浙 江 大 學

本 科 生 毕 业 设 计 报 告

项目名称 基于 SMO 算法 SVM 的试卷信息统计

姓　　名 陈威志

学　　号 3120000521

指导教师 吴健

专　　业 计算机科学与技术

学　　院 计算机科学与技术学院

A Dissertation Submitted to Zhejiang University for the Degree of Bachelor of Engineering

TITLE: Paper statistics based on SVM using SMO algorithm

Author: Weizhi Chen

Student ID: 3120000521

Mentor: Jian Wu

Major: Computer Science

College: College of Computer Science and Technology

Submitted Date: 05/30/2016

浙江大学本科生毕业论文（设计）诚信承诺书

1.本人郑重地承诺所呈交的毕业论文（设计），是在指导教师的指导下严格按照学校和学院有关规定完成的。

2.本人在毕业论文（设计）中引用他人的观点和参考资料均加以注释和说明。

3. 本人承诺在毕业论文（设计）选题和研究内容过程中没有抄袭他人研究成果和伪造相关数据等行为。

4. 在毕业论文（设计）中对侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

毕业论文（设计）作者签名：

_____年_____月_____日

摘要

本文主要对手写字符的识别进行研究，并实现了学生试卷识别和管理系统。手写字符的识别对数字化试卷管理系统的建立有着重要的意义。如何准确快速地提取出教师的判卷信息，并进行识别与统计是试卷识别和管理工作中关键的一步。在手写字符的识别方面，其难点在于不同人的手写字体变化多样，难以规范化；待识别的字符有较多的噪声干扰，而且在试卷系统中对于识别正确率的要求十分苛刻。目前国内外对各种手写字体的识别尚位于发展阶段，如基于神经网络的手写字体识别；基于笔划分析的手写识别等。往往识别效果都不甚理想，很难做到试卷系统中要求的极高的准确度。对于脱机状态的手写识别则更低。而且目前的研究工作没有针对评卷信息中的手写字体的识别，多数是对手写数字，字母的识别研究。因而在需要在传统的手写识别上，设计新的研究方案。所以本文的工作具有一定的研究和应用价值。

本文的具体工作和优势有：

1.在研究传统手写数字的特征提取基础上，结合手写字符(手写对号√和错号×)的特点，设计多种特征提取方法，通过实际调试，比较选出最能代表字符特性的一组特征，组成特征向量。

2.通过搜集大量不同手写字符样本，比较多种特征提取方法。最后本文选择了包括傅里叶变换特征，轮廓特征，笔画密度特征等 102 维特征向量。特征提取的结果良好。基本可以做到对样本空间的线性可分，对于线性分类器也可以有很好的效果。

3.建立一个基于 SMO 算法的 SVM 分类器。

4.通过大量样本的测试，选取最优的核函数，并通过网格搜索和交叉验证确定最优参数。实验结果表明，该分类器在实际使用中有良好的表现，对 100 余组不同手写字符的测试参数能有 100%的识别结果。

5.建立了学生试卷识别和管理系统。可以对试卷进行识别，信息统计，逐题查看，保存和读取功能。

关键词 图像处理 试卷管理 手写字符识别 支持向量机 SMO 特征提取 核函数 最优参数选取

Abstract

This paper focuses on study of identifying handwritten characters, and the implementation of student papers Identification and Management System. Character recognition is important for the establishment of digital paper management system. How quickly and accurately can the system extract the teacher's grading information, identify and statistics papers is one crucial step of the work of paper identification and management. It is difficult to recognize handwritten characters, due to diverse handwriting style which is hard to standardize. In addition, the process of handwriting recognition is easily disturbed by noise. What's more, the correctness required by a paper system is strict. Currently a variety of domestic and aboard handwriting recognition is still located in the stage of development, such as handwriting recognition based on neural network; stroke-based analysis of handwriting recognition. Recognition results are often unsatisfactory. It is difficult to achieve a high degree of accuracy required in the paper system. Offline handwriting recognition is even lower. And the current study does not recognize the handwriting marking symbol, the majority of the research is to identify the handwriting of numbers and letters. It is necessary to design new research programs which is based on traditional handwriting recognition. So work in this paper has a certain research and application values.

Specific work and the advantages of this paper are:

1. Based on the study of traditional handwritten digital feature, design a variety of feature extraction methods for correct mark √ and wrong mark×.

2. Extract 102 dimensions feature vector from numerous samples, which works well. The sample space is almost linearly separable. The result of classification is good even for the linearly classifier.

3. Construct SVM classifier using SMO algorithm.

4. Through numerous samples, choose the best kernel function. And select the best parameters using grid-search and cross-variation. According the test result, the classifier shows good performance. It can reach 100% correctness with 100 real

samples.

5. Construct student paper recognition and management system. It can recognize paper, count information, show contents by each problem, save and retrieve paper.

Keywords Image Preprocessing, Paper Management, Handwritten Character Recognition, Support Vector Machine, SMO, Feature Extraction, Kernel Function, Optimal Parameter Selection

目录

第1章 项目背景

1.1 课题背景

随着社会主义科学发展观的普及，我国教育工作变得越来越受人们的重视。现在的教育平台为不同年龄，不同职业的劳动者提供开放的，多样化的，社会化的受教育机会。因此考试就成了衡量教学水平，判断学生能力的一个重要标准。学生要参加越来越多的考试，这就给试卷的评判及分析工作带来了巨大的压力。试卷的审阅也向自动化，智能化，有效化的方向发展。因此一个数字化试卷管理系统就显得尤为关键。

而试卷的识别，分析与保存是现代数字化试卷管理系统中必不可少的一个环节，对于学校如何建立系统化教学管理系统，对于学生个人如何建立自己的试题库，对于教师如何系统分析管理学生成绩都有着重要的意义。

传统的试卷管理需要消耗大量的人力与物力。在一次考试结束后，学生要自己手动整理一张试卷，包括哪些题目回答错误，这套试卷的正确率有多少，与以往的考试相比有没有提高。同时教师则需要更多的时间整理学生的成绩，需要手动统计全班所有同学的信息，再进行统计分析，耗费了教师大量的时间和精力，试卷整理周期过长。

为了避免传统阅卷的不足，在国家教育部门的推荐及要求下，网上阅卷已经越来越多的被人们使用，并得到大力推广。在 2006 年度的高考评卷中，已经有近 20 个省市开始使用网上阅卷系统。对于网上阅卷的信息整理就显得相对简单了。但是现在大多数考试包括学生平时的小测验，模拟考试等等还是采用人为手工阅卷的方式。这些考试的信息也很重要。所以就需要一个试卷信息的录入及管理系统。

采用一个试卷结果信息的识别，录入及管理系统主要有以下几点优势：

1) 可以显著降低人为整理试卷信息所要花费的人力，物力。帮同学节省出更多的时间来学习新的内容；帮老师省出更多的时间来提高教学质量。
2) 改变了传统的试卷管理模式，可以帮助学生或老师分科目，分类型地有效整理试卷信息，方便信息的提取。
3) 方便成绩的统计和分析，借助一个试卷识别和管理系统，学生可以纵向

分析自己的考试历程，比如在这一个学期的某门科目中，包括平时的小测验和考试，自己答题的准确率有了什么样的变化，在哪种题型中得到了提高，哪种还有不足。而教师可以横向分析自己班级上的学术成绩，如哪种题型自己的学术掌握的还不是很好，哪种题型有了较大的改善。借助这些统计信息，教师就可以更合理的制定教学方案，提高教学水平。

4) 随时查看试卷信息。有了自己的数据库，学生和教师不用再保存之前的纸质试卷，可以随时随地地查看历史成绩,方便快捷。真正实现了现代化数字教育。

1.2 国内外研究现状

1.2.1 试卷信息的提取和版面分析

近年来在试卷信息的提取方面国内主要分为两个方向。第一个是已经成熟投入使用的基于光学标记的阅读机 OMR(Optical Mark Reader)[1]，它的主要原理是根据光电转换的原理运行，将原本的光信号转化为电信号，再采用 A/D 转换，将电信号转换为数字信号，进而再在数字信号方面上使用相应的处理，从而完成了从答题卡上的涂点到符号的采集转换过程，并且同步存储在计算机中。因为 OMR 的性能高效，准确性也比较高，所以 OMR 在国内教育市场上获得广泛应用。但是同时 OMR 也存在一些不足：不如适应性相对单一，只能针对答题卡和 2B 铅笔的样本；使用要求也较高，在答题时，需要考生严格按照标准进行答题，填涂，否则无法进行识别；在硬件方面不仅仅需要专门的光标阅读器，也需要大量的答题卡，增加了成本消耗；如果需要复查阅卷，还需要调出原本的答题纸，增加了复查工作的复杂度。

另一个方向是基于图像处理上的试卷识别和评阅系统。这种系统基本上将扫描的试卷图像经过图像预处理，将得到的图像保存在计算机中。在通过版面理解将客观题，主观题等不同答题区域分割出来，然后分发给不同的阅卷老师进行批阅，最后由系统回收并进行核分和校验的工作。此类方法是一种新的技术革新，对数字化试卷管理系统的建立有着很大的帮助。但是这种方法还尚且存在一定的技术缺陷，在实现手写字符识别方面以及版面理解和分隔方面都存在着很大难度。这使得光学标记阅读机 OMR 暂时还很难被替代，为了辅助阅卷系统的实现还需要做进一步的研究。

