只在最后的卷积层上使用滑动窗口，用Anchor机制和边框回归得到多尺度多长宽比的候选区域。使用RPN可以得到更少而且质量更高的候选区域。Faster R-CNN虽然在目标检测上取得了不错的成果，但对于行人检测来说，效果并不好。

中山大学的张立亮等人对Faster R-CNN为什么不能很好的检测行人进行研究，并提出了RPN+BF的方式进行行人检测[2]。他们认为Faster R-CNN只在RPN最后一层卷积层提取特征，特征图的分辨率不足以检测尺度比较小的行人，而且也缺少hard negative example进行训练。于是提出RPN+BF的方法，同样使用RPN提取候选区域，但是在更底层但是分辨率比较高的卷积特征图上提取特征，在多几层上提取特征并简单的连接起来。把得到的特征给下游的级联增强森林（Cascaded Boosted Forest）进行分类。该方法在多个行人检测基准数据集上取得了不错的效果，在Caltech数据集合理行人的评估设置上排名第二，检测时间约每秒两帧。

在Caltech数据集多个评估设置上排名第一的是融合深神经网络（FDNN）[5]。该方法利用SSD[11]生成行人的候选区域，SSD是把检测当做回归问题来做的一个目标检测算法，得到的候选区域尽可能包含全部的行人。然后利用基于软拒绝的网络融合方法（soft-rejection based network fusion method），融合了多个深层神经网络分类器形成一个分类网络，减少SSD产生的大量的假阳例，给出最终的分类结果。他们还进一步提出了利用有语义分割作用的上下文聚集扩展卷积网络作为另一个分类器集成到网络的融合架构里，以提高检测精度。他们的方法检测时间约每秒十帧，这个速度相比于检测精度排名靠前的几个行人检测算法来说是最快的了。

## 研究内容

YOLOv2是目前先进的实时目标检测系统之一，是YOLO[12]目标检测器的改进版本。它把目标检测问题当作回归问题来解决，直接在原图上进行目标的检测与定位，在保持精度的情况下，极大地提高了检测速度。具体的，在Titan X上处理图像的速度在40-90 FPS，在VOC 2007数据集上mAP（Mean Average Precision）为78.6%，在COCO test-dev数据集上mAP为44.0%。

本文旨在修改YOLOv2目标检测器，将它用于行人检测，经过实验和分析提高YOLOv2检测的精度。

## 论文组织结构

本文一共包括五章，论文的具体内容安排如下：

第一章 绪论。本章主要阐述行人检测的研究现状，本文的研究内容及研究意义。

第二章 相关知识概述。本章主要介绍本篇论文开展的基础，包括神经网络，卷积神经网络，YOLOv2目标检测器以及行人检测权威的几个数据库

第三章 实验准备。本章主要介绍将YOLOv2目标检测器改成行人检测器的几个修改方向，进行实验的环境，以及前期的数据准备和评估设置。

第四章 实验和结果分析。本章主要介绍具体的实验设置和展示实验结果，并进行分析。

第五章 总结与展望。本章主要对全文进行总结以及对以后工作的展望。

# 第二章 相关知识概述

## 人工神经网络

### 产生背景

人工神经网络是机器学习的一个新的领域。传统的机器学习大致过程如图2-1所示：

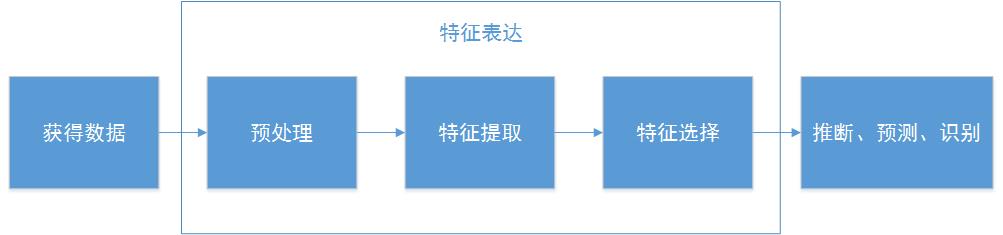


图 2-1 传统机器学习过程图

从开始的通过传感器来获得数据，然后经过预处理、特征提取、特征选择，再到推理、预测或识别，传统的机器学习主要在最后一个阶段，即推理、预测或识别做了很多的研究，比如线性回归、SVM等。而中间三个部分，合起来就是特征表达。良好的特征表达，对最终算法的准确性起到了非常关键的作用，而且系统主要的计算和测试工作都耗在这一大部分。而这一部分实际一般是由人工完成的，依靠人工来提取特征。虽然也有一些人工设计的效果不错的特征，比如SIFT，HoG特征。但是由人工设计特征，不仅需要专业的知识，而且设计出来的特征也难以具有通用性，不够鲁棒。于是我们希望算法能够自动提取好的特征。人工神经网络就是一种可以自动提取特征的机器学习算法，它的灵感来源于人脑的视觉机理。

1981年的诺贝尔医学奖，颁发给了David Hubel、Torsten Wiesel和Roger Sperry。前两位的主要贡献是发现了视觉系统的信息处理：可视皮层是分级的。他们利用猫做实验来研究瞳孔区域与大脑皮层神经元的对应关系。在牺牲了若干只小猫后，他们发现了一种称为“方向选择性细胞”的神经元细胞。当瞳孔发现了眼前的物体边缘，而且这个边缘指向某个方向时，这种神经元活跃。这个发现激发了人们对于神经系统的进一步思考，神经-中枢-大脑的工作过程，或许是一个不断迭代、不断抽象的过程，从原始信号，做低级抽象，逐渐向高级抽象迭代。如图2-2所示：

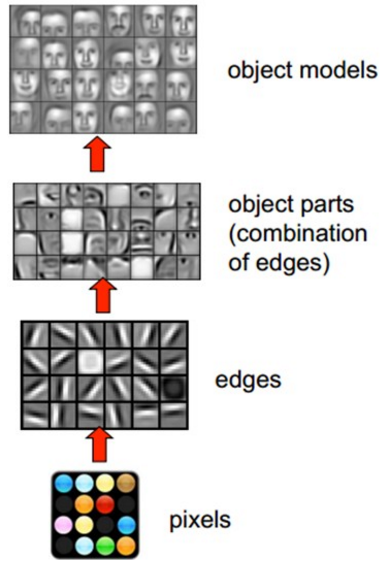


图 2-2 大脑视觉信息处理过程[13]

从原始信号摄入开始，瞳孔摄入像素，接着做初步处理，大脑皮层某些细胞发现边缘和方向，然后抽象，大脑判定眼前物体的形状是耳朵，然后进一步抽象，判定该物体是人脸。人工神经网络就借鉴了这个过程。机器学习实际上就是在找从给定输入到目标输出的一个映射函数，人工神经网络其实就是用一个层次型的结构表示出了一个复杂的映射函数。

### 人工神经网络结构

神经元是神经网络的基本单元，它的设计灵感来自于生物学上神经元的信息传播机制。神经元有两种状态，兴奋和抑制。一般情况下，大多数神经元处于抑制状态，一旦某个神经元受到刺激，导致它的电位超过一个阈值，这个神经元会被激活，处于“兴奋”状态，进而向其他神经元传播化学物质，其实就是信息[14]。图2-3为生物神经元：

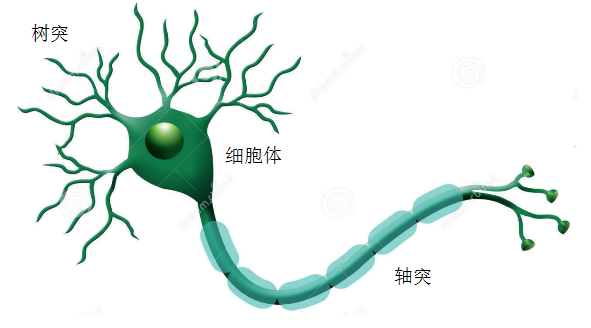
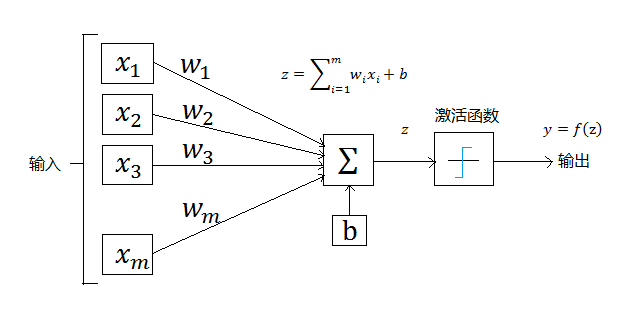


图 2-3 生物神经元[14]

1943年，McCulloch和Pittss将上图的神经元结构用一种简单的模型进行了表示，构成了一种人工神经元模型，也就是我们现在经常用到的“M-P神经元模型”，如图2-4表示：

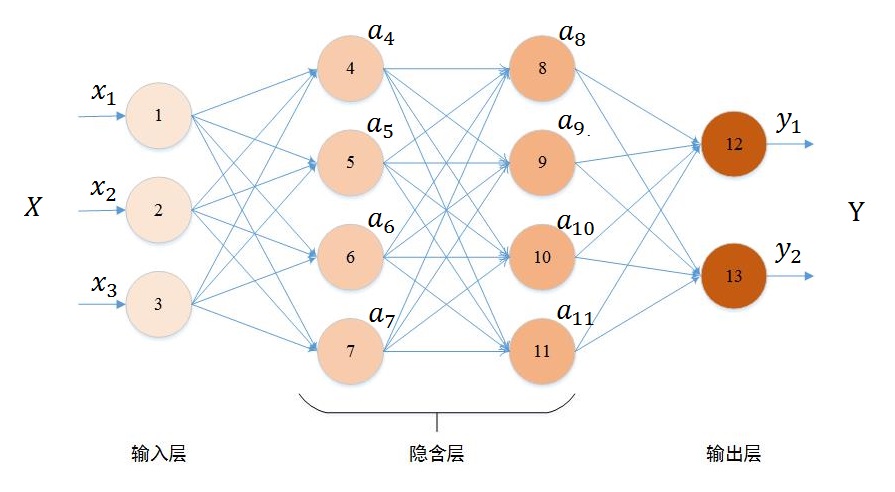


**图 2-4 人工神经元模型**

图中x1,x2,…xm是来自其他神经元的刺激，也就是这个神经元的输入，w1,w2,…,wm是权重，b是偏置。这个神经元的树突部分对接收到的刺激做个汇总，把不同的刺激乘以他们的权重再加起来。因为不同的神经元对收到的不同的刺激反应可能不同，所以需要乘以相应的权重。比如一个神经元对颜色比较敏感，那么有关颜色刺激的权重就大一点。有了总刺激后，要看看这个总刺激有没达到使这个神经元兴奋的阈值，所以要加个偏置b，最后超过阈值的刺激就为z。然后神经元要决定怎么处理这个刺激，所以刺激z要经过一个激活函数f，这个激活函数决定这个神经元的输出。最后这个神经元的输出y如下：

|  |  |  |
| --- | --- | --- |
|  |  | （公式2-1） |

神经元不同的连接就组成了不同的神经网络结构，权值和偏置就是网络的参数。图2-5是全连接神经网络，是神经网络的一种：



**图 2-5 全连接网络示意图**

全连接网络就是指上层神经元和下层神经元全连接的神经网络。神经网络一般包含输入层，隐含层和输出层，可有多个隐含层。说一个神经网络含有几层，一般是指除了输入层之外所有层的层数。

### 人工神经网络使用方法

统计学习方法，是利用大量的数据，在数据中发现规律和模式。这种数据驱动的方法分为如下三个步骤：

1、准备用于训练和测试的数据集

2、利用机器学习方法训练模型

第2步又分为以下三步：

1. 定义模型的假设空间
2. 定义模型选择的准则
3. 定义模型选择的算法

3、用测试集来评估分类器的好坏

神经网络模型是机器学习模型的一种，所以利用神经网络模型解决问题，也就是上述的三个步骤。

首先要定义模型的假设空间。假设空间其实就是一组函数（也称为一组假设，得分函数），我们要定义这组函数，使他包含最优的函数。如果函数的参数确定，就是一个函数；如果函数的参数不确定，就是一组函数。人工神经网络其实就是以分层的形式表示的一个映射函数，他的参数就是网络的权值和偏置。神经元的连接方式确定了网络结构，网络结构就决定了网络的权值和偏置个数。也就确定了模型的假设空间。最后要从假设空间中选择最好的假设，其实就是确定网络参数的过程。