通过上面的介绍我们知道版面分析与版面理解是一个很大的难题，这种技术伴随着计算机的发展而逐渐成熟起来，它的研究到现在已经有几十年的历史。目前，版面分析技术的研究，主要是对于不同领域的图像的版面，做有针对性的分隔和理解，整体不具备通用性。目前，国内外许多大学和研究机构，都在这个领域取得了研究成果，这些机构也是根据不同的版面进行分类，有针对地对版面信息进行分隔处理。在国外方面，这些机构有：加拿大机器智能研究中心研发的字符识别系统，版面理解分析以及自然语言理解；德国 DFKI 研究中心对版面分析及理解进行研究；美国 Nevada LasVegas 大学的智能系统实验室以及 Washington 大学的智能研究工作室，开发除了基于 CD_ROM 版面数据中心，该数据重心主要应用基于 OCR 的版面分析和理解算法[1]，它包含了常见图像页面中的对象的信息，如文本，字符，线条等，可以有效的对版面分析及理解算法进行合理分析与评价。

在国内方面也有一些相关的研究与清华，汉王集团推出的汉王尚书，表格自动录入系统；蒙丹 OCR 字符扫描和识别系统，清华紫同 ZT-OCR，清华晨光 OCR，复旦大学的字符识别机。但是在试卷版面的分析与研究方面，基本都需要人为提供相关信息如版面样式，类型等信息，这些问题还需要等待进一步研究才能得到解决。

综上所述版面理解是一项很复杂的工作，不仅仅涉及到字符等对象的识别，还包括语义理解等多个层面。中西方都在版面理解方面做了大量的研究，并且也推出了一定可以实际运用的系统，其中中文版面的分析还不是很成熟，以及对版面的分隔也需要针对不同样式的试卷做有针对性的设计。这些都有待做进一步的研究。

1.2.2 脱机手写字符的识别

我在这次项目中的主要负责工作的一个方面是脱机状态手写字符的识别。字符识别是机器学习领域一个传统的研究方向。但是现在的研究成果，大多数都是放在手写数字的识别上，在手写对号 √和错号×的识别方面还没有人做过相关的研究，所以本文做的将是一个全新的工作。不过通过多年的研究，现在已经有许多文献和相关报告提供有关手写数字的研究方案，这些方法在很多领域也得到了应用，如汽车车牌号的识别，订单扫描中订单编号的识别。这些工作都可以为本文工作的开展提供丰富的经验。

目前手写数字的识别方法主要有以下几种：

1.基于 K-邻近算法的脱机手写字符识别[2]。该方法主要是通过预处理操作后，基于 K-邻近算法对图像进行分类处理。K-邻近算法是以待识别字符的平均值作为基准，这个基准的选取基于一个构造函数，如对特征向量取均值，然后取待识别的字符的均值，然后判断其与哪个类最近。这种识别方法相对较为简单，实现起来较为简单，效率较高。但是精度比较低，只以均值作为标准，所以对均值的选取就变得尤为关键。这种方法的应用在邮政编码识别，然后对邮件进行分类等多个领域。

2.基于 HMM 的脱机字符识别[3]。隐式马尔科夫模型 HMM 是一种机器学习模型。这种方法也是通过对图像的预处理操作后，构造传统 HMM 模型对手写字符进行分类。这种方法在马尔科夫模型状态数较低时效果不理想，但是当模型状态数增加时(如状态数=20)识别率也会有相应的提高，但是这会使识别时间也相应增加。所以为了达到更高的精度需要更高的时间复杂度。而且不同手写字符的形变会大大增加误识率。这种方法的应用有银行票据中字符的识别等。

3. 基于 BP 神经网络的手写字符识别[4]。BP 神经网络是模拟人类的神经结构，如神经的传导，在神经网络中通过输入层，中间的隐层，最后到输出层得到识别结果。这种方法相对于其他算法，训练过程相对繁琐，需要不断调整隐式网络中各层节点的输出值。并且需要用到大量的样本来训练网络。基于 BP 神经网络的识别算法能带来相对高的精度，但是 BP 神经网络在识别时是所有训练算法中需要耗时较长的。这种方法应用在车牌中数字的识别等领域。

下表是对不同手写字符识别方法的比较：

识别方法	优点	缺点	应用
基于 K-邻近算法的识别	相对简单，效率高	精度低，均值的选取难点大	邮政编码识别
基于 HMM 模型的识别	精度较高(在状态数高的时候)	识别时间随状态数增加	银行票据中字符的识别
基于 BP 神经网络的识别	识别准确率高	模型复杂，训练和识别的时间长，需要大量的样本	车牌中数字的识别

表 1-1 不同手写字符识别方法的比较

上述算法中基于 BP 神经网络的算法可以达到较高的 94.6%的平均识别率和 45ms 的平均识别时间。HMM 算法可以达到 93.7%的平均识别率和 35ms 的平均识别时间。但是对于试卷系统中的手写字符识别则需要更高的精确度来保证不

误判任何一道试题，并且需要做到更快的识别时间。

而本文所采用的是支持向量机 SVM。SVM 方法是学习的泛化能力的体现属于机器学习中监督学习的范畴。由于我们所需要识别的是手写对号 √和错号×，较其他字符识别率较高，一个合理训练的 SVM 分类器理论上可以满足我们的要求。关于 SVM 的学习国内外已经都做过许多研究。因此具有坚实的理论基础。如 Anthony et al.(1999)等人给出的关于硬邻持向量学习误差的严格理论界限，Shawe Taylor(2000)和 Cristianini(2000)也给出了类似的关于软邻域 SVM 和回归情况下的误差界限；Smola(1998)和 Schoelkopf(1999)提出了支持向量机一般意义下的损失函数数学描述; Girosi(1990)，Smola(1998)，Schoelkopf(1999)等讨论了正则化网络和支持向量机的关系。

其实 SVM 在实际应用方面的第一个领域就是手写数字的识别。最初应用在美国的邮政局的手写邮编识别上。SVM 为传统图像和字符识别领域提供了新的工具，也在多个领域得到了广泛应用，如模式识别和函数拟合领域。在本文中结合特征提取方法，基于 SMO 算法的训练和参数选取，SVM 可以表现出比 BP 神经网络更高的识别率和更短的时间复杂度。

1.3 本文的讨论内容和重点

本文在第二章首先概述了项目的技术流程。及每部分内容的简要介绍。然后介绍了我的工作重点和工作流程。

在第三章介绍了特征向量的提取方法，初步使用在手写数字领域常用的 13 点特征提取法。然后通过分析手写字符(对号 √，错号×)的特点，设计了基于笔画密度特征，重心及重心距特征，傅里叶变换特征，字符轮廓特征，投影特征等 102 维特征向量。

第四章介绍了 SVM 的基本原理和 SMO 算法的推导及实现。选取了多种核函数进行测试，最终选取了基于 RBF 内核的 SVM 分类器。使用网格搜索和交叉验证的方法确定了分类器的最优参数。

第五章里搜集了大量不同手写字符的样本对特征选取，核函数和不同算法进行比较，分析，并对本项目中使用的分类器进行评估和性能测试。

第六章介绍了学生试卷识别和管理系统包括其基本界面和实现的基本功能。

第2章 项目实施方案

2.1 项目整理技术流程

我和同组成员的项目是在图像处理和 SMO 算法 SVM 的基础上，实现一个试卷识别，录入和管理系统。实现了图像处理，手写字符的脱机识别，学生个人试卷管理系统的建立等部分功能，具体流程如下：

1) 对图像进行获取，然后将获取到的图像经过灰度化，二值处理，降噪，倾斜矫正等图像处理，得到利于进行版面分析的图像；利用 OCR 技术对试卷进行版面分隔，分割出每道试题。然后利用二次投影法对待识别字符区域进行定位，提取手写字符图像，最后分隔和归一化得到带提取手写字符。
2) 结合手写字符(手写对号 √ 和错号×)的特点，设计多种特征提取方法，通过实际调试，比较选出最能代表字符特性的一组特征，组成特征向量。
3) 特征向量的归一化。
4) SVM 的 SMO 算法的实现。通过训练，比较，分析其与传统二次规划算法的训练速率，准确程度上的提高。
5) 核函数的确定，及参数的优化。比较多种核函数如线性核函数，多项式核函数，径向基函数神经核函数等，最终确定所用核函数。再通过网格搜索(Grid-Search)和交叉验证(Cross-Validation)选取出最优的惩罚项 C 和参数 g。
6) 搜集大量样本进行训练，对分类器进行评估。横向比较使用不同内核的差异，纵向比较选取不同参数的差异。给出具体学习效率，与识别率。
7) 学生个人数据库的建立及管理。

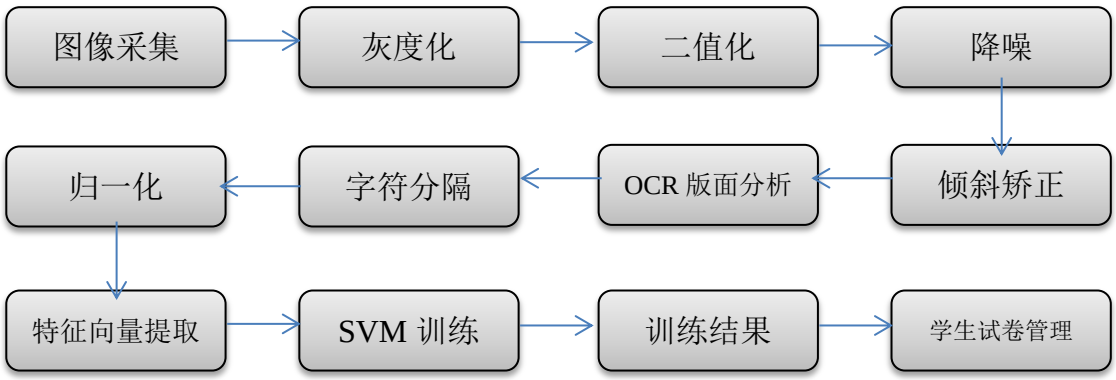


图 2-1 项目技术流程

2.2 各技术环节简要介绍

2.2.1 图形提取及预处理技术简单分析

2.2.1.1 灰度化

我们所采集的图片是全彩的 JPG 格式的图像，我们首先需要将彩色图像转换为灰度图像。灰度化后，图像内的信息只反映图像的亮度。亮度信息的值域为[0，255]。此时每个像素的 RGB 值都是相同的。

图像灰度处理的方法一般如下式：

灰度值=0.299R+0.587G+0.144B

但是结合实际中的应用还需要进行具体的技术改进，将在下一小节中与二值化处理一起介绍。

2.2.1.2 二值化

无论是对版面的分析和待识别字符的提取，都需要对试卷进行不同的二值化操作。对于版面分析，我们可以使用现在研究中常用的二值化方法，通过设定阈值，将图像转化为 0-1 矩阵。

但是当今的图像识别相关研究所针对的都是扫描版的图像，这就从一定程度上避免由于光线，明暗等环境因素带来的影响。但是为了简便用户的使用，提高可行性，我们针对的是拍照的图像。这就为项目带来了不小的挑战。

所以对于手写字符的提取部分的二值化操作，我们将 RGB 颜色体系投影到 HSV 颜色体系中，再将 HSV 色谱中红色的区域通过表达式提取出来，来获取图像中待识别的部分。

2.2.1.3 降噪

处理图像需要对噪声进行消除。这些噪声有可能来源如传输，扫描或拍摄的过程中，这些干扰是难以避免的。消除噪声对我们版面分析的位置选取，以及手写字符识别的精度都有很大的提高。

这里我们使用的是二值图像的黑白点噪声滤波。基本思想是选取某一指定像素，然后选取其周围像素的平均值，如果低于某一阈值则说明该点为噪声点。

2.2.1.4 基于 OCR 的版面分隔

使用 OCR 技术对版面进行分析。首先对试卷进行每行的分隔，对每行的字符进行 OCR 处理，分析开头字符，提取题目编号。对属于同一编号的行进行整合，从而分割每道试题。

2.2.1.5 字符的分隔和提取

我们使用的是投影法，即提取出我们关心的手写对勾√或错误×字符。为此，我们可以先进行水平投影，在 x 轴中自左至右进行扫描遇到第一个黑像素点进行记录，这样可以记录出第一个字符的宽度，然后仅针对这个宽度内的字符进行 y 轴投影，接着在 y 轴扫描找出的黑色像素区域就是字符的高度，综合两个坐标轴反馈的信息，我们可以得出第一个字符的所在区域，进而进行提取。

2.2.1.6 归一化

不同人的手写习惯不同，写出的字符大小也有不确定性。所以为了后续的操作，特征选取的研究，和识别准确率的提高，我们采用等比例缩放的方式，将提取到的图片归一成同样的大小：50*50 像素。

2.2.2 基于 SMO 算法训练的 SVM 分类器

2.2.2.1 特征向量的选取

特征向量的提取是本文一个十分关键的工作方面。特征选取对于图像识别的结果有很大影响，如何选取特征向量，使其可以准确反映不同字符的特征，并且具有抗干扰性，可靠性和独立性。我先使用在传统数字识别中常用的 13 点特征向量提取算法，然后分析其识别率。接着设计了包括傅里叶变换，轮廓特征，投影特征等，能够更好地反应字符的信息，从而提高识别精度。

2.2.2.2 基于 SMO 的 SVM 分类器

SVM 分类器传统的训练算法需要解决二次规划 QP 问题，这往往需要花费大量的时间，也需要借助第三方二次规划工具。SMO，序列最小化算法，在每次调整过程中只选取两个变量，同时固定其它变量，将原问题简化为对两个变量的优化问题。

2.2.2.3 核函数的选取和参数调整

核函数的选取一般有两种方法：第一种是利用相关研究和文献预先确定所使用的核函数；第二种是分别适用不同的核函数，根据训练集和测试集的训练结果反馈确定最终使用的核函数。由于样本的特异性，我采用后者通过搜集数据，仿真实验，最后确定了 RBF 内核。

然后再通过网格搜索和交叉验证的方式，确定了 RBF 内核参数γ，和惩罚项 C。最后搜集大量的样本进行测试和验证。这些部分再后文中都会有详细的介绍。

2.3 具体工作介绍

本项目由我和同一课题组的另外一名学生共同完成。我的重点工作在于脱机手写字符的识别。手写字符的联机识别，可以参考字符书写时的动态信息如时间，笔划顺序等，所以识别起来相对简单，也在许多方面取得了研究进展。而脱机字符识别，针对手写字符，因为手写字符因不同人差异较大，受周围环境影响也较大所以被认为是模式识别领域的最后一个堡垒。

我的主要工作集中在：1.特征向量的提取；2.SVM 分类器的训练，核函数的选取和最优参数确定；3.学生试卷识别和管理系统，包括功能设计，界面设计以及存取功能。

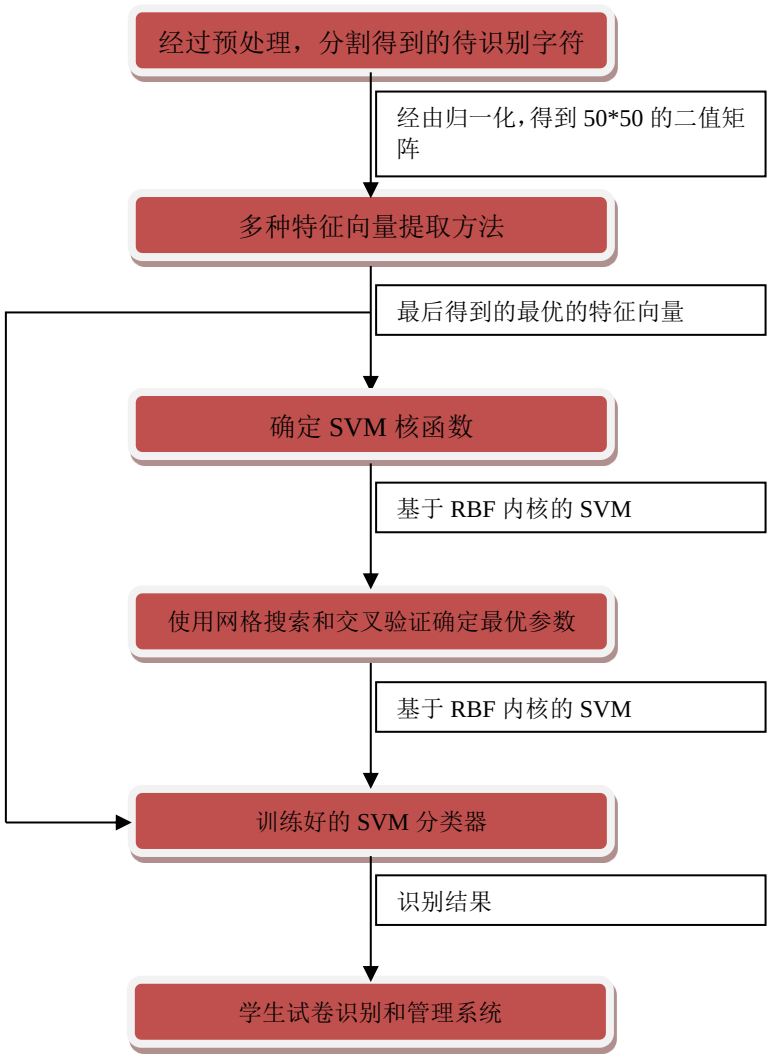


图 2-2 具体工作流程

2.4 本章小结

本章介绍了项目所要完成的整体工作及其技术流程，分别介绍了每个流程中的各个技术环节以及所要实现的预期成果。使读者从整体上对项目有一个整体的认识和概念。还介绍了项目的部署，我的工作内容，项目整体的整合以及表现形式。

第3章 特征提取和特征选择

3.1 特征提取概述

特征向量的提取对识别结果有很大的影响，相似的特征会使得样本空间变得混乱而不可分。所以怎么选择特征使得选取的特征具有可分辨性，可靠性，独立性，鲁棒性同时又要避免过多特征所带来的维数灾难[5]。具体到本项目上，特征的选取要对于手写对号√和错号×要取区别比较大的值，这样才可以从不同的样本中做出区分。特征的可靠性是指特征要能反应待识别字符的信息，如轮廓，笔画数等等。特征的独立性是指不同的特征选取出来应该互不相关，这样就可以避免信息冗余。特征的鲁棒性是指我们应该选取那些抵抗干扰，噪声能力较强的特征。特征的维数灾难是指，选取的特征个数过多，其模型往往会变得异常复杂，不尽加剧了算法的时间和空间复杂度，也降低了其推广能力。更重要的是训练所必要的样本数量也会根据特征选取的多少呈指数增长。如图