然后要定义模型选择的准则，也就是希望模型达到什么样的效果，即定义目标函数。比如手写字体识别，给定一张手写数字的图片，我们希望网络输出的结果与该图片真实表示的数字一致。所以我们训练的目标就是使得网络的输出与真实值越接近越好，这样我们就可以定义目标函数（也成称为损失函数）。这样模型的选择问题就变成了最优化问题，如何选择一组参数使得损失函数越小越好。

最后，要确定模型选择的算法，即如何得到最优的参数组。神经网络常用的优化方法是反向传播算法（Backpropagation algorithm, 简称BP算法）。BP算法的主要思想如下：

* 1. 将训练集数据输入到人工神经网络输入层，经过隐含层，到达输出层，并输出结果，是前向传播过程。
  2. 由于输出结果与真实值有误差，则计算输出结果与真实值之间的误差，并将该误差从输出层向隐含层反向传播，直至传播到输入层，这是反向传播过程。
  3. 在反向传播过程中，根据误差调整各个参数的值
  4. 不断迭代上述过程，直至收敛。

所以利用神经网络模型解决问题，通常是需要根据实际问题来构造出网络结构，然后确定损失函数和优化方法，然后通过训练样本和优化方法来迭代找到最优参数组。

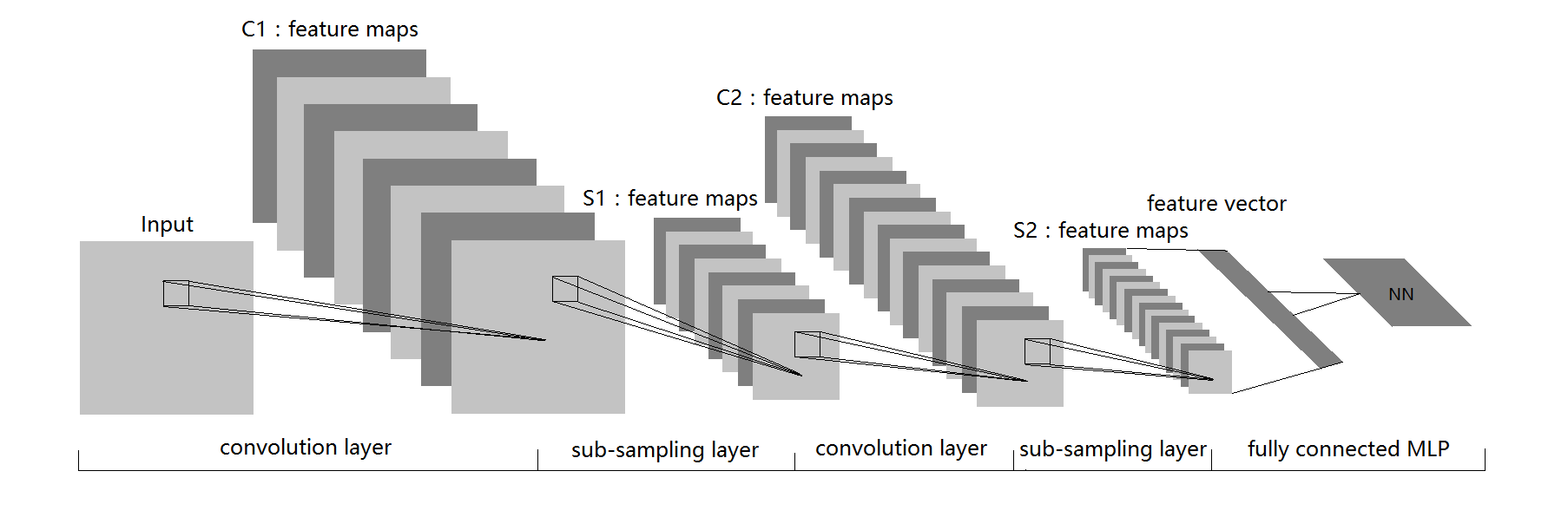
## 卷积神经网络

### 2.2.1 基础介绍

卷积神经网络是人工神经网络的一种，其网络结构更接近于生物神经网络，主要用于图像处理和语音识别领域。它在人工神经网络的基础上，引入卷积运算来自动提取特征，这样得到的特征对于平移、尺度缩放、旋转、扭曲等其他形变有着高度的不变性，通用性也更好。卷积神经网络还利用空间位置关系，采用稀疏连接和权值共享的方式，降低网络模型的复杂度，减少需要训练的参数数量，提高反向传播训练效率。

### 2.2.2 网络结构

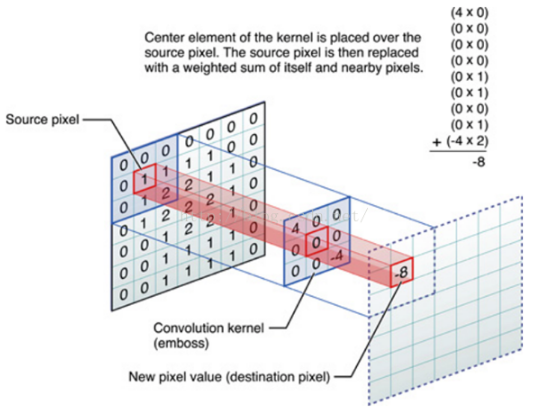
卷积神经网络模型的种类很多，但基本结构大致相同，如图2-6所示。



**图 2-6 卷积神经网络示意图**

卷积神经网络是一种多层的神经网络结构，每层包含多个二维平面，每个平面包含多个独立的神经元。在卷积神经网络中，这种二维平面称为特征地图（Feature Map），图中的C-层表示卷积层（Convolution Layer），S-层表示下采样层（Sub-sampling Layer）。典型的卷积神经网络由输入层、卷积层、下采样层、全连接的多层感知机（Fully Connected MLP）构成。经过简单预处理的输入图像作为输入层，常用的预处理操作为图像去均值（将输入数据各个维度都中心化为0，避免数据过多偏差，影响训练效果）和调整图像大小等。输入层后接一系列交替排列的卷积层和下采样层。

卷积层的主要作用是利用卷积操作提取出图片中的特征，它是卷积神经网络中非常重要的层。卷积操作就是对图像（不同数据窗口数据）和滤波矩阵做内积（逐个元素相乘再求和）的操作。滤波矩阵又叫卷积核（Convolution Kernel）或滤波器（Filter），是一组固定的权重。图2-7是卷积操作的示意图。



**图 2-7 卷积操作示意图[15]**

上图左侧为原始输入数据，中间为滤波矩阵，右侧为输出的新的二维数据。中间滤波矩阵与数据窗口做内积，对应位置上数字先相乘再相加，即4\*0 + 0\*0 + 0\*0 + 0\*0 + 0\*1 + 0\*1 + 0\*0 + 0\*1 + (-4)\*2= -8。数据窗口的大小和滤波矩阵的大小是一致的，对应的数据窗口是卷积神经网络中的一个重要概念--感受野（Reception Field）。在卷积神经网络中，滤波矩阵对局部输入数据（感受野中的数据）进行卷积计算，每计算完一个数据窗口内的局部数据后，数据窗口不断平移滑动，直到计算完所有数据。这个过程涉及到如下几个参数：

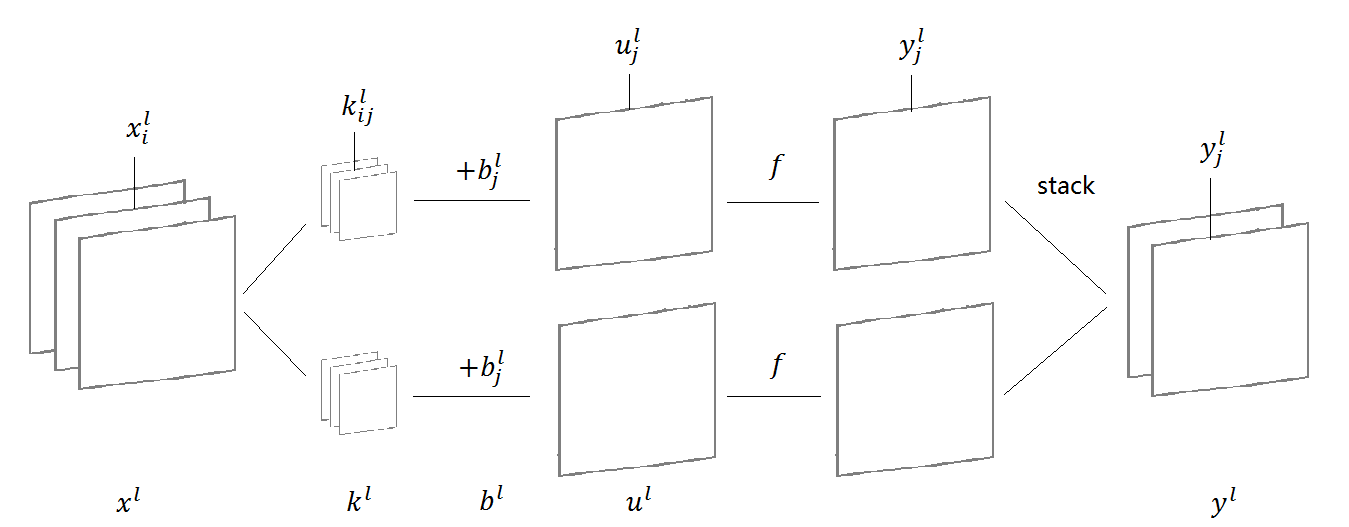
* 1. 深度（Depth）：滤波器的个数，决定输出几层特征地图
  2. 步长（Stride）：决定滑动多少步可以到达边缘
  3. 填充值（Zero-padding）：在输入图像或者特征地图外围边缘填充若干圈0，方便从初始位置以步长为单位可以刚好滑到末尾位置

通常情况下，对卷积操作的输出用一个非线性函数（Non-linear Function）来转换，以使此模型获得非线性特性，并把输出限制在给定的范围内。这个函数也被称为激活函数（Activation Function）。而且，原始输入数据和输出的特征地图也为多个。

卷积层的具体计算公式如下：

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | （公式2-2） |
|  | |  | （公式2-3） |
| 式中 |  | | |
|  | 第l层的第j个特征地图的卷积输出，即图2-7右侧的二维平面 | | |
|  | 第l层的第i个输入特征地图 | | |
|  | 连接第l层额第i个输入特征地图和第j个特征地图的卷积核 | | |
|  | 第l层的第j个特征地图的偏置 | | |
|  | 激活函数 | | |
|  | 第l层的第j个输出特征地图，也是第l+1层的第j个输入特征地图 | | |

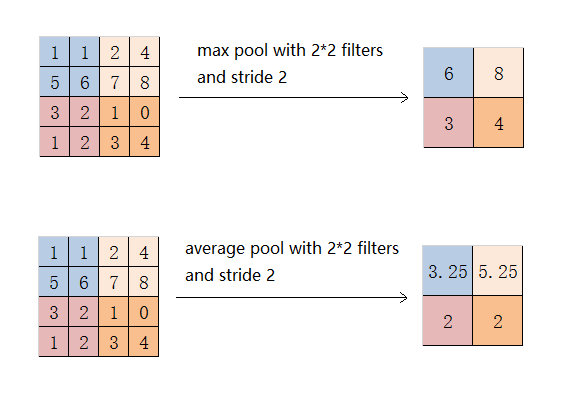
卷积具体计算过程如图2-8：



**图 2-8 卷积具体计算过程示意图**

图中最左侧为第l层特征地图，该层有3个特征地图，i的范围即为[1,3]，每个卷积核的层数应和输入特征地图个数一致，即为3层，这里有两个卷积核，j的范围为[1,2]，第l层特征地图和2个卷积核做卷积运算的结果加上偏置，即得到l层的卷积输出，卷积输出经过激活函数后堆叠起来，即得到集l层输出的j个特征地图。这样得到的特征对图片具有较好的旋转、平移、缩放的不变性。

下采样层的主要作用是对图片进行降维。常用的下采样操作是池化。池化层的输出结果是一个的不重叠矩形区域的最大激活值或者平均激活值，对应叫做最大池化和平均池化。最大池化，就是取矩形区域中的最大值来代表这个区域；平均池化，就是取矩形区域中的平均值来代表这个区域。图2-9为最大池化和平均池化的示意图：



**图 2-9 最大池化与平均池化示意图**

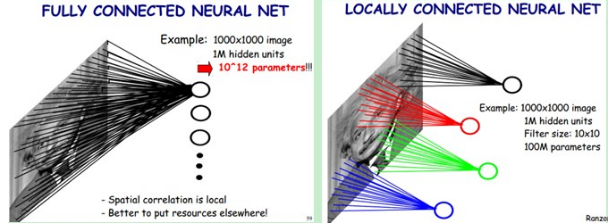
一系列交替的卷积层和下采样层之后，将特征地图光栅化为特征向量，输入全连接的多层感知机进行分类等操作。