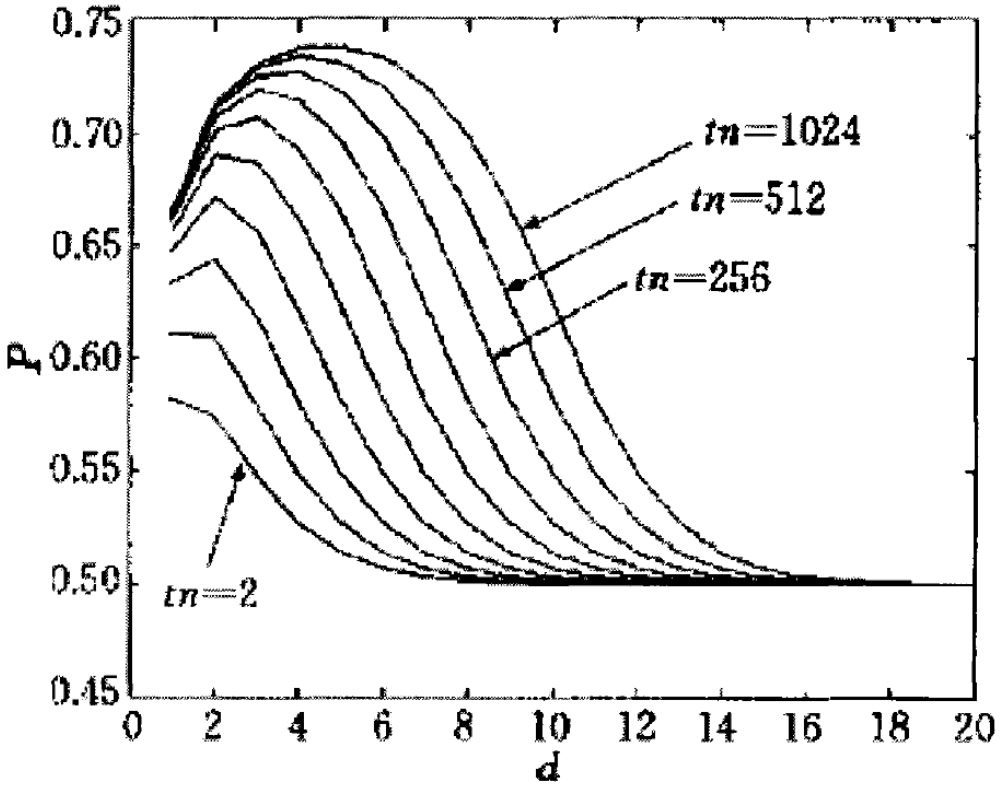


图 3-1 平均识别率 d 与特征维数的关系(出自手写体数字识别中的特征提取和特征选择研究)

上图是 Bayes 算法的识别率 d 与向量维数的关系。tn 为样本容量，d 为特征

维数。当向量维数增加时，我们需要相应地增加样本集的数量才能达到更好的识别效果，而样本空间数量的增加是呈指数集的。另外这个规律也适用于朴素贝叶斯，高斯等分类器，当然也包括我们的 SVM 分类器。

我所得到的原始特征是 50*50 的二值矩阵，也就是 2500 维的特征向量，显然这是不合理的。因为这种原始特征肯定不能直接反应对象的本质而且会带来维数灾难。一种方法是通过 PCA 降维的方法，降低向量的维度。第二种是从 50*50 的二值矩阵中通过分析，选取等操作提取出能反应字符信息的向量。一般分为结构特征提取和统计特征提取两种模式。

3.1.1 结构特征提取

结构特征选取是一种较为直接的方法[6]。它把字符拆分成不同部分，笔画和字段。再从字段到笔画到部分按级分析字符的组成结构，对各个部分的数量，位置等关系进行判断从而判断整个字符。这种有许多相关的文献，对于基于笔画和字段的手写字符识别结果较为理想。

虽然人们对于手写对号 √ 和错号×的差异比较大，每个人会有自己不同的书写习惯，但是以笔划，位置关系等拓扑结构是不会改变的。人识字的时候就是抓住了这样一个特征。同样地我们也可以训练计算机去进行这样一个识别过程。

3.1.2 统计特征提取

统计特征提取，是从模式识别的观点上来看，人在进行物体识别的过程时，是一种从物体到记忆中的知识库的映像，从而得到识别结果[6]。这个过程就是模仿人脑对物体的识别：我们在识别物体识，会从记忆库中进行搜索，对物体进行映像，然后找出相应的匹配模式。对于计算机来说，具体上由两个步骤完成的：首先，选取适当的特征来描述物体，即完成由$x \to f(x)$的函数映射；接着计算机执行某种由$f(x) \to c(x) = X$的映像。上述过程是统计模式识别的一般方法，需要做的工作就是特征的提取和映像函数的设计。对于特征的提取，我们则需要针对不同的目标进行具体分析，从字符的形态，轮廓等方面找出能区别字符的特性，找出能帮助识别字符的，有独立性，特异性的特征。这需要一定的思考和参考相关领域文献等先验知识。

3.2 13 点特征提取

13 点特征提取方法是一种新的针对手写数字的特征提取方法[7]。其适应性较强，可以针对字符在一定程度上的倾斜和偏移。我认为可以尝试应用于手写对号 √和错号×的识别中。

13 点特征提取的总体思路就是：首先将字符左右 2 等分，然后再竖直上 4 等分，分成 8 个相等的区域，统计每一个区域上像素点的数量，组成 8 个特征；然后计算水平方向中间 3 等分处的两行与字符相交的黑像素个数作为 2 个特征；计算垂直方向中间 3 等分处的两列与字符相交的黑色像素个数作为后两个特征；最后统计所有像素的和作为第 13 个特征。

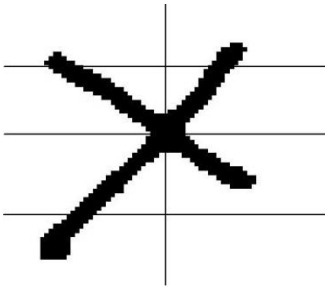


图 3-2 8 个区域

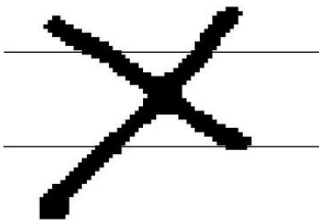


图 3-3 水平 3 等分

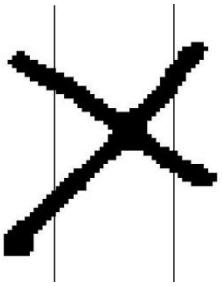


图 3-4 竖直 3 等分

从实验结果来看 13 点特征提取法可以从一定程度上反应字符的特征，比如在第一个特征(8 等分的第一个区域)中，错号的特征值要一般大于对号的特征值。再入竖直 3 等分线与字符交点的两个特征中错号的特征值也要远大于对号，因为对号只有单一的轨迹。但是 13 点特征提取法受不同手写方法影响和环境影响也较大。比如不同类型的笔或不同力度写出来的字，笔划的像素宽度也会不一样，这样一来在从数值上进行统计就有可能出现误判。比如一副像素较宽对号的竖直 3 等分特征会近似于一副像素较窄的错号的竖直 3 等分特征。这样就会从一定程度上增加误识率。

虽然 13 点特征提取法会存在一定问题，但是从结果来看还是可以做到 90%以上的识别率，这点会在第四章项目结果中来具体分析。但是为了达到 100%的识别效果我们还需要跟优秀的特征选取。

```
function e = GetFeature(d)
%根据13点特征提取法返回图片特征向量
count=13;%选取特征个数
feature=zeros(1,count);%特征
[m,n]=size(d);
%B=blkproc(Image,[m n])
%水平分成四份,垂直分成两份,分别统计这8个区域的白像素的个数,得到8个特征
low = 4; % 水平块数
```

```
row=2;%垂直块数
ch = m/low; cw = n/row;
t1 = (0:low-1)*ch + 1; t2 = (1:low)*ch;
t3 = (0:row-1)*cw + 1; t4 = (1:row)*cw;
for i = 1 : low
    for j = 1 : row
        temp = d(t1(i):t2(i), t3(j):t4(j));
        %subplot(low, row, (i-1)*row+j);
        %imshow(temp, 'border', 'tight');
        feature((i-1)*row+j)=sum(sum(temp));%统计这8个区域的白像素的个数,得到8个特征
    end
end
%水平和垂直各划两条线把水平和垂直分割成三分，统计这四条线穿过的白像素个数,得到4个特征
num=2;%划分数
for k=1:num
    feature(low*row+k)=sum(d(round(k*m/(num+1)),:));%水平线白像素和
end
for k=1:num
    feature(low*row+num+k)=sum(d(:,round(k*n/(num+1))));%垂直线白像素和
end
feature(count)=sum(sum(d));%图像全像素和
e=feature;
```

3.3 改进特征向量提取

针对 13 点特征不能很好反应字符特点，并且抗感染能力差，识别率较低等问题。我针对手写对号√和错号×的特点，抽取了包括傅里叶系数特征，笔划密度特征，重心及重心矩特征等，共 102 维特征。

3.3.1 笔划密度特征

笔划密度特征是：以不同方向扫描字符，计算扫描线和笔划相交的次数，进而形成笔划密度特征向量。在本项目中我将 50*50 像素的字符以行和列为标准各 10 等分，在水平方向上的 9 个等分线中与黑色像素的交点数提取 9 个特征；在垂直方向上的 9 个等分线中于黑色像素的交点数提取 9 个特征。总共有 18 个特征。

分析笔划密度特征，我们可以看到，这种特征可以对不同人的手写习惯，和图像的传输过程中带来的畸形形变作出较好的适应。但是会受到噪声带来的影响。如在扫面线上存在噪声点，则会使该出的特征值出现混淆。其实与 13 点特征提取中的水平和竖直特征提取类似，但是不同的是增多了隔行/列扫描的次

数，这就从一定程度上减少了不同人书写习惯不同，或采样不充分带来的形变。

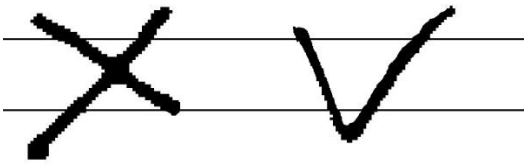


图 3-5 13 点特征提取中的水平特征对比

从上图对比我们可以看出一个书写稍不规范的对号和正常错号，在进行 13 点特征提取法中的水平特征提取时，水平扫描线都经过两次笔划，尽管在细微方面还是会有些不同，但是一定程度上可以当做近似特征，没有区别性(当然在垂直扫描中不具备这种问题)。但是如果增加扫描的次数，就一定会有若干维水平扫描的特征能明显揭示不同点。从数值上来看这些特征t一般$t_correct_i \approx 0.5 \times t_error_i$。

```
%%
%笔画密度特征50*50的样本隔10行提取一个，
%行9个，列9个共18个
num = 9;
temp = [];
for k = 1:num
    temp = [temp,sum(pic(round(k*m/(num+1)),:))];%水平线白像素和
end
for k=1:num
    temp = [temp,sum(pic(:,round(k*n/(num+1))))];%垂直线白像素和
end
feature = [feature, temp];
```

3.3.2 投影特征

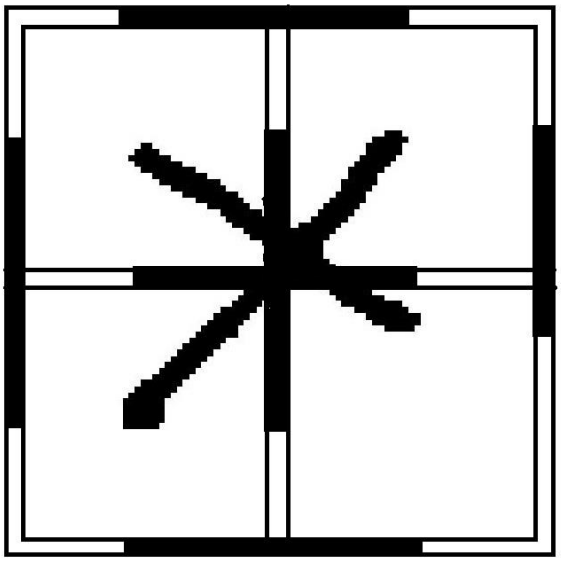


图 3-6 字符投影特征示意图

如图 3-6 所示，将字符映射到坐标系的四个象限中，包括 x 轴的正负，y 轴的正负和周围边线，一共组成了 12 条线段。随机选择一个像素点，将该点分别映射到所在象限的 4 条边界线上。最后分别统计所有边界线上的像素长度，组成 12 维特征向量。

投影特征的具体方法如下图：

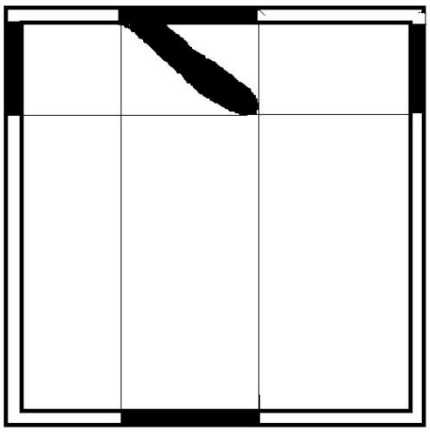


图 3-7 局部投影特征

这是一幅字符点阵的第四象限局部特征，对于点阵中的每一个点被投影到边线上时，激活该边界上的某一个 bit 或某些 bit。当这些 bit 被激活时取值为 1，

没有被激活的 bit 取值为 0。当所有投影都完成时，计算一条边界线上的 1 的数目即为特征值。

投影特征的选用有以下几点原因：第一点，投影特征的计算简单，仅需要对字符整体进行扫描，即可将点投影到边界上。第二点，投影特征通过四个象限的分隔可以反映字符的内部特征。不同字符之间的特征差异比较明显，可以作为分类的细特征。第三点，考虑大脑的感知模型，我们哺乳动物在用视线捕捉物体时，物体在头脑中的影响也存在类似的边界线，我们会自动联系物体投影到边界线上的特征。所以这也符合人类认知物体的过程。

```
%%
%投影特征，将一个字符点阵划分为4个像素区域，共12个边界，
%对每个点向所在象限的4个边界投影
%12条边界上的投影长度作为12个特征
mid = m/2;
a = zeros(1, mid);
b = zeros(1, mid);
c = zeros(1, mid);
d = zeros(1, mid);
e = zeros(1, mid);
f = zeros(1, mid);
g = zeros(1, mid);
h = zeros(1, mid);
x = zeros(1, mid);
y = zeros(1, mid);
k = zeros(1, mid);
l = zeros(1, mid);
for i = 1:m
   for j = 1:n
      if(pic(i,j))
         if(i <= mid) && (j <= mid)
            a(j) = 1;
            l(j) = 1;
            h(i) = 1;
            x(i) = 1;
         elseif(i > mid) && (j <= mid)
            l(j) = 1;
            f(j) = 1;
            g(i - mid) = 1;
            k(i - mid) = 1;
         elseif(i <= mid) && (j > mid)
            x(i) = 1;
            c(i) = 1;
            b(j - mid) = 1;
            y(j - mid) = 1;
         else
            k(i - mid) = 1;
            d(i - mid) = 1;
            y(j - mid) = 1;
            e(j - mid) = 1;
```

```
            end
        end
    end
end
feature = [feature, sum(a), sum(b), sum(c), sum(d), sum(e), sum(f),
sum(g), sum(h), sum(x), sum(y), sum(k), sum(l)];
```

3.3.3 重心及重心矩特征

不同字符的笔划分部和位置都有所不同，所以我们可以根据字符的重心和各个象限的重心矩位置来找出能区分字符的几何特征。令$f(m,n)$表示二值矩阵中第 m 行，第 n 行的值。则字符的重心位置$(\overline{m},\overline{n})$为：

$$\overline{m}=\sum_{n=1}^{N}\sum_{m=1}^{M}f(m,n)\frac{m}{\sum_{n=1}^{N}\sum_{m=1}^{M}f(m,n)}$$

$$\overline{n}=\sum_{n=1}^{N}\sum_{m=1}^{M}f(m,n)\frac{n}{\sum_{n=1}^{N}\sum_{m=1}^{M}f(m,n)}$$

其中$m=1,2,3,\ldots,M$表示矩阵的行，$n=1,2,3,\ldots,N$表示矩阵的列。这样$(\overline{m},\overline{n})$就定义了矩阵的重心位置。

我们还是把整个字符分解成 4 个象限，类似地我们可以求出 4 个象限的重心位置$(\overline{m_1},\overline{n_1}),(\overline{m_2},\overline{n_2}),(\overline{m_3},\overline{n_3}),(\overline{m_4},\overline{n_4})$：

$$\overline{m_1}=\sum_{n=1}^{N/2}\sum_{m=1}^{M/2}f(m,n)\frac{m}{\sum_{n=1}^{N/2}\sum_{m=1}^{M/2}f(m,n)}$$

$$\overline{n_1}=\sum_{n=1}^{N/2}\sum_{m=1}^{M/2}f(m,n)\frac{n}{\sum_{n=1}^{N/2}\sum_{m=1}^{M/2}f(m,n)}$$

$$\overline{m_2}=\sum_{n=1}^{N/2}\sum_{m=M/2}^{M}f(m,n)\frac{m}{\sum_{n=1}^{N/2}\sum_{m=M/2}^{M}f(m,n)}$$

$$\overline{n_2}=\sum_{n=1}^{N/2}\sum_{m=M/2}^{M}f(m,n)\frac{n}{\sum_{n=1}^{N/2}\sum_{m=M/2}^{M}f(m,n)}$$

$$\overline{m_3}=\sum_{n=N/2}^{N}\sum_{m=1}^{M/2}f(m,n)\frac{m}{\sum_{n=N/2}^{N}\sum_{m=1}^{M/2}f(m,n)}$$

$$\overline{n_3}=\sum_{n=N/2}^{N}\sum_{m=1}^{M/2}f(m,n)\frac{n}{\sum_{n=N/2}^{N}\sum_{m=1}^{M/2}f(m,n)}$$

$$\overline{m_4} = \sum_{n=N/2}^{N}\sum_{m=M/2}^{M} f(m,n)\frac{m}{\sum_{n=N/2}^{N}\sum_{m=M/2}^{M} f(m,n)}$$

$$\overline{n_4} = \sum_{n=N/2}^{N}\sum_{m=M/2}^{M} f(m,n)\frac{n}{\sum_{n=N/2}^{N}\sum_{m=M/2}^{M} f(m,n)}$$

最后取点阵的重心位置$(\overline{m},\overline{n})$为两个特征，再取 4 个象限距离重心的重心矩(距离重心的矢量距离，包括负距离)$(\overline{m}-\overline{m_k},\overline{n}-\overline{n_k},),k=1,2,3,4$，为 8 个特征，共组成 12 维特征。