### 2.2.3 主要特点

卷积神经网络是新型的人工神经网络，它有几个重要的概念：局部感受野、权值共享和池化，这也是它的主要特点。

1、局部感受野

局部感受野的概念是受生物学中的视觉系统结构启发，视觉皮层的神经元就是局部接收信息的。传统的神经网络，隐含层的每一个神经元都与前一层所有神经元相连，如图2-10左侧。如果前一层是1000\*1000的特征地图，隐含层有一百万个神经元，那之间的连接参数一共有1012个。实际上，图像是有空间位置关系的，这种位置关系是局部的，人的视觉细胞在感受外界图像时，每个神经元一个时刻只接受图像的局部信息。卷积神经网络因此受启发，将原来全连接的方式改成局部连接，如图2-10右侧所示，每个隐含层的神经元，只与前一层的一个10\*10的局部感受野相连。这种连接方式一共需要学习108个参数，比原来全连接方式需要学习的参数少了一万倍。



**图 2-10 全连接与局部连接[16]**

2、权值共享

图像不同区域可能具有相同的特征，比如垂直方向的边缘信息。这意味着在图像不同的感受野区域提取同一种特征的方法是一样的，这就是权值共享概念的来源。卷积网络中提取同一种特征时的每个隐含层神经元与对应的局部感受野区域之间的权值是相同的。需要提取图像中的某一种特征，用固定的卷积核（固定的权值矩阵）在图像上滑动进行卷积计算即可。这样如果卷积核的大小为10\*10，那么学习一种特征只需要100个参数，学习100种特征也只需要一万个参数。局部感受野和权值共享大大减少了需要学习的参数数量，使模型更简单，易于训练。

3、池化

经过卷积操作得到的特征维度还是很大。卷积神经网络利用池化操作，对一小块区域进行聚合统计，利用他们的统计特性（如区域均值或区域最大值）作为整个区域的特征，这样不仅减少了为特征的维度，还能防止过拟合。

## Quartz作业调度框架

Quartz是OpenSymphony开源组织在任务调度领域的一个开源项目，完全基于Java实现。该项目于2009年被Terracotta收购，目前是Terracotta旗下的一个项目。作为一个开源的作业调度框架，它完全由Java写成，并设计使用于J2SE和J2EE应用中，具有很大的灵活性，能够用它来为执行一个作业而创建简单或复杂的作业调度[14]。

JDK中的Timer、ScheduledExecutor是Java中实现定时调度任务的基本方法，但由于其功能简单，较难实现复杂的任务调度运行规则，在实际的企业应用中，更多的选择开源调度框架，如Quartz、Cron4j、JCronTab等，其中Quartz很容易与Spring集成实现灵活可配置的调度功能，使用最为广泛。具有以下三大特点：

1. 强大的调度功能，支持丰富多样的调度方法，可以满足各种常规及特殊需求；
2. 灵活的应用方式，支持任务和调度的多种组合方式，支持调度数据的多种存储方式；
3. 分布式和集群能力，Terracotta 收购后在原来功能基础上作了进一步提升，不仅支持调度的水平集群部署，也支持调度的垂直集群部署。

Quartz任务调度框架的核心元素是Scheduler，Trigger和Job，其中Trigger和Job是任务调度的元数据，Scheduler是实际执行调度的控制器。

在Quartz中，Trigger是用于定义调度时间的元素，负责具体的触发策略，即按照什么时间规则去执行任务。Quartz中主要提供了四种类型的Trigger：SimpleTrigger，CronTirgger，DateIntervalTrigger，和NthIncludedDayTrigger。这四种Trigger可以满足企业应用中的绝大部分需求。

在Quartz中，Job用于表示被调度的任务。主要有两种类型的Job：无状态的（stateless）和有状态的（stateful）。对于同一个trigger 来说，有状态的Job能被并行执行，只有上一次触发的任务被执行完之后，才能触发下一次执行。Job主要有两种属性：volatility 和durability，其中volatility 表示任务是否被持久化到数据库存储，而durability表示在没有Trigger关联的时候任务是否被保留。两者都是在值为true 的时候任务被持久化或保留。一个Job可以被多个Trigger关联，但是一个Trigger只能关联一个Job。

在Quartz中，Scheduler由Scheduler工厂创建：DirectSchedulerFactory或者 StdSchedulerFactory。第二种工厂StdSchedulerFactory使用较多，因为 DirectSchedulerFactory使用起来不够方便，需要作许多详细的手工编码设置。Scheduler 主要有三种：RemoteMBeanScheduler，RemoteScheduler和StdScheduler。具体Quartz核心元素之间的关系如下图所示：

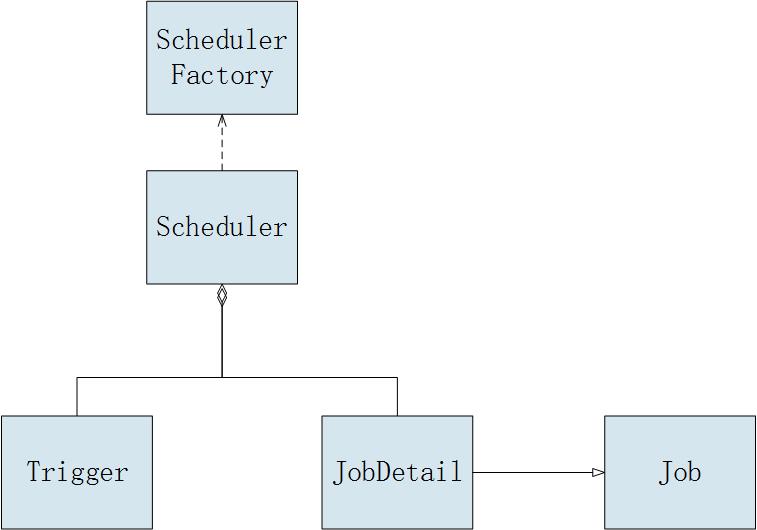


图 2-5 Quartz核心元素关系图

Scheduler任务调度器是Quartz的核心，是调度任务的容器和管理，Job和Trigger都需要注册到Scheduler上。Scheduler负责任务调度器的初始化，进行任务的启动、停止以及任务数据、事件的管理。

在J2EE系统中，Quartz可以和Spring进行集成。利用Spring的事务管理，对Job进行统一的管理，可以很方便的在Job中注入相关业务Bean，并由Spring进行调度容器的初始化、加载全部Job和Trigger并启动运行[15]，有利于在Spring环境下创建对应的组件对象，并结合Spring容器生命周期进行相应的操作。

## HTTPS

* 1. HTTPS介绍

HTTPS（Secure HyperText Transfer Protocol）安全超文本传输协议，是一个安全通信通道，用于在客户端和服务器端之间进行信息交换。传统的Web请求时使用HTTP协议，但安全性较差，在进行Web访问的时候，用户的某些敏感信息，账号、密码等可能会被恶意窃听，为保证这些敏感信息能加密传输，Netscape公司设计了SSL(Secure Sockets Layer)协议用于对HTTP协议传输的数据进行加密，从而产生了HTTPS。SSL协议在1999年被升级为TLS（Transport Layer Security）1.0，目前最新的为TLS1.3[16]。TLS协议连接服务器，支持X.509数字认证，并使用40位关键字作为RC4流加密算法，这对于商业信息的加密都是有利的。阿里巴巴于2015年底旗下电商平台已实现全站HTTPS，谷歌和百度等搜索引擎也先后启用了HTTPS，保护用户隐私，防止流量劫持。

HTTPS和HTTP大部分情况下是相同的，浏览器会设立一个连接到Web服务器指定的端口。当服务器接收到请求时，会返回一个状态码以及信息，可能是正确的信息数据，或者某个错误信息的内容。HTTPS在HTTP上加以SSL协议，导致两者之间存在一些不同，具体如下表所示：

表 2-1 HTTPS和HTTP区别

|  |  |  |
| --- | --- | --- |
|  | HTTP | HTTPS |
| URL | http:// | https:// |
| 安全 | 否 | 是 |
| 端口 | 80 | 443 |
| OSI模型中 | 应用层 | 传输层 |
| 数据加密 | 不需要 | 需要 |
| 认证证书 | 不需要 | 需要 |

* 1. HTTPS工作原理

HTTPS协议传输的数据，在客户端和服务器端都要进行数据的加密、解密，整个传输过程中数据都是经过加密的，所以保证了数据的安全性。这就规定了客户端和服务器端需要有一致的数据加解密的规则，具体HTTPS协议传输数据工作原理如下：

1. 客户端提交HTTPS请求：用户在浏览器输入一个https的网址，然后连接到服务器的443端口；
2. 服务器端配置：采用HTTPS协议的服务器必须有一套完整的数字证书，获取数字证书的方式有两种。一是自己制作，采用java自带的keytool.exe生成keystore文件，进入命令行，输入“keytool -genkey -alias tc -keyalg RSA -keystore c:\server.keystore -validity 3650”，在C盘根目录下得到；二是向组织申请，DigiCert Global Root CA、StartSSL、Symantec等都可以。区别在于自己制作的证书需要客户端验证通过才能访问，而使用受信任公司申请的证书，浏览器不会弹出提示页面。得到的数字证书中包括一对公钥和密钥，用于对传输的数据进行加解密；
3. 传送证书：服务器端将证书，发送给客户端，包括证书的颁发机构、过期时间等信息；
4. 客户端解析证书：由客户端的TLS完成。首先，验证接收到的公钥是否有效，如过期时间、颁发机构等信息。如果发现异常信息，则会警示该证书有问题；如果证书没有问题，就生成一个随机值。然后用证书对这个随机值加密；
5. 传送加密信息：客户端将经证书加密码后的随机值传送给服务器端，那么客户端和服务器端的通信就可以通过这个随机值来进行加密、解密；
6. 服务器端解密信息：服务器端对接收到的私密信息用私钥解密，对需要传回给客户端的信息进行对称加密（将信息和私钥通过算法混合在一起，只有知道私钥才能获取内容）；
7. 传输加密后的信息：这部分加密信息，能够在客户端被还原；
8. 客户端解密信息：客户端用之前得到的私钥对收到的信息进行解密，获得请求的返回数据。在整个过程，即使被外部监听，传输的信息经过加密，也不会泄露。

工作流程图如下所示：

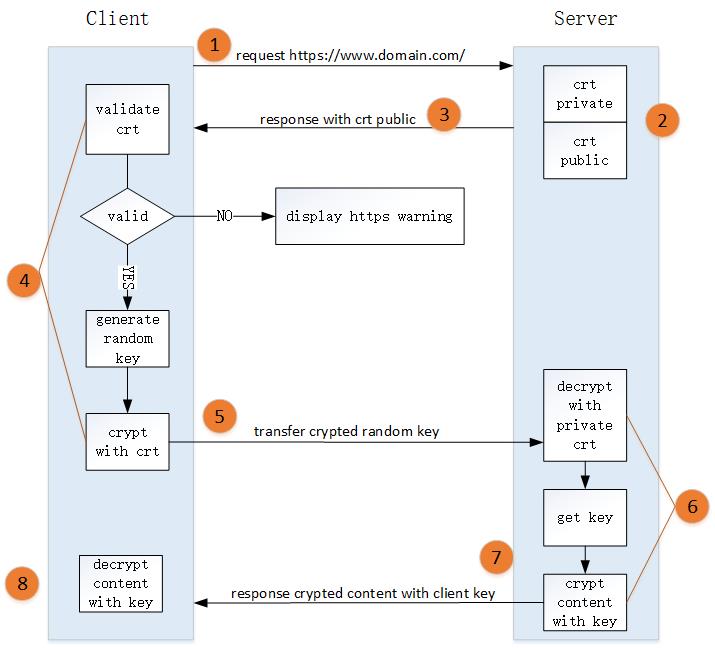


图 2-6 https加密过程图

## GZIP

* 1. GZIP基本概念

GZIP是GNU Zip的缩写，最早由Jean-loup Gailly和Mark Adler创建，用于UNIX系统的文件压缩。现今已经成为Internet 上使用非常普遍的一种数据压缩格式，或者说一种文件格式，在Linux系统中，后缀为.gz的文件就是GZIP格式。GZIP默认使用Deflate算法压缩数据部分。