重心特征反应了重心的相对位置，比如错号的重心相对居中，而对号的重心相对偏小；四个象限重心相对矢量位置反应了与整体字符分部的关系，可以得出四个象限偏离重心的关系。很好地反应了字符的形态分部。

```
%%
%重心及重心距特征
%求出重心的二维特征
%及4个象限关于重心偏离的重心距二维特征

%首先求出整个图像的重心
x = ones(m,1)*[1:n];
y = [1:m]'*ones(1,n);
area = sum(sum(pic));
meanx = sum(sum(pic.*x))/area;
meany = sum(sum(pic.*y))/area;
%第一象限
temp = pic(1:mid, 1:mid);
x = ones(mid,1)*[1:mid];
y = [1:mid]'*ones(1,mid);
area = sum(sum(temp));
if area==0
    area = 1;
end
meanx1 = sum(sum(temp.*x))/area;
meany1 = sum(sum(temp.*y))/area;
%第二象限
temp = pic(mid+1:m, 1:mid);
x = ones(mid,1)*[1:mid];
y = [1:mid]'*ones(1,mid);
area = sum(sum(temp));

if area==0
    area = 1;
end
meanx2 = (sum(sum(temp.*x))/area) + mid;
meany2 = sum(sum(temp.*y))/area;
%第三象限
temp = pic(1:mid, mid+1:n);
x = ones(mid,1)*[1:mid];
y = [1:mid]'*ones(1,mid);
```

```
area = sum(sum(temp));

if area==0
   area = 1;
end
meanx3 = sum(sum(temp.*x))/area;
meany3 = (sum(sum(temp.*y))/area) + mid;
%第四象限
temp = pic(mid+1:m, mid+1:n);
x = ones(mid,1)*[1:mid];
y = [1:mid]'*ones(1,mid);
area = sum(sum(temp));

if area==0
   area = 1;
end
meanx4 = (sum(sum(temp.*x))/area) + mid;
meany4 = (sum(sum(temp.*y))/area) + mid;
feature = [feature, meanx, meany, (meanx - meanx1), (meany - meany1),
(meanx - meanx2), (meany - meany2), (meanx - meanx3), (meany - meany3),
(meanx - meanx4), (meany - meany4)];
```

3.3.4 傅里叶变换特征

对某一手写字符来说，图像的边界一般是由一条封闭的曲线组成的。在边界上人选一个点，让其顺时针移动，这个轨迹所构成的是一个周期函数。将这个周期函数展开成傅里叶级数的形式。显然，边界的信息可以用傅里叶级数中的一系列系数来表示，我们可以取这些系数作为特征向量来区别不同字符的形状和轮廓信息。我们将原函数展开到很高的阶数时，几何可以完全重构出图像原本的轮廓和几何形状[8]。

傅里叶变换是一种在图像处理中非常常用的二维正交变换，图像的频率是表征图像中灰度变化剧烈程度的指标，是灰度在平面空间上的梯度。如：广阔的沙漠图像是一片色调变化不明显的区域，这种区域在频谱中呈现低的频率值；而对应的色调变化多变丰富的森林，在频谱中呈现的是较高的频率值。从宏观上来看傅里叶变换将图像从空间域变化到了频率域，从另一方面来说就是讲图像的灰度函数变换成了图像的频域分布函数。相反地，逆变换是将图像从频域函数变化回灰度梯度函数。在频率函数中，低频的区域表示了图像的边缘的变化强度和方向。而手写体字符一般可以用封闭轮廓曲线来表示，我们通过找到傅里叶变换后的低频分量区域，也就找到了可以反映字符轮廓的离散量，可以将这些离散量作为我们的特征向量。这些特征向量的值与相似字符的平移，旋转，位移和尺寸大小无关，具有很高的鲁棒性，抗干扰性，和独立性。

我们还是用$f(m,n)$二维图像点阵的值。对于(m=1,2,3…M, n=1,2,3…N)的图片其离散傅里叶变换为：

$$F(u,v)=\frac{1}{\sqrt{M}\sqrt{N}}\sum_{m=1}^{M}\sum_{n=1}^{N}f(m,n)e^{-j2\pi(\frac{mu}{M}+\frac{nv}{N})}$$

在上式中，u=1,2,3…M, v=1,2,3…N,称为空间频率。其中 j 为虚数单位，其逆变换为：

$$f(m,n)=\frac{1}{\sqrt{M}\sqrt{N}}\sum_{m=1}^{M}\sum_{n=1}^{N}F(u,v)e^{-j2\pi(\frac{mu}{M}+\frac{nv}{N})}$$

在本项目的图像处理中选取 50*50 像素的二维点阵，所以 M=N，所以$F(u,v)$用矩阵表示为：

$$\begin{pmatrix}F(1,1) & \cdots & F(1,M)\\ \vdots & \ddots & \vdots\\ F(M,1) & \cdots & F(M,M)\end{pmatrix}$$

从图像上来看，我们可以从傅里叶频谱上观察到亮度不一样的亮点，其意义就是指图像上的一点与其相邻点差异的强弱，也就是梯度的大小，称为该点的频率大小。我们可以观察图像中较亮的点亮度强，梯度大，而频率低，这些点往往可以反映字符的轮廓特征。

未经频谱移频的图像是四个角落低频，最亮；而频谱移频到原点以后，我们可以观察到图像的频域是以原点为中心，呈对称状展开的，图像的中心低频，最亮。我们可以提取图像中心处的频谱作为我们的特征向量。

综上所述，我将字符的二维点阵先经过二维离散傅里叶变换，再将直流分量移动到频谱中心。接着取傅里叶变换的实部和虚部计算频谱幅值，经过归一化后得到傅里叶频谱矩阵$\overline{F(u,v)}$。我们所取的特征向量为：

$$\begin{pmatrix}\overline{F(23,23)} & \cdots & \overline{F(23,29)}\\ \vdots & \ddots & \vdots\\ \overline{F(29,23)} & \cdots & \overline{F(29,29)}\end{pmatrix}$$

当然我们要去除中心的$\overline{F(26,26)}$这是图像的原点，频谱的中心，所有图像都为同一数值(归一化后为 255)，是冗余的特征值。

综上我们在傅里叶特征提取中，提取了 48 个特征向量。

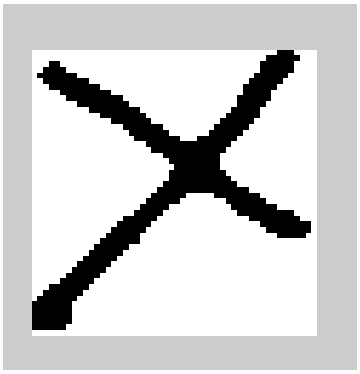


图 3-8 手写字符原图

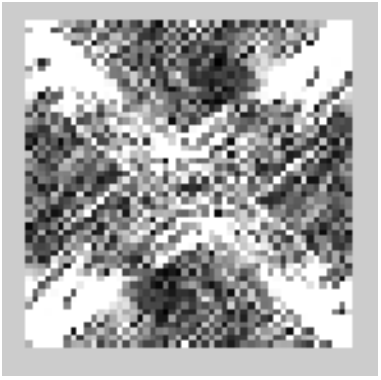


图 3-9 傅里叶变换后四角较明

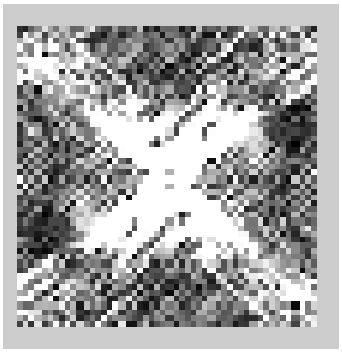


图 3-10 频谱移频到原点后，频谱中心叫亮，为低频域，可以提取特征向量

```
%%
%Fourier变换平移后提取图像中心原点附近低频区域
%除中心区域外48个特征点
fftI1=fft2(pic);                    %二维离散傅立叶变换
sfftI1=fftshift(fftI1);              %直流分量移到频谱中心
RR1=real(sfftI1);                   %取傅立叶变换的实部
II1=imag(sfftI1);                   %取傅立叶变换的虚部
A2=sqrt(RR1.^2+II1.^2);              %计算频谱幅值
A2=(A2-min(min(A2)))/(max(max(A2))-min(min(A2)))*225;%归一化
temp                                                              =
[A2(23,23:29),A2(24,23:29),A2(25,23:29),A2(26,23:25),A2(26,27:29),A2(
27,23:29),A2(28,23:29),A2(29,23:29)];
[temp, ps] = mapminmax(temp);
feature = [feature, temp];
```

3.3.5 字符轮廓特征

轮廓特征可以很好地反应字符的总体构成和轮廓信息。字符的轮廓特征包括字符的宽度，高度，高宽比，字符的左右边线变换等，可以很好反应字符在轮廓上的变化规律。

针对本项目中 50*50 像素的图片具体的提取方法如下：

首先我们记$L_P(k), R_P(k)$为点阵的第 k 行的左，右轮廓。

$$L_P(k) = j_0, j_0\text{满足}f(k, j_0) = 1, \text{且当}j < j_0, f(k, j) = 0$$
$$R_P(k) = j_0, j_0\text{满足}f(k, j_0) = 1, \text{且当}j > j_0, f(k, j) = 0$$

从直观上来看$L_P(k), R_P(k)$就是第 k 行中，第一个和最后一个出现黑色像素的位置。

1) 字符某一行的宽度定义为：

$$W(k) = R_P(k) - L_P(k)$$

k=1,2,3…M 表示行数。对于本项目中的手写对号 √ 和错号×，错号往往从宽度上能明显区别于对号。

2) 比率定义为：

$$\text{比率} = \frac{M}{W_{\max}}$$

$W_{\max}$ 是指字符的有效宽度，也是字符的最大宽度，$W_{\max} = \max(W(k))$,k=1,2,3…M。

利用比率也可以很好从宽度上区分对号和错号。

3) 字符有效高度

与字符的有效宽度同理，字符的有效高度可以表示为：$H_{\max} = \max(H(k))$,k=1,2,3…M。$H(k)$是字符第 k 列的字符高度，其求法与$W(k)$的求法相似，也是用该列最后一个出现像素的位置行坐标减去第一个出现像素的位置的行坐标。

4) 字符的宽高比定义为：

$$Ratio = \frac{W_{\max}}{H_{\max}}$$

5) 字符轮廓线特征：

我们定义字符点阵的左右边缘线的一阶有限差分别为：

$$L_{dif}(k) = L_P(k) - L_{P-1}(k)$$
$$R_{dif}(k) = R_P(k) - R_{P-1}(k)$$

然后我们定义如下特征：

字符轮廓线最大和最小值所在位置：

$$L_{max} = \{k | k = \max(L_P(k))\}$$
$$R_{max} = \{k | k = \max(R_P(k))\}$$
$$L_{min} = \{k | k = \min(L_P(k))\}$$
$$R_{min} = \{k | k = \min(R_P(k))\}$$

字符的左右边缘线的正负峰值：

$$L_{peak+} = \max\{L_{dif}(k)\}$$
$$R_{peak+} = \max\{R_{dif}(k)\}$$
$$L_{peak-} = \min\{L_{dif}(k)\}$$
$$R_{peak-} = \min\{R_{dif}(k)\}$$
$$L_{peak} = \left|L_{peak+}\right| + \left|L_{peak-}\right|$$
$$R_{peak} = \left|R_{peak+}\right| + \left|R_{peak-}\right|$$

字符的轮廓线特征能很好反应字符的边缘变化情况，如字符左轮廓线最小位置，手写对号的出现位置一般在字符中心左右(笔划开始的地方)，而手写错号的左轮廓线最小位置一般在字符的上下两端(每个笔划分别起始的地方)。包括轮廓线的正负峰值，都能体现字符的特征，并且有很高的独立性。

综上所述，共有 14 个特征，分别是字符的有效宽度，有效高度，字符比率，字符的宽高比，字符轮廓线左最大值，字符轮廓线左最小值，字符轮廓线右最大值，字符轮廓线右最小值，字符边缘线左正峰值，字符边缘线左负峰值，字符边缘线右正峰值，字符边缘线右负峰值，字符边缘线左正负峰值绝对值之和，字符边缘线右正负峰值绝对值之和。

```
%%
%字符轮廓特征
%包括字符有效宽度，有效高度，宽高比等14个特征
B = edge(pic, 'canny');
w = zeros(1,m);
Lp = zeros(1,m);
Rp = zeros(1,n);
h = zeros(1,n);
for i = 1:m
    lp = 100;
    rp = 0;
    for j = 1:n
        if(j < lp) && (B(i,j) == 1) && ((j == 1) || (B(i,j-1) == 0))
            lp = j;
        end
        if(j > rp) && (B(i,j) == 1) && ((j == n) || (B(i,j+1) == 0))
            rp = j;
        end
    end
    if(lp ~= 100) %有点，存在轮廓
        Lp(i) = lp;
        Rp(i) = rp;
        w(i) = rp - lp;
    end
end
wmax = max(w);%有效宽度
ratio = n/wmax;%有效比率
%求出字符有效高度
for j = 1:n
    lp = 100;
```

```
    rp = 0;
    for i = 1:m
        if(i < lp) && (B(i,j) == 1) && ((i == 1) || (B(i-1,j) == 0))
            lp = i;
        end
        if(i > rp) && (B(i,j) == 1) && ((i == m) || (B(i+1,j) == 0))
            rp = i;
        end
    end
    if(lp ~= 100) %有点，存在轮廓
        h(j) = rp - lp;
    end
end
hmax = max(h);%有效高度
w_h = wmax/hmax;%宽高比
Lmax = 0;Lmin = 100;%字符轮廓线最大和最小值位置
Rmax = 0;Rmin = 100;
for i = 1:m
    if(Lp(i) ~= 0)
        if(Lp(i) > Lmax)
            Lmax = Lp(i);
        end
        if(Lp(i) < Lmin)
            Lmin = Lp(i);
        end
    end
    if(Rp(i) ~= 0)
        if(Rp(i) > Rmax)
            Rmax = Rp(i);
        end
        if(Rp(i) < Rmin)
            Rmin = Rp(i);
        end
    end
end
Ldif = [];
Rdif = [];
for i = 1:49
    if(Lp(i) ~= 0) && (Lp(i+1) ~= 0)
        Ldif = [Ldif, Lp(i+1)-Lp(i)];
    end
    if(Rp(i) ~= 0) && (Rp(i+1) ~= 0)
        Rdif = [Rdif, Rp(i+1)-Rp(i)];
    end
end
Lpeak_plus = max(Ldif);%左正峰
Rpeak_plus = max(Rdif);%右正峰
Lpeak_minus = min(Ldif);%左负峰
Rpeak_minus = min(Rdif);%右负峰
Lpeak = abs(Lpeak_plus) + abs(Lpeak_minus);%左峰
Rpeak = abs(Rpeak_plus) + abs(Rpeak_minus);%右峰
contour = [wmax, ratio, hmax, w_h, Lmax, Lmin, Rmax, Rmin, Lpeak_plus,
Rpeak_plus, Lpeak_minus, Rpeak_minus, Lpeak, Rpeak];
```

```
feature = [feature, contour];
```

3.4 特征提取总结

综上所述我选取了共 102 维的特征向量。在特征向量的独立性，有效性，鲁棒性以及抗干扰性上都比传统手写字符识别中使用的 13 点特征提取法有了很大的提高。而高纬度特征向量带来的问题就是需要更多的训练样本来进行训练，在本项目中，我随机挑选了多位拥有不同手写习惯的同学的 2000 个手写对号和错号样本作为训练集，200 个样本作为测试集。测试结果与 13 点特征提取法相比在识别率上有了很大的提高与后文中提到的训练好的 SVM 分类器结合可以达到 100%的识别率，这点将在第四章结果分析中做具体介绍。

第4章 支持向量机与序列最小优化算法

4.1 支持向量机 SVM

支持向量机(Support Vector Machine, SVM)是属于机器学习中监督学习的一种通用学习方法，在处理非线性及高维度模式识别中展现出许多独有的优势。SVM 的理论体系所覆盖的范围非常广，包括对偶表示，特征空间，学习理论，优化理论和学习算法等。这几年来，关于 SVM 的理论和算法都获得了里程碑性的突破，逐渐变为克服“维数灾难”和“过学习”等机器学习中传统难题的有力解决手段。SVM 的应用也延伸到生活中的各个领域，如: 用于文本和超文本的分类，在归纳和直推方法中都可以显著减少所需要的有类标的样本数; 用于图像分类。实验结果显示：在经过三到四轮相关反馈之后，比起传统的查询优化方案，支持向量机能够取得明显更高的搜索准确度; 在医学中用于分类蛋白质种类，已有超过 90%的蛋白质能够被正确识别和分类; 用于手写字体识别，与神经网络算法相比，使用 SVM 训练的手写字体识别往往有更好的学习效率和识别率。

4.1.1 线性可分割问题

支持向量机解决的最基本的问题就是解决线性可分割情况下的最优分类面问题。考虑 n 维空间上的分类问题：给定训练集$T = \{(x_1, y_1), (x_2, y_2), \ldots., (x_n, y_n)\}$。其中$x \in X = R^n$是 n 维空间上的 n 维向量，$y \in Y = \{-1,1\}$，y 的取值为-1 或 1，我们这里只考虑简单的二分类问题，-1 和 1 表示的类别的标签 label 用来表示不同的分类，根据后面的函数间隔的定义我们这里使用-1 或 1。i=1,2,3…n。(x_i, y_i)的二元对表示的是在训练集中的一个样本x_i所对应的类别y_i。

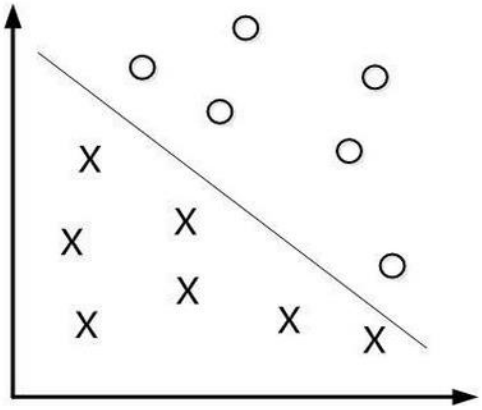


图 4-1 线性可分割问题

如上图所示在我们的训练集中 x 表示的问题的一类，〇表示的是问题的另一类。线性可分割指的是在我们的 n 维平面中，找到一条如图上的直线，这条直线就是一个超平面，超平面的一边所对应的数据点都是 1，另一边都是-1。不存在误分类的情况，这种问题就叫线性可分割问题。

进一步这个超平面可以用函数$f(x) = w^T x + b$表示，当$f(x)$=0 的时候，x则是位于超平面上的点，$f(x) > 0$的时候，x就是属于 y=1 的点；$f(x) < 0$的时候，x就是属于 y=-1 的点。如下图所示：

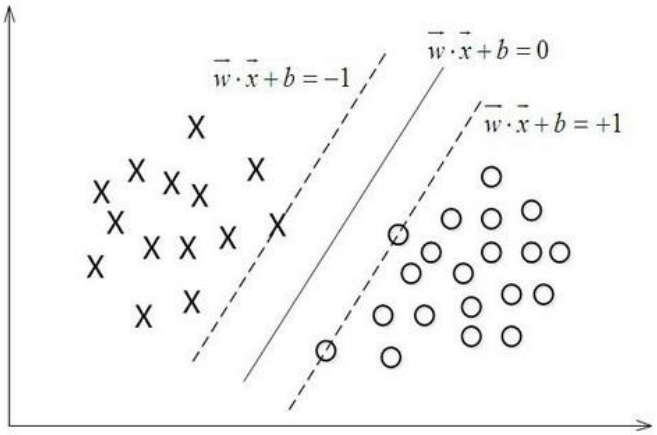


图 4-2 超平面与对应类别

综上所述，我们在实际分类的时候，从集合中选取一个向量 x，将其带入 f(x)，如果值小于 0 则属于标签为-1 的类，相反如果值大于 0 则属于标签为 1 的类。