HTTP协议上的GZIP编码是一种用来改进WEB应用程序性能的技术。大流量的WEB站点常常使用GZIP压缩技术来让用户感受更快的速度。当用户访问网站某个功能时，服务器将网页内容压缩后传输给用户，出文本内容可压缩到原大小的40%。对要传输的数据进行压缩，可以减少传输的时间，节省流量，改善用户的浏览体验。另一个潜在的好处是GZIP与搜索引擎的抓取工具有着更好的关系。例如 Google就可以通过直接读取GZIP文件来比普通手工抓取更快地检索网页。在Google网站管理员工具（Google Webmaster Tools）中可以看到，sitemap.xml.gz 是直接作为sitemap被提交的[17]。

* 1. Deflate算法

Deflate算法是一种将LZ77压缩算法和Huffman编码算法相结合的无损数据压缩算法[18]，最初是由Phil Katz在他第二版PKZIP归档工具中首次提出的，后来定义在RFC 1951规范中。广泛认为DEFLATE不受任何专利所制约，并且在LZW（GIF文件格式使用）相关的专利失效之前，这种格式除了在ZIP文件格式中得到应用之外也在GZIP压缩文件以及PNG图像文件以及Phil Katz设计的ZIP文件中得到了应用。

LZ77算法主要完成重复字符的消除，LZ77的实现方法大都进行了专利保护，Deflate提出的无专利的设计是基于LZ77实现的[19]。在对文件块进行压缩的过程中，如果发现一系列重复字符串，则用指向历史重复字符串的向后引用代替该重复字符串。Huffman编码算法是一种无损数据压缩算法，在GZIP压缩的第二阶段对数据进行进一步的压缩，将常用的字符用较短的符号代表，不太常用的字符用较长的符号代表。Huffman编码采用的方法产生一个没有重叠区间的非前缀树，在每个字符的比特序列的长度与需要被编码的概率成反比。符号越有可能被编码，其对应的比特序列越短。

* 1. GZIP与Deflate对比

目前GZIP算法定义在RFC1952标准中，是一种Lempel-Ziv（LZ77）译码格式[20]。GZIP是一种数据格式，默认使用Deflate算法压缩数据部分，Deflate是一种压缩算法，是Huffman编码的一种加强。GZIP和Deflate解压的代码几乎相同，可以合成一块代码。

简单的说，GZIP就是带有一些头部可和尾部的Deflate。Deflate使用inflateInit()，而GZIP使用inflateInit2()进行初始化，比 inflateInit()多一个参数:-MAX\_WBITS，表示处理raw deflate数据。因为GZIP数据中的zlib压缩数据块没有zlib header的两个字节，使用inflateInit2时要求zlib库忽略zlib header。在zlib手册中要求WindowBits为8.15，但是实际上其它范围的数据有特殊作用，如负数表示raw deflate。目前HTTP1.1定义了两种常用的HTTP压缩：DEFLATE和GZIP，即分别为：“Content-Encoding:deflate”和“Content-Encoding:gzip”，但是目前浏览器对“Content-Encoding:deflate”的支持不好，Verve工作室曾经测试了一个庞大的浏览器数量，结果是不好的，跨浏览器最好的压缩格式还是使用GZIP。

* 1. Content-Encoding的处理过程

WEB服务器对HTTP过程中Content-Encoding的处理过程如下：

1. Web服务器接收到浏览器的HTTP请求后，检查浏览器是否支持HTTP压缩（Accept-Encoding信息）；如果HTTP头显示有“Accept-Encoding:gzip”或“Accept-Encoding:gzip，deflate”，则表示浏览器支持压缩；
2. 如果浏览器支持HTTP压缩，Web服务器检查请求文件的后缀名；
3. 如果请求文件是HTML、CSS等静态文件，Web服务器到压缩缓冲目录中检查是否已经存在请求文件的最新压缩文件；
4. 如果请求文件的压缩文件不存在，Web服务器向浏览器返回未压缩的请求文件，并在压缩缓冲目录中存放请求文件的压缩文件；
5. 如果请求文件的最新压缩文件已经存在，则直接返回请求文件的压缩文件；在响应头中，指明“Content-encoding:gzip”；
6. 如果请求文件是动态文件，Web服务器动态压缩内容并返回浏览器，压缩内容不存放到压缩缓存目录中。

如下图所示：

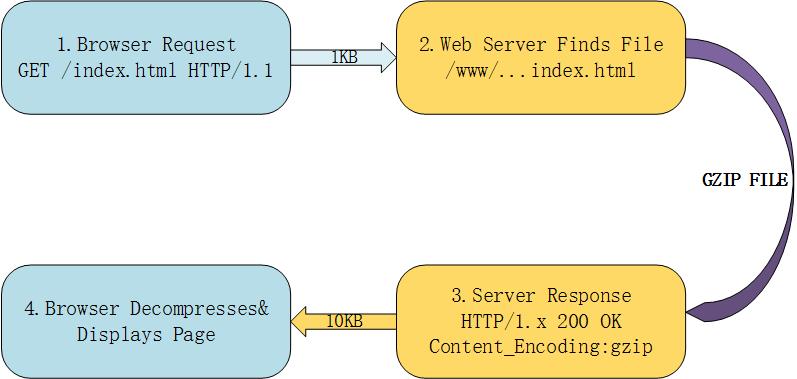


图2-7 GZIP压缩过程

* 1. GZIP压缩比

以下是通过抓包，在互联网上实际获取到的原始文件大小与GZIP压缩后的大小对比数据，包括了各种文件格式。

表 2-2 文件GZIP压缩对比【21】

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文件类型 | 文件路径 | 原始大小 | GZIP后大小 | 压缩比 |
| JS | https://ss1.bdstatic.com/5eN1bjq8AAUYm2zgoY3K/r/www/cache/static/protocol/https/jquery/jquery-1.10.2.min\_f2fb5194.js | 91.48KB | 32.96KB | 64% |
| JS | https://ss1.bdstatic.com/5eN1bjq8AAUYm2zgoY3K/r/www/cache/static/protocol/https/global/js/all\_async\_search\_a582e6e2.js | 200.49KB | 65.98KB | 67.1% |
| html | https://www.baidu.com/?tn=95184484\_hao\_pg | 98.2KB | 26.79KB | 82.7% |
| html | http://www.163.com/ | 704.94KB | 186.3KB | 73.6% |
| CSS | http://img3.126.net/kaola/dsp1e/css/ad750x100.css?v=3.15 | 3.07KB | 0.96KB | 68.% |
| CSS | https://ss1.bdstatic.com/5eN1bjq8AAUYm2zgoY3K/r/www/cache/static/protocol/https/soutu/css/soutu\_b8a8c9ad.css | 6.62KB | 1.84KB | 72% |
| swf | http://img1.126.net/channel8734/022740/640100\_0317\_sy03.swf | 24.9KB | 18.67KB | 25% |
| swf | http://img1.126.net/channel8734/021096/640100\_1112.swf | 32.76KB | 24.57KB | 25% |
| jpeg | http://f2.mediav.com/fpass/item1/20160304/23874-5999875170349317011\_160x160\_20160304215001\_49377.jpg | 5.98KB | 4.2KB | 29.8% |
| jpeg | http://f2.mediav.com/fpass/item1/20160222/23874--1669022456395391293\_160x160\_20160222131955\_10268.jpg | 4.2KB | 2.94KB | 30% |

备注：压缩比 =（原文件大小 - GZIP压缩后大小）/ 原文件大小

从表格中可以看出，对于基于文本的内容压缩传输，GZIP有很高的压缩比。但是对于非文本的，比如图片、flash、视频等，则压缩比不大，一般网站对于非文本的内容都不会采取GZIP压缩。

## 本章小结

本章主要针对本文中“两舱”休息室管理系统所涉及的相关知识、技术进行介绍。

本章首先介绍了系统采用的SSH应用框架的相关知识，着重分析了Struts2、Spring、Hibernate各框架的的特性以及运行流程。

紧接着，在本章的第二节，介绍了FreeMarker技术的基本概念，并详细分析了FreeMarker在MVC模式的视图层的优势。

在本章的第三节，介绍了Quartz任务调度框架，详细介绍了Quartz的核心模块，以及在任务调度中定时任务对于企业信息系统的重要性。

在本章的后两节中，主要介绍了关于提升系统性能的两项技术：HTTPS和GZIP。说明网站启用HTTPS代替HTTP协议对于系统安全的重要性，以及启用GZIP对于提升网站访问效率等性能的优势。

# 第三章 系统设计与实现

## 系统架构

本系统选择了基于B/S模式下的MVC三层架构开发，采用分层的思想，实现Web系统的功能分工。其中，Model层实现系统的业务逻辑，View层用于与用于的交互，Controller层是Model层和View层的中间桥梁，可以将后台经过逻辑处理得到的数据合理的显示到界面中，又能接收前端的页面请求数据，交由相应的逻辑层进行处理。B/S模式下的MVC三层结构最大的优势在于能够实现“分而治之”，降低系统耦合度。系统架构如图3-1所示。

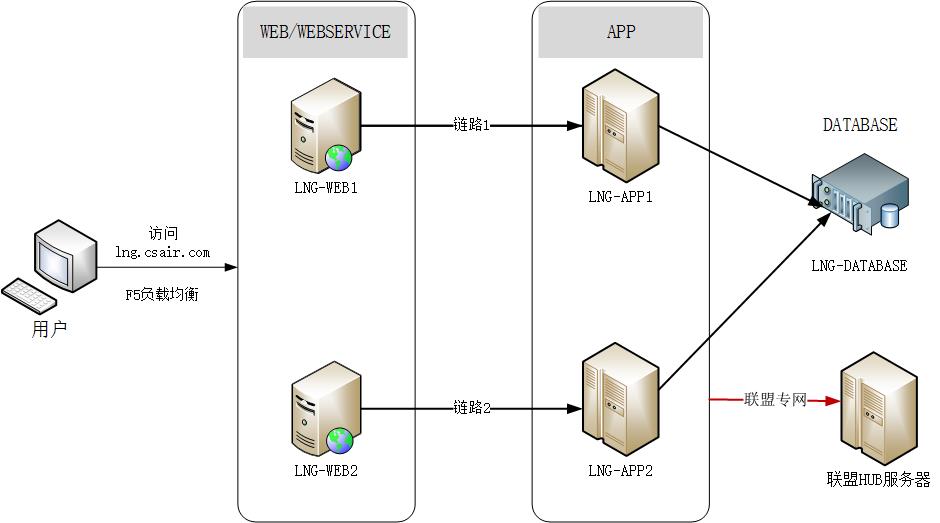


图3-1 系统架构图[22]

系统通过域名lng.csair.com访问，考虑到高访问量的情况，采用硬件的方式进行负载均衡处理，将访问请求合理的分配到不同的链路当中。采用双链路的设计，一方面，能够均衡系统负载，提高访问效率；另一方面，能有效的提高系统的容灾能力。可以根据系统访问量的多少考虑增加链路。Web服务器、应用服务器以及数据库服务器分别部署在不同的机器上，系统采用远程调用的方式进行各层之间的沟通。其中，APP应用服务器访问联盟HUB服务器时，需要通过专门的联盟专网。

## 业务流程

图3-2 所示为系统业务流程图。从图中可以看到，休息室管理人员通过扫描枪，扫描旅客登机牌上的条码，读取旅客信息到客户端。客户端按照既定的规则解析旅客信息，并录入数据。客户端将获得的旅客数据发送至应用程序服务器，应用程序服务器整合客户端的数据和离港旅客数据，对需要进入两舱休息室的旅客进行识别。若没有查询到离港数据，则进行ET数据查询，并对ET获得的旅客信息通过SVC系统进行数据填充，补全旅客信息。对于能进入休息室的旅客信息进行保存到数据库以供后期休息室管理人员进行报表统计，并添加服务标识。应用服务器通过获得实时的机场航班信息，不断更新进入休息室的旅客航班信息，生成对应的航班起飞提醒，休息室管理人员对旅客进行登机提醒。

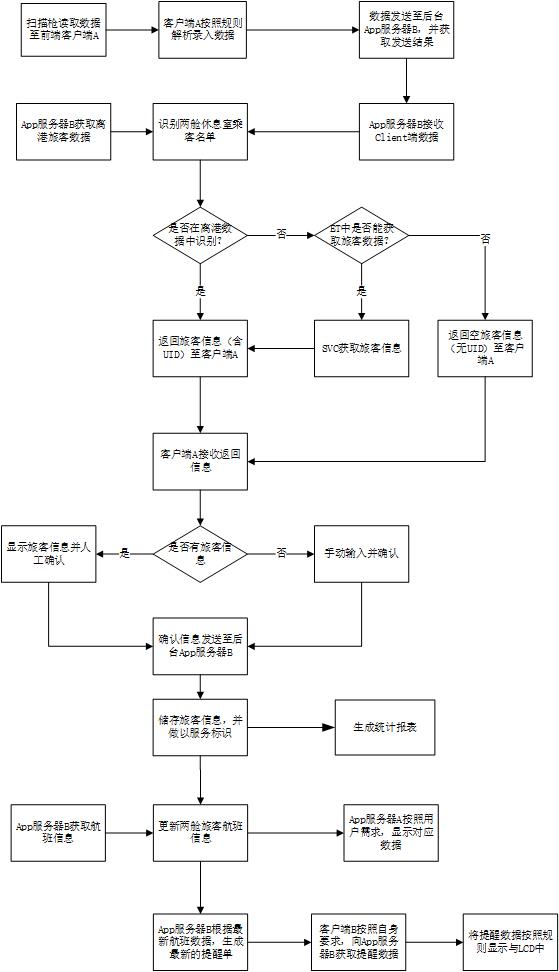


图 3-2 系统业务流程图

## 系统功能模块

系统按照各模块功能划分为前端数据录入、后台数据处理、休息室旅客管理以及统计报表等模块。

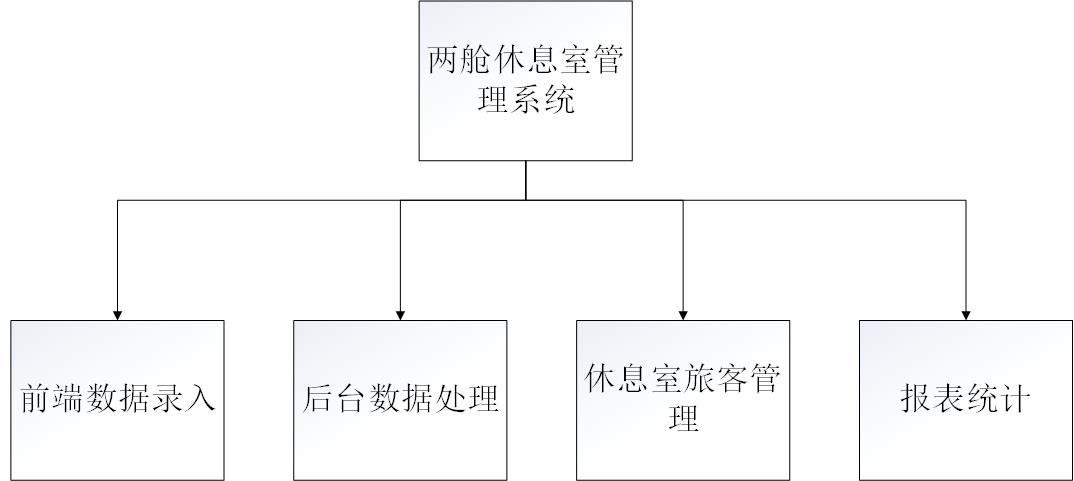


图 3-3 功能模块图

其中，前端数据录入模块对不同机场的条码规则进行解析，对不同休息室录入的数据进行识别以及前期的数据解析和数据发送；后台数据处理模块主要是获取其他航空系统的数据，对进入“两舱”休息室的旅客进行识别，为前端数据录入提供最新的客户端配置更新信息；休息室旅客管理模块对“两舱”休息室旅客数据、旅客航班信息以及休息室管理；报表统计模块主要是以不同的规则统计与“两舱”休息室相关的旅客和财务报表。

### 3.3.1 前端数据录入

（1）功能

前端数据录入为客户端系统，以便于对不同机场的条码规则进行解析，以及对不同休息室所录入的数据进行识别。该客户端的主要功能是支持“两舱”休息室旅客管理系统的前端数据录入，以及数据的前期解析和数据发送。客户端要达到的最终目的如下：

1. 获取扫描枪输入的数据，并加以分析；
2. 定时往服务端发送已录入的有效数据，并标注是哪个客户端所发送的数据；
3. 获取旅客识别信息，进行人工确认；
4. 出现网络故障时，能重新将发送失败的数据再次发送。
5. 对录入数据的分析规则的修改。

（2）模块

按照客户端所需功能，该客户端将分成四个模块，包括：

1. 配置管理模块：对客户端需要配置的信息进行管理，包括：当前客户端所在机场和所属休息室，数据发送的服务端地址，数据检验，数据解析规则等。因为每个客户端都代表着不同的“两舱”休息室，所发送的数据也需要包含所在休息室的代号，所以在系统安装后，需要对该客户端进行初始化配置。而且，机票的条形码在不同机场中，所能解析的规则未必完全相同，因此，解析的规则也同样需要用户手动配置。
2. 数据录入模块：获取扫描枪所录入的信息，按照已配置的数据规则对获取的数据进行检验，并且对通过检验的数据按照解析规则进行解析，最终储存本地数据库中，以待数据传输。需要配置的信息包括：录入数据的长度，值机号(bodyNum)、航班号(flightNum)、航班日期(flightDate)的解析规则。
3. 数据发送模块：按照服务端所需，将本地的数据以XML格式发送至服务端。并按照返回结果，修改客户端数据的数据状态。对发送失败的数据，按照处理规则，进行数据再发送流程。需要配置的信息包括：服务器地址。
4. 数据显示模块：按照服务器返回信息，显示旅客信息，并提供由人工修改及确认旅客信息的功能。

（3）流程图

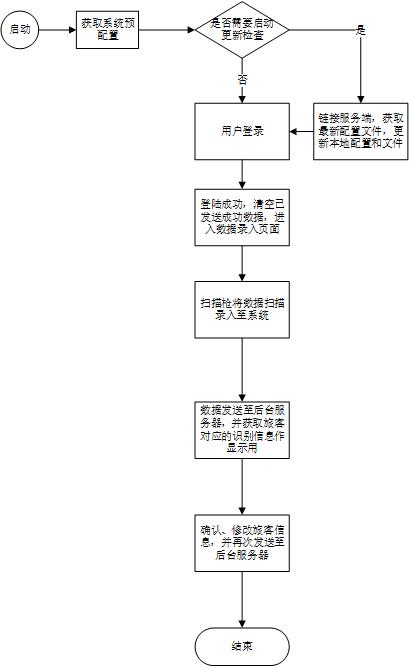


图3-4 数据录入流程图

（4）实现

通过扫描枪扫描得到的二维码数据格式如下所示

|  |
| --- |
| **条码信息:**  3CP#1C01#01V#05GUANJIANHUA #07关建华#08CZ #103869 #11@12#14NH9JQ4#151135#  1716#1821DEC#191055#2037C#21B212#22001#24NI620104198205020535#25CNY590 #28ET  KT#297841187323275/1#30GUANGZHOU#31CAN#32R#33V#35HANG ZHOU#36HGH#39XXX#43CZ 2139  22329600#44C#53R#61CZ3869 2137CCAN001#62广州#63杭州#72M1GUANJIANHUA ENH9  JQ4 CANHGHCZ 3869 356V037C0001 147>3180O 6356BCZ 29784118732327500CZ CZ 213922329600 20K#7752#$ |

以下是白云机场两舱休息室对于登机牌二维码数据的识别规则

|  |
| --- |
| **条码识别规则：**  3CP#1C01#01V  #05:旅客姓名（英文）  #07：旅客姓名（中文）  #08：承运人  #10：航班号  #11@12  #14：PNR号  #15：起飞时间  #  1716  #18：航班日期  #19：登机时间  #20：座位号  #21：登机口  #22：值机序号  #24：证件号  #25：票价  #28：  #29：票号  #30：始发站拼音  #31：始发站三字代码  #32R：  #33：舱位  #35：目的地三拼音  #36：目的地三字代码  #39：  #43：常客卡号  #44  #53R  #61CZ3869 2137CCAN001  #62：始发站中文  #63：目的地中文  #72：二维码内容  #7752#$ |

前端扫码得到的数据被识别成一个个的字段，分别注入到相应的属性中。通过AJAX/JSON技术，将全部字段传到后台处理，完成前端数据录入功能。

|  |
| --- |
| **var** ajaxScan = $.ajax({  type: "POST",  timeout: 20000,  url: "recordAndRegPsgPassenger.action?",  data: $("#recogForm").serialize(),  dataType:"json",  success: **function**(data) {  **if**(data.code==0) {  **var** request = data.attachment;  setRows(request, pramJson);  } **else** {  document.getElementById("fSound").play();  $(getRowByID(trackId)).find("div").html('<span style="color:red;">识别失败</span>');  $(getRowByID(trackId)).find("#note").attr('display','block');  $(getRowByID(trackId)).find("input").attr('display','block');  }  },  complete : **function**(XMLHttpRequest,status){ // 请求完成后最终执行参数  **if**(status=='timeout'){ // 超时,status还有success,error等值的情况  　　 ajaxScan.abort();  **var** data= {  　　 "benefitE": "N",  "bodyNum": pramJson.bodyNum,  "flightNT": pramJson.flightNo,  "scheduledFDDT": pramJson.flightDate,  "loungeId": pramJson.loungeId,  "flightOT": pramJson.airport,  "cabinCT": pramJson.cabin,  "origin": pramJson.origin,  "operatingCode": pramJson.operatingCode,  "errorOrADCode": "timeout",  "errorOrADMessage": "识别超时"  };  　　 setRows(data, pramJson);  　　}  }  }); |

### 3.3.2 后台数据处理

（1）功能

后台数据处理为应用服务系统，该部分的主要功能是获取多方航空系统数据，对进入“两舱”休息室的旅客进行识别，以及对“前端数据录入”和“前端航班提示”发送配置更新信息，并且对“前端航班提示”发送最新旅客航班数据信息。后台数据处理要实现的功能点如下：

1. 为“前端数据录入”提供信息接收接口，并返回旅客识别信息；
2. 为“前端数据录入”提供旅客信息进入“两舱”休息室数据录入接口；
3. 获取离港旅客数据，识别旅客身份；
4. 获取航班信息数据，更新旅客航班信息；
5. 为“前端数据录入”和“前端航班提示”提供最新的客户端配置更新信息；
6. 为“前端航班提示”提供即时的休息室旅客航班信息；
7. 信息的获取、发送、接收规则的配置。

（2）模块

按照所需功能将分成六个模块，包括：

1. 前端旅客信息接收模块：为“前端数据录入”的数据发送提供接口，旅客身份识别模块，对旅客数据进行识别，并返回旅客信息至“前端数据录入”客户端。
2. 旅客进入“两舱”休息室信息录入模块：为“前端数据录入”的数据发送提供接口，对进入休息室的旅客进行信息录入。
3. 旅客离港数据获取模块：从离港数据系统中，按照配置要求，定时获取旅客离港数据，并保存于本地数据库中，以待旅客身份识别模块的调用。需要配置的信息包括：离港服务器地址，获取离港数据定期获取的时间间隔，需要获取离港数据的机场列表。
4. 航班信息数据获取模块：按照配置中所需的机场信息，向SOC获取最新的航班时刻动态。需要配置的信息包括：SOC服务器地址。
5. 旅客身份识别模块：对传入的旅客信息进行精准识别，先从以获取的离港数据中进行识别，若识别不出，则向ET系统中获取旅客资料，再从SVC中对旅客资料做精准识别。需要配置的信息包括：ET服务器地址，SVC服务器地址。
6. 旅客航班信息更新模块：对在“两舱”休息室的旅客航班信息做实时更新，并对将要起飞的航班旅客，进行标示。对每个获取信息的休息室“航班信息提醒”客户端，只实行有效增量提供，即只发送的信息要同时满足：信息在有效期中，信息从没发送过或信息发送后曾经过修改。

（3）流程图

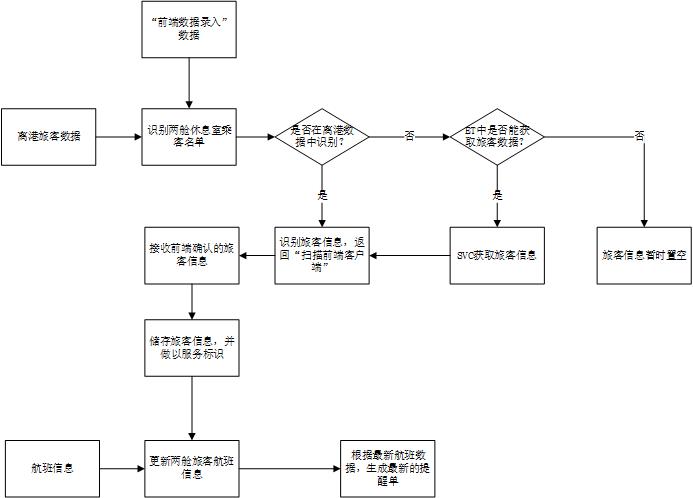


图3-5 数据处理流程图

（4）实现

在Web层的action中，通过属性驱动的方式获取前端页面各字段的值，获取待识别旅客信息。依次调用离港旅客数据、ET常旅客数据以及SVC旅客信息，对旅客进行识别。若识别通过则保存旅客信息至数据库，在旅客表中添加一行记录。

采用httpClient调用离港指令接口：

|  |
| --- |
| String url = "http://" + host + ":" + port + uri + path;  url = url.replace("null", "");  String userStr = userName + "," + password;  String encodedAuthorization = "Basic "  + org.apache.cxf.common.util.Base64Utility  .encode(userStr.getBytes());  GetMethod getMethod = **new** GetMethod(url);  getMethod.setRequestHeader("Authorization", encodedAuthorization);  HttpClient httpClient = **new** HttpClient();  httpClient.getParams().setContentCharset("UTF-8");  httpClient.getParams().setParameter(  CoreConnectionPNames.CONNECTION\_TIMEOUT, Integer.*valueOf*(timeOut));  // 读取超时  httpClient.getParams().setParameter(CoreConnectionPNames.SO\_TIMEOUT, Integer.*valueOf*(timeOut));  **try** {  httpClient.executeMethod(getMethod);  **if** (getMethod.getStatusCode() == HttpStatus.SC\_OK) {  String message = getMethod.getResponseBodyAsString();  LngPrPassengerInfoVo psgInfo = (LngPrPassengerInfoVo) xstream.fromXML(message);  **if**("OK".equalsIgnoreCase(psgInfo.getMessageTip())) {  **return** psgInfo;  }  } **else** {  String errorStr = getMethod.getResponseBodyAsString();  String errorTip = "";  **if** (StringUtils.isNotBlank(errorStr) && errorStr.indexOf("message")!=-1  && errorStr.indexOf("\"}")!=-1) {  errorTip = errorStr.substring(errorStr.indexOf("message")+10, errorStr.indexOf("\"}"));  }  **throw** **new** Exception(errorTip + ": " + url + " ---- " + errorStr); // 细化异常提示  }  } **catch** (Exception e) {  ***logger***.error("离港接口异常: " + e.toString());  **throw** **new** Exception("离港接口异常 " + e.getMessage());  } **finally** {  getMethod.releaseConnection();  httpClient.getHttpConnectionManager().closeIdleConnections(0);  } |

### 3.3.3 休息室旅客管理

（1）功能

休息室旅客管理为web应用系统的一部分，该部分主要功能对两舱休息室旅客数据以及对休息室的管理。该部分要达到的最终目的如下：

1. 休息室信息的管理；
2. 休息室旅客信息管理；
3. 旅客详细信息查看。

（2）模块

按照所需功能将分成三个模块，包括：

1. 休息室管理模块

功能：对休息室信息进行增删改查操作。

* + - 1. 新增休息室页面，包含的信息如下表：

表3-1 休息室表字段

|  |  |  |
| --- | --- | --- |
|  | 是否必输项 | 备注 |
| 休息室名称 | 是 | 20个汉字字符 |
| 所属公司 | 是 | 包括南航14个分公司，5个子公司，代理合作的航空公司 |
| 所在城市 | 是 | 支持城市控件的点击输入，当选择所属公司后，根据公司所管辖范围缩小所能选择的城市，选中将自动的把对应中文城市名选入对应文本框中 |
| 机场 | 是 | 当城市选择后，可自动匹配城市所在的机场，当所选城市有多个机场时，可让用户用下拉菜单选择；机场支持手输，只能输入3个字母，如果是小写字母，自动转化为大写 |
| 休息室状态 | 否 | 状态有“可用”、“停用” |

* + - 1. 休息室信息修改

修改管理页面与新增页面样式相同。

* + - 1. 休息室查询页面，查询条件如下：
* 所属公司：默认为用户当前所属的公司，必输项。
* 城市：关联关系与新增休息室相同属性关联相同，非必输项。
* 机场：关联关系与新增休息室相同属性关联相同，非必输项。

1. 休息室旅客信息管理模块

功能：该模块的作用是对旅客在系统中的信息进行增删改查操作。其中查询休息室旅客信息查询条件：

1. 航空公司：必输项，南航和南航代理贵宾服务的航空公司（川航……），默认为“南航”。
2. 航班日期：必输项，提供日历控件，也可手输，支持手输8位日期，如 “20170418”，默认为当天日期；
3. 航班：非必输项，可输字母和数字，长度不超过7；
4. 休息室：必输项，“全部”和状态为“可用”的休息室，默认为“全部”。
5. 旅客状态：必输项，下拉菜单，“全部”、“未进入”、“已进入”、“离开休息室”、“登机”，默认为“全部”。
6. 旅客详细资料模块

调用CRM系统，获取旅客详细信息。根据不同管理人员的权限对旅客信息进行修改和删减。

### 3.3.4 统计报表

1. 功能

为业务人员提供休息室的各种数据统计，以减少业务人员的工作量。提供数据统计和分析的方法接口。处理程序调用接待记录模块的接口程序，获取相应的接待记录，通过梳理和汇总统计，最终将计算出来的结果保存到相应的数据表中。同时，本模块将提供相应的搜索API，供给报表展现端的调用，并提供相应的报表数据。其中数据统计分析程序提供给任务调度平台调用。

1. 实现

我们采用Apache POI用于在系统中生成导出Excel文件，Apache POI是目前流行的API，可以通过Java程序创建、修改、显示Microsoft Office文件，是由Apache软件基金会开发使用Java分布式设计或修改Microsoft Office文件的开源库。项目中主要引用的Jar包为”poi-3.6.jar”，程序中引入的package如下：

|  |
| --- |
| **import** org.apache.poi.hssf.usermodel.HSSFWorkbook;  **import** org.apache.poi.ss.usermodel.Cell;  **import** org.apache.poi.ss.usermodel.Row;  **import** org.apache.poi.ss.usermodel.Sheet;  **import** org.apache.poi.ss.usermodel.Workbook; |

具体实现代码如下：

|  |
| --- |
| FileOutputStream outStream = **null**;  **try** {  List<TouristsInfoDto> result = dataStatisticsService.findTouristsInfo(startDate, endDate, airport);  // 判断是否有记录  **if** (result == **null** || result.size() <= 0) {  Log.error(“无数据”);  }  Workbook workbook = getPsgWorkbook(result);  // 生成目录  String outputPath = ***FILE\_SPLIT*** + *passenger\_properties*.getValue("report.path")  + ***FILE\_SPLIT*** + path;  File reportOutputBaseFile = **new** File(outputPath);  **if** (!reportOutputBaseFile.exists()){  reportOutputBaseFile.mkdirs();  }  // 生成文件名  SimpleDateFormat smf = **new** SimpleDateFormat("yyyyMMdd");  Calendar c = Calendar.*getInstance*();  c.setTime(endDate);  c.add(Calendar.***DAY\_OF\_MONTH***, -1);  Date outEndDate = c.getTime();  String outputFileName = **null**;  **if** (startDate.equals(outEndDate)) {  outputFileName = "PassengerReport\_" + smf.format(startDate)  + "\_" + airport + ***DOC\_TYPE***;  } **else** {  outputFileName = "PassengerReport\_"+ smf.format(startDate)+ "-"  + smf.format(outEndDate) + "\_" + airport + ***DOC\_TYPE***;  }  String outputFileFullName = outputPath + outputFileName;  // 把相应的Excel工作簿存盘  outStream = **new** FileOutputStream(outputFileFullName);  workbook.write(outStream);  outStream.flush();  outStream.close();  // 记录报表信息到数据库  ReportFileInfo entity = dataStatisticsService.findReportFileInfoByDateAndAirport(  smf.format(startDate), smf.format(outEndDate), airport);  **if** (entity == **null**) {  entity = **new** ReportFileInfo();  }  entity.setTouristsFileName(outputFileName);  entity.setStartDate(smf.format(startDate));  entity.setEndDate(smf.format(outEndDate));  entity.setTouristsFilePath(outputFileFullName);  entity.setOpId("AUTO");  entity.setOpDate(**new** Date());  entity.setAirport(airport);  dataStatisticsService.saveReportFileInfo(entity);  } **catch** (Exception e) {  Log.error("生成报表PassengerReport出错："+e.getMessage());  **throw** e;  } **finally** {  **try** {  **if** (outStream != **null**) {  outStream.close();  }  } **catch** (IOException e) {  ***logger***.error(e.getMessage());  }  } |

## 网页优化

### HTTPS

启用HTTPS安全传输协议，提高系统的安全性能。在JBOSS服务器上配置HTTPS请求。

* 生成keystore文件

用本机java自带的keytool生成server.keystore文件。进入dos命令行，按照提示操作，在本机生成keystore文件，具体如下图所示：



* 修改JBOSS的server.xml文件

server.xml所在目录位置：

"/opt/jboss-eap-4.3/jboss-as/server/lng-web/deploy/jboss-web.deployer"

并做如下修改：

|  |
| --- |
| 在server.xlm增加  <Connector port="80" address="${jboss.bind.address}"  maxThreads="250" maxHttpHeaderSize="8192"  emptySessionPath="true" protocol="HTTP/1.1"  enableLookups="false" redirectPort="443" acceptCount="100"  connectionTimeout="20000" disableUploadTimeout="true" />  <Connector port="443" address="${jboss.bind.address}"  protocol="HTTP/1.1" SSLEnabled="true"  maxThreads="150" scheme="https" secure="true"  emptySessionPath="true"  clientAuth="false" sslProtocol="TLS"  keystoreFile="/opt/jboss-eap-4.3/jboss-as/keystore.jks"  keystorePass="123456"  ciphers="TLS\_ECDHE\_RSA\_WITH\_AES\_128\_CBC\_SHA256,TLS\_ECDHE\_RSA\_WITH\_AES\_128\_  CBC\_SHA,TLS\_ECDHE\_RSA\_WITH\_AES\_256\_CBC\_SHA384,TLS\_ECDHE\_RSA\_WITH\_AES\_256\_C  BC\_SHA,TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA256,TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA,TLS\_RS  A\_WITH\_AES\_256\_CBC\_SHA256,TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA" /> |

* 修改工程的web.xml文件

在项目的web.xml文件的末尾进行以下配置：

|  |
| --- |
| <security-constraint>  <web-resource-collection >  <web-resource-name >SSL</web-resource-name>  <url-pattern>/\*</url-pattern>  </web-resource-collection>  <user-data-constraint>  <transport-guarantee>CONFIDENTIAL</transport-guarantee>  </user-data-constraint>  </security-constraint> |

根据需要，web-resource-collection节点可以配置多个，而url-pattern则配置需要强制HTTPS的URL，本系统中采用的全站使用HTTPS。

### GZIP压缩

JBoss AS 7.0.x到AS 7.1.0，使用了新的结构，默认不支持GZIP压缩。解决办法是通过CompressionFilter。

1. 下载webutilities-0.0.6.jar到WEB-INF\lib下，官网版本只能下载到0.0.6。还可以通过pom.xml引入，最新版本0.0.8：

|  |
| --- |
| <dependency>  <groupId>com.googlecode.webutilities</groupId>  <artifactId>webutilities</artifactId>  <version>0.0.8</version>  <classifier>jdk15</classifier>  </dependency> |

1. 修改web.xml文件，增加以下filter：

|  |
| --- |
| <filter>  <filter-name>compressionFilter</filter-name>  <filter-class>com.googlecode.webutilities.filters.CompressionFilter</filter-class>  <init-param>  <param-name>compressionThreshold</param-name>  <param-value>50</param-value>  </init-param>  <init-param>  <param-name>ignoreURLPattern</param-name>  <param-value>..(flv|mp3|mpg)</param-value>  </init-param>  <init-param>  <param-name>ignoreMIMEPattern</param-name>  <param-value>image/.|video/.\*|multipart/x-gzip</param-value>  </init-param>  <init-param>  <param-name>ignoreUserAgentsPattern</param-name>  <param-value>.\*(MSIE).\*</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>compressionFilter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping> |

1. Filter参数说明

CompressionFilter的参数都是正则表达式，通过正则表达式匹配结果。

下面是可用的参数清单：

表 3-2 CompressionFilter的配置项

|  |  |  |
| --- | --- | --- |
| 序号 | 参数名 | 参数说明 |
| 1 | ignoreURLPattern | 忽略的URL模式 |
| 2 | acceptURLPattern | 接受的URL模式 |
| 3 | ignoreMIMEPattern | 忽略的MIME Type |
| 4 | acceptMIMEPattern | 接受的MIME Type |
| 5 | ignoreUAPattern | 忽略的User-Agent |
| 6 | acceptUAPattern | 接受的User-Agent |

## 本章小结

本章主要是关于“两舱”休息室管理系统的设计与最后功能的实现方面的介绍。

本章首先对系统的架构进行了介绍，表明系统的web层和ejb层分别部署在不同的服务器上，以及多链路的部署方式。

在本章的第二小节，用流程图的方式详细的介绍了系统的整个业务流程。

在本章的第三小节，详细介绍了系统各个功能模块的设计过程和思路。

在本章的最后，介绍了关于系统性能的提升上，HTTPS和GZIP的设计过程。

# 第四章 系统测试与运行结果

## 测试环境

本系统分web层和ejb层分别部署在不同的服务器上，在本机开发是运行在jetty上，发布生产环境是在linux操作系统中的JBOSS容器，各服务器的具体配置如表4-1所示，部署环境如图4-1。

表4-1 服务器配置

|  |  |  |  |
| --- | --- | --- | --- |
|  | | WEB服务器 | APP服务器 |
| CPU数目 | | 4 | 4 |
| 磁盘 | | 80G | 80G |
| 内存 | | 8GB | 8GB |
| 操  作  系  统 | 名  称 | 64位 Red Hat Enterprise Linux Server release 6.1 | 64位 Red Hat Enterprise Linux Server release 6.1 |
| 主  机  名 | NJF-VM-APP7 --- LNG-APP1  NJF-VM-APP8 --- LNG-APP2 | ZLQ-VM01 --- LNG-WEB1  ZLQ-VM02 --- LNG-WEB2 |
| 本  地  硬  盘  规  划 | /boot 200M ext4 主分区  swap 4G /opt 10G ext4 /var/log 10G ext4 /home 10G ext4 / 约40G ext4 | /boot 200M ext4 主分区  swap 4G /opt 10G ext4 /var/log 10G ext4 /home 10G ext4 / 约40G ext4 |

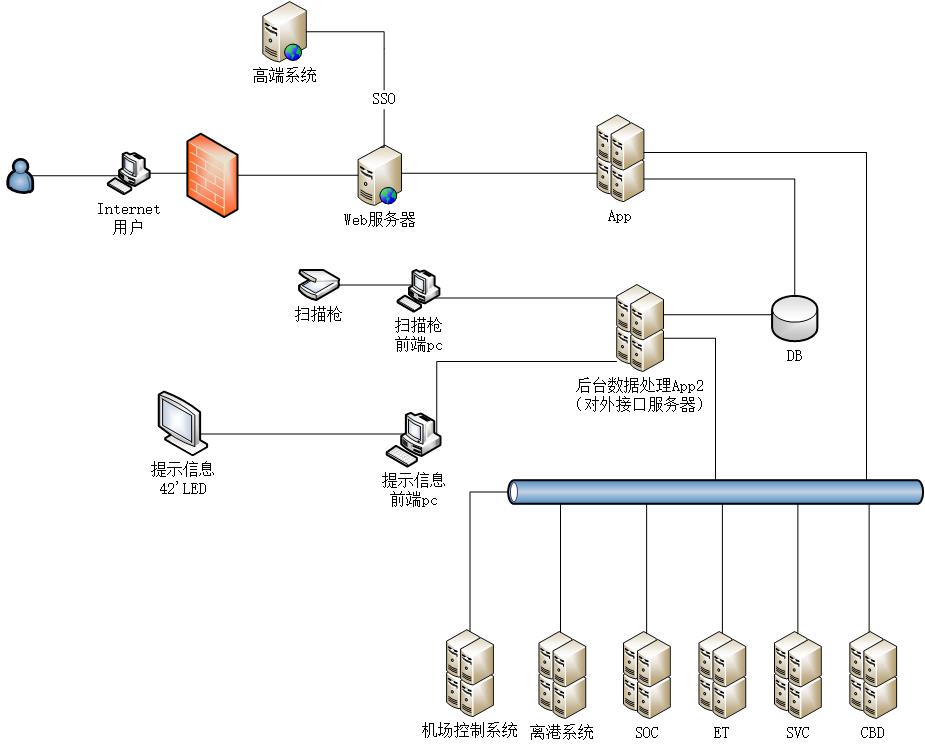


图 4-1 部署图

## 部署过程

在本机将通过Maven将项目打包，部署到生产环境的JBOSS上，具体部署过程有以下步骤：

1. 在命令行下进入LNG根目录：mvn clean install –Dmaven.test.skip=true 进行打包
2. 删掉部分依赖包：

LNG-Scheduler：jboss-tiny-client-4.3-client.jar

LNG-Web：jboss-tiny-client-4.3-client.jar,weblogic-lib-full.jar

LNG-WebService：jboss-tiny-client-4.3-client.jar,weblogic-lib-full.jar

1. 远程控制到监控机
2. 使用SSHClient连接到远程APP服务器和WEB服务器
3. [\\lng\_temp](file:///\\lng_temp)，新建两个目录，一个为放更新的，一个backup原来的
4. 查看jboss进程并关闭进程：

#ps aux|grep java

#kill -9 <进程ID>

1. 删除临时文件夹（${jbosshome}/server/lngejb、lngweb、lngws、scheduler）下：tmp、work文件夹

#rm –rf tmp work

1. 删除原有的项目文件
2. 启动应用（${jbosshome}/server/runejb.sh、runweb.sh、runws.sh、runscheduler.sh）

观看日志：tail –f /opt/runejb.txt 、runweb.txt、runws.txt、runscheduler.txt

## 运行结果

### 旅客管理模块

其中旅客识别模块，作为系统的主界面。具体效果如下



用扫描枪或者手工输入登机牌



保存随从



按照一定的条件查询进入休息室的旅客列表



修改旅客乘机信息



旅客姓名附有超链接，点击查询旅客详细个人信息



### 休息室管理模块

休息室操作员绑定

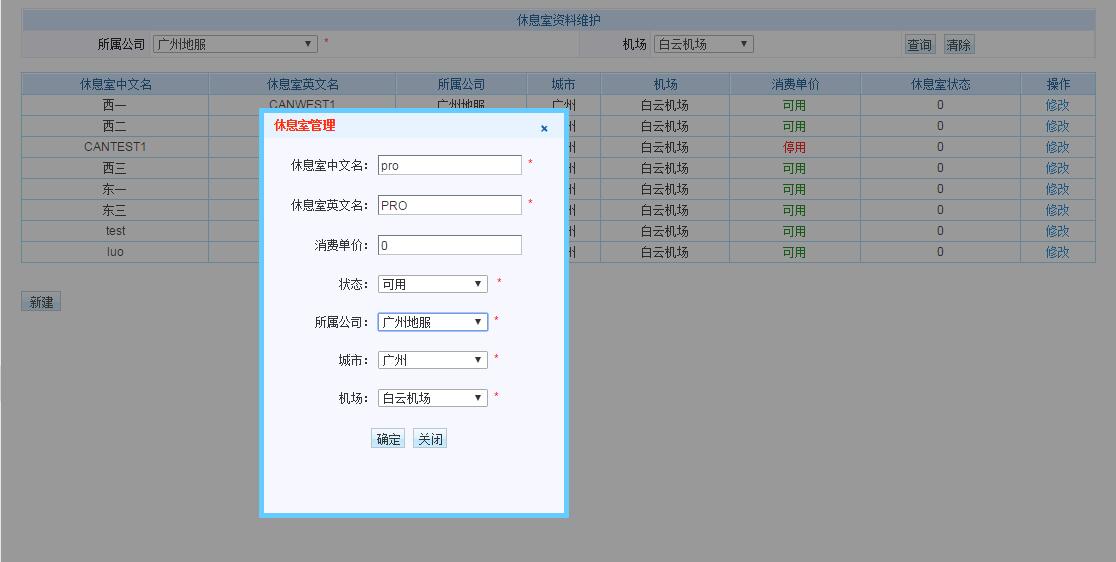


对休息室的维护管理部分：

查询部分



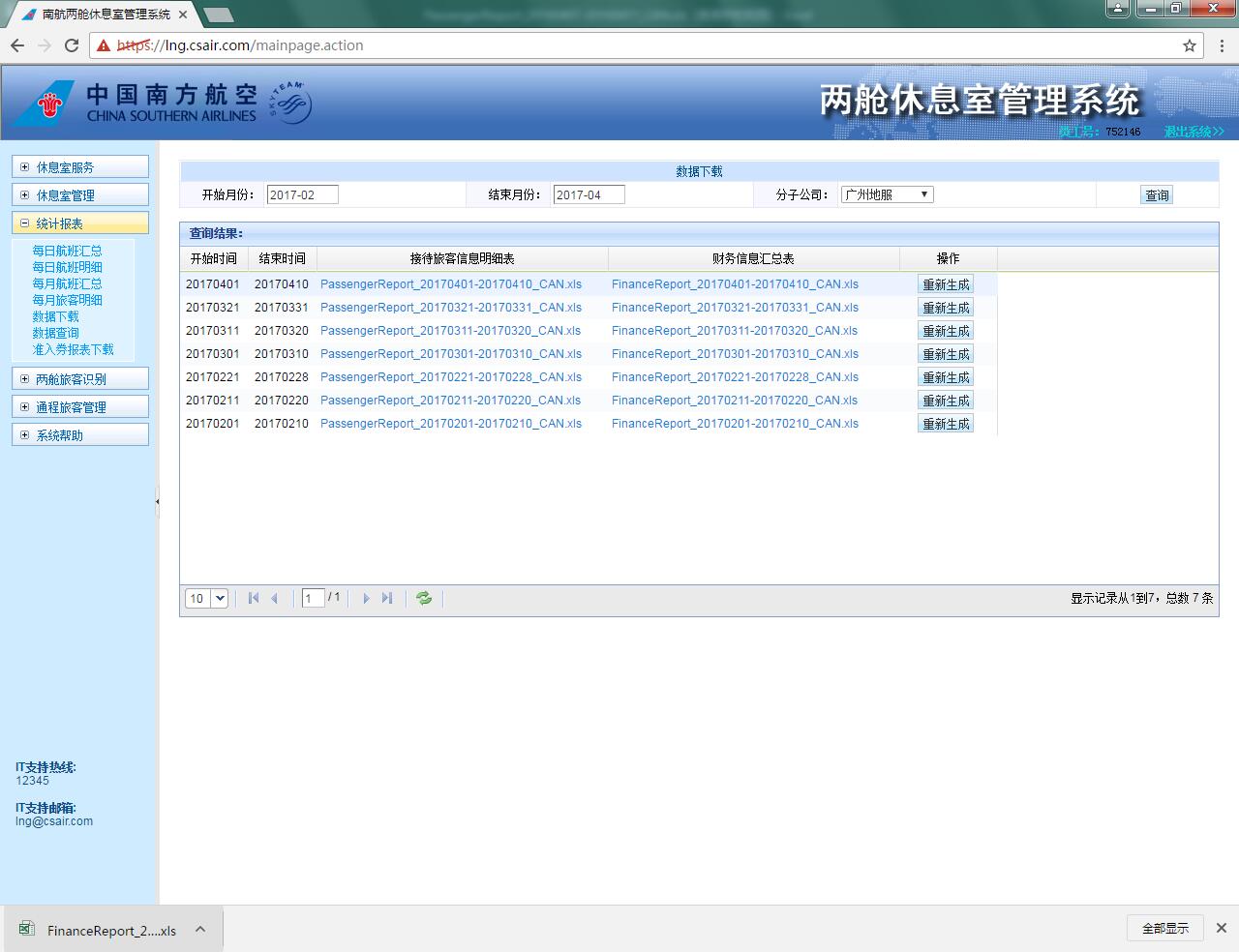
休息室更新



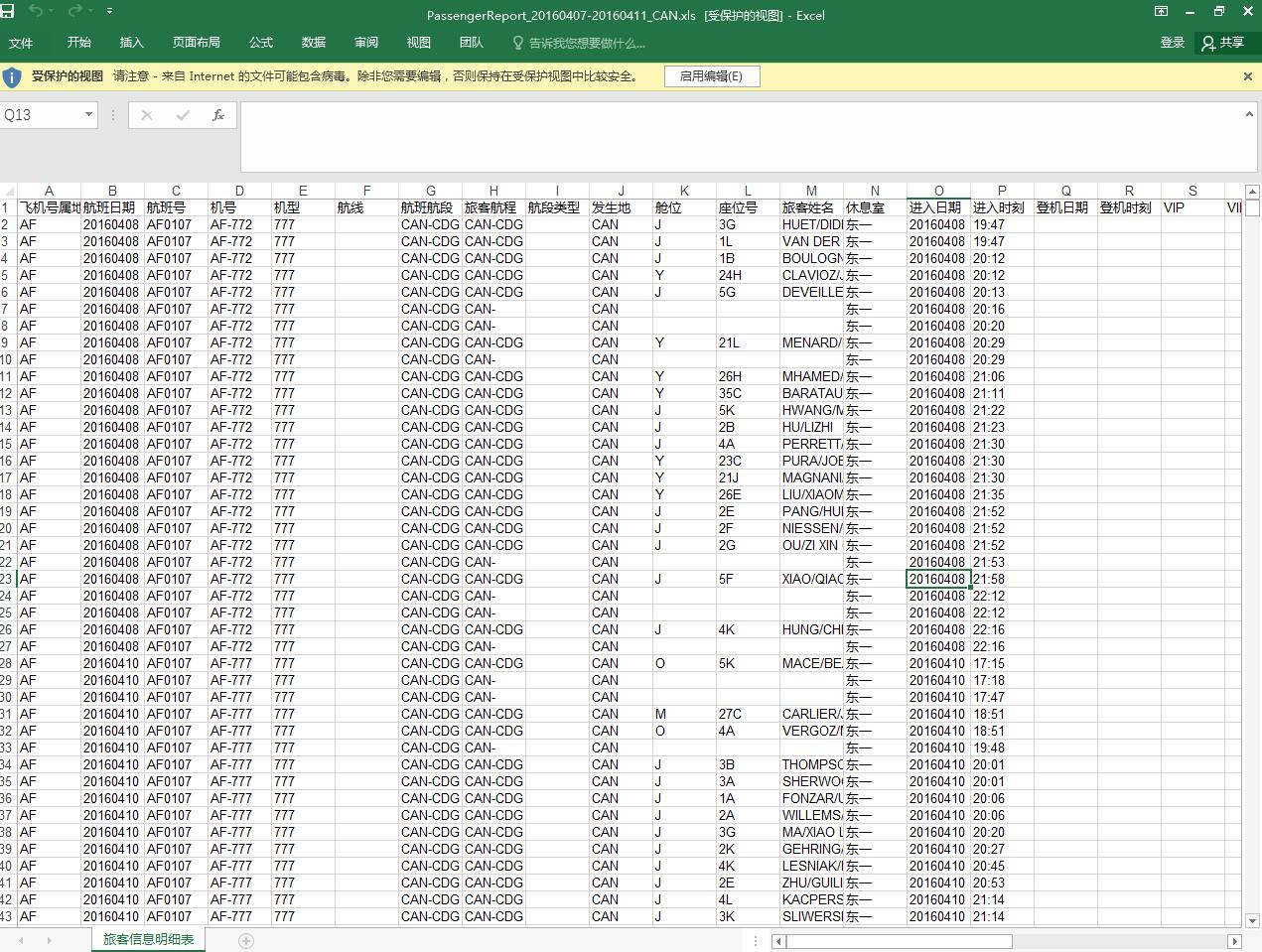
飞机属地管理



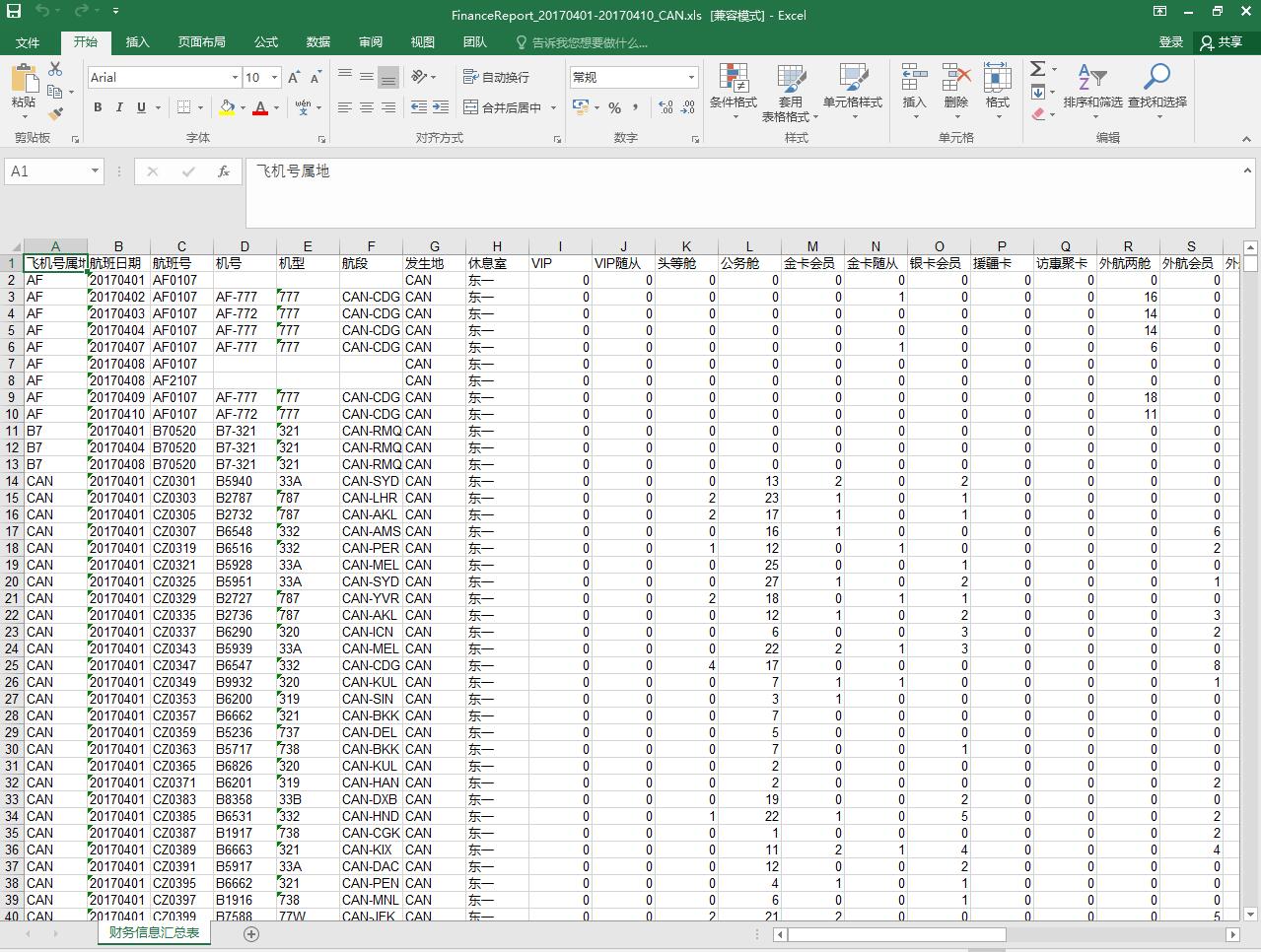
### 统计报表模块



旅客信息表



机场休息室财务汇总表



## 本章小结

本章主要对系统的实际运行结果进行说明，展示系统目前所完成的具体功能。

首先，介绍了系统部署的环境，APP服务器和WEB服务器的配置。

在本章的第二节，介绍了本机项目如何通过maven打包发布到生产服务器上运行的过程。

最后，根据目前系统实现的功能，分旅客、休息室以及报表统计三个部分展示系统运行结果图。

# 第五章 总结与展望

## 论文总结

对一个企业，尤其像民航运输业这种服务性行业的公司来说，是符合“二八定律”的，即企业80%的利润由20%的旅客创造，所以提高对于高端旅客的服务质量，提高高端旅客的“忠诚度”，对于每一家民航企业都是至关重要的。在这个信息化的时代，通过信息管理系统有效的整合、利用信息，以提高服务质量和高端旅客的“忠诚度”，对于民航企业的信息化发展具有重要意义。

针对目前高端旅客信息分散、机场休息室对于高端旅客信息的支持力度较小等问题，本文设计并实现“两舱”休息室管理系统，通过构建一个统一的管理系统，整合高端旅客信息、航班信息以及机场休息室信息等等，为服务人员及两舱旅客管理提供更好的数据支持。本系统的服务对象为乘坐中国南方航空航班的“两舱”旅客、南航金银卡旅客及随从；协议航空公司的“两舱”旅客、金银卡旅客及随从。

本系统采用目前较为流行的Web应用程序开源框架—SSH集成框架，设计并实现“两舱”旅客信息管理系统。在旅客识别、旅客信息记录、定时任务以及报表统计等方面已实现基本功能。本文首先介绍了构建一个专门服务于“两舱”等高端旅客的信息管理系统的必要性以及对航空企业的重要性。在本文的第二部分，详细介绍了系统相关的技术，着重分析了SSH框架中的Struts2、Spring、Hibernate框架的功能和特性，FreeMarker模块引擎在MVC视图层的作用，Quartz定时任务调度框架以及关于网页优化的相关技术HTTPS和GZIP压缩。然后，第三部分，详细介绍了此系统的实现过程，以系统功能模块为划分依据，分别介绍旅客管理模块、休息室管理模块、报表统计模块以及关于网页优化模块的实现过程和设计思路的分析。在本文的最后，就系统现有已实现的功能进行展示，从起始的旅客识别进入休息室，到休息室旅客的管理以及最后的服务人员对统计报表的管理，逐一进行说明。

## 工作展望

本文基于SSH 集成框架设计实现“两舱”旅客管理系统，主要是对于“两舱”旅客信息的整合处理，为高端旅客提供更好的信息化服务，已实现基本功能，但随着系统访问量的不断增加，发现系统仍然存在许多不足。未来的工作主要围绕以下几个方面展开：

1. 本系统目前是基于两条链路进行访问，随着系统访问量的增加，系统负载也许会存在问题，接下来可以考虑根据实际访问量，再增设新的链路。
2. 关于网页优化方面，目前增加了HTTPS 和GZIP 压缩，今后可以在Web 层增加新的功能，以提升系统性能。

# 参考文献

1. 中国民航局. 2015年民航行业发展统计公报.  
   <http://www.caac.gov.cn/XXGK/XXGK/TJSJ/201605/t20160530_37643.html>
2. 陈淑君. 打造“新两舱”服务 提升航空公司竞争力[J]. 空运商务，2005.
3. 张海燕. 东航中美航线两舱服务质量提升管理研究[D]. 上海交通大学，2012.
4. 梁弼. SSH框架整合技术研究及应用[J]. 西安文理学院学报:自然科学版, 2011, 14(3):16-19.
5. Erich Gamma， Richard Helm，Ralph Johnson，John Vlissides.Design Patterns: Elements of Reusable Object-Oriented[J].Software Addison-Wesley Professional，1995.
6. 林碧英.设计模式在电子商务交易网站中的应用[J].计算机系统应用，2005.
7. <http://blog.csdn.net/redarmy_chen/article/details/7342289>
8. 陈尚松, 杜旭英, 俞欢军. 基于Struts+Hibernate+Spring框架的毕业设计管理系统[J]. 计算机工程与设计, 2008, 29(15):4054-4057.
9. 黄烟波, 张红宇, 李建华. 基于Struts和Hibernate的J2EE架构[J]. 计算机时代, 2004(10):29-30.
10. <http://www.cnblogs.com/eflylab/archive/2007/01/09/615338.html>
11. 梁兴波. FreeMarker模板引擎在Java开发中的应用[J]. 硅谷, 2013(21):46-46.
12. 刘全飞，周相兵，LIUQuan-fei，等.基于FreeMarker的站群系统模板设计实现[J].信息技术，2015.
13. <http://www.xzbu.com/8/view-4759410.htm>
14. 胡利强，周冬初，王伟.Quartz调度器与Web程序整合的研究和应用[J].计算机与现代化，2010.
15. 丁振凡，李馨梅.Spring的任务定时调度方法的研究比较[J].智能计算机与应用，2012.
16. Clark J,Oorschot P C V.SoK:SSL and HTTPS:Revisiting Past Challenges and Evaluating Certificate Trust Model Enhancements[C]//Security and Privacy.IEEE,2013.
17. <http://baike.baidu.com/view/966625.htm>.
18. <http://www.tuicool.com/articles/uQRfAn>
19. 李明, 杨雷, 黎山峰. 不同压缩算法性能的研究[J]. 通信技术, 2009, 42(4):175-177.
20. <http://www.cnblogs.com/shuidao/p/3151067.html?utm_source=tuicool&utm_medium=referral>
21. <http://pagespeed.webkaka.com/youhua/gzip/>
22. 中国南方航空股份有限公司信息中心