所以，接下来我们要解决的问题就是，如何确定这个超平面？从宏观上讲，这个超平面需要最合适的分开两类样本的一条“直线”。具体来讲就是这条“直线”距离两类样本的间隔最大。所以，得寻找有着最大间隔的超平面。

4.1.2 函数间隔 Functional margin 与几何间隔 Geometrical margin

在我们确定了超平面$w^T x + b = 0$ 以后，我们可以用$|w^T x + b|$来表示点 x 到超平面的距离。如果$w^T x + b > 0$则该点 x 属于 y=1 标签一类，$w^T x + b < 0$ 则该点 x 属于 $y = -1$标签一类。所以我们可以用$y \times (w^T x + b)$的正负性来判定或表示分类的正确性。如果我们将该点 x 正确分类的话，则有 $y \times (w^T x + b) > 0$，否则 $y \times (w^T x + b) < 0$。于此，我们便引出了函数间隔（functional margin）的概念。

我们定义函数间隔$\hat{\gamma}$为：

$$\hat{\gamma} = y(w^T x + b)$$

而超平面$w^T x + b = 0$关于 T 中所有样本点(x_i, y_i)的函数间隔最小值，便为超平面$w^T x + b = 0$关系训练数据集 T 的函数间隔：

$$\hat{\gamma} = min(\widehat{\gamma_i})$$ i=1,2,3…n

但是如果我们仅仅定义函数间隔会遇到问题，即如果改变超平面参数比例w, b (如分别将它们扩大两倍到$2w, 2b$)，函数间隔变为原来的 2 倍，但是超平面的位置并没有改变。所以仅有函数间隔还远远不够。

因此我们需要对超平面的法向量 w 加以限制，从而我们得出了真正判定分类器间隔的几何间隔(geometrical margin)概念。

假定对某点 x ，其在直线上的投影为X_0，w 为垂直于超平面的向量，则点 x 到超平面的距离，如下图所示：

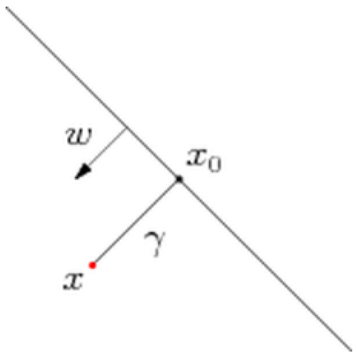


图 4-3 法向量与几何间隔

我们可以根据解析几何中的点到直线距离公式，或向量求解等方法(在这里不加赘述)求出给定一点 x 到超平面$w^T x + b = 0$ 的表示为：

$$x = x_0 + \gamma \frac{w}{\|w\|}$$

其中 x0 为该点 x 到超平面上的投影点，γ为 x 到超平面的距离。$\|w\|$为w 的二阶范数(相当于 w 的模长)，$\frac{w}{\|w\|}$则是单位法向量（一个向量除以它的模谓之单位向量）。

因为 x0 是超平面上的点，所以我们将 x0 带入 f(x)=0 中有$w^T x_0 = -b$

让$x = x_0 + \gamma \frac{w}{\|w\|}$两边同时乘以$w^T$,再根据$w^T x_0 = -b$和$w^T w = \|w\|^2$，则有：

$$\gamma = \frac{w^T x + b}{\|w\|} = \frac{f(x)}{\|w\|}$$

所以我们可以用函数间隔来表示几何间隔：

$$\gamma = \frac{\hat{\gamma}}{\|w\|}$$

综上所述，我们可以分析出：几何间隔就是函数间隔除以$\|w\|$，而且函数间隔$y(w^T x + b) = yf(x)$实际上就是$|f(x)|$，只是人为定义的一个间隔度量，而几何间隔$\frac{|f(x)|}{\|w\|}$才是直观上的点到超平面的距离。

4.1.3 最大间隔分类器 Maximum Margin Classifier

我们对一个数据集进行分类，如何保证我们的分类器是最优的呢？我们说，如果超平面距离两类样本的距离越大，则分类的可信度也越大。所以为了使分类的可信度提高，需要最大化超平面的几何间隔。这个间隔就是下图中的 Gap 的一半：

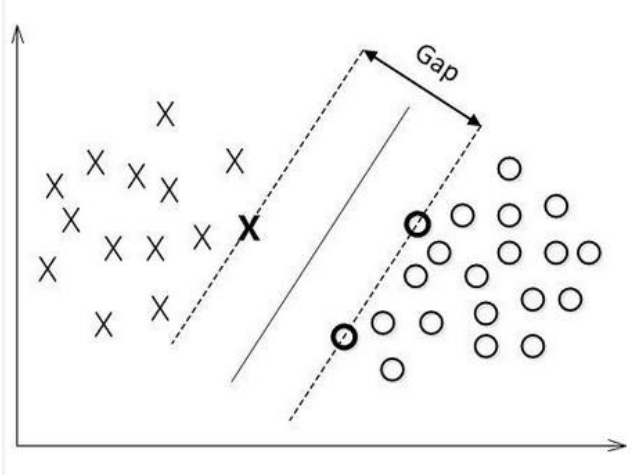


图 4-4 最大间隔分类器

由上图可以看到我们要选取的超平面位置从直观上要处于两类数据点的中间，也就是到两边的间隔相同。通过由前面的分析可知：使用函数间隔来固定超平面是不可行的，因为可以等比例改变w, b使得函数值发送改变。所以我们应该用几何间隔来表示上述提到的超平面距离数据点的间隔。

所以最大间隔分类器的目标方程为：

$$\tilde{\gamma} = \max(\gamma)$$

$$subject\ to.$$

$$y_i(w^T x_i + b) = \hat{\gamma}_i \geq \hat{\gamma}, i = 1,2,3, \dots, n$$

$$\gamma = \frac{\hat{\gamma}}{\|w\|}$$

我们根据函数间隔$\hat{\gamma}$的定义可知，我们可以随意放大缩小函数间隔的比例，我们不妨设$\hat{\gamma} = 1$(这是为了方便我们后文的推导和优化)，于是我们有：

$$\tilde{\gamma} = \max(\frac{1}{\|w\|})$$

$subject\ to.$

$$y_i(w^T x_i + b) \gg 1, i = 1,2,3,\dots,n$$

$$\gamma = \frac{1}{\|w\|}$$

这个目标函数就是在相应的约束条件下最大化这个$\frac{1}{\|w\|}$的值，而$\frac{1}{\|w\|}$的值就是我们的几何间隔。

举个例子，如下图所示：

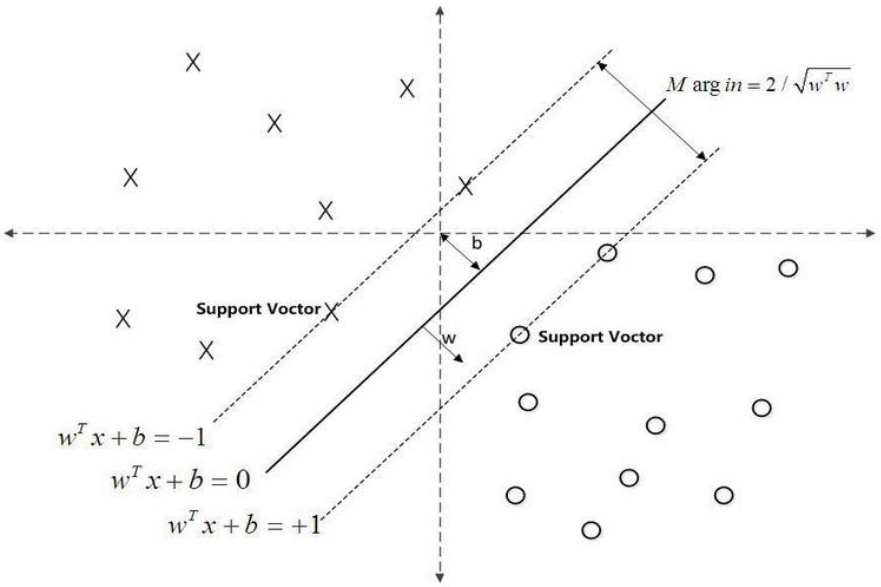


图 4-5 最大间隔超平面选取及支持向量

从图中我们可以看到，中间的实线就是我们要找的最优平面，它到两边不同类别数据集的距离相等(图中虚线位置)，距离分别为$\tilde{\gamma} = \frac{1}{\|w\|}$。正好位于虚线上的点我们叫做支持向量，这些向量$x_i$满足$y_i(w^T x_i + b) = 1$。(实际上也就是这些向量决定了我们超平面的位置，进而也就决定了该分类器的最终形态。所以这就是支持向量机这个名字的由来。)显然我们的点可以被分为支持向量的点和非支持向量的点，对于非支持向量，显然有$y_i(w^T x_i + b) > 1$。

4.1.4 求解 SVM 的一种方法——对偶方法

回忆我们需要求解的式子：

$$\tilde{\gamma} = \max(\frac{1}{\|w\|})$$

$subject\ to.$

$$y_i(w^T x_i + b) \gg 1, i = 1,2,3,\dots,n$$

我们需要找到满足条件的最大值。从另一个角度看，我们也可以把问题转化

为求$\|w\|$的最小值。为了方便后文的计算我们将原问题化为求：

$$\min(\frac{1}{2}\|w\|^2)$$

$subject\ to.$

$$y_i(w^T x_i + b) \geq 1, i = 1,2,3,\dots,n$$

二次的目标方程加上线性的约束条件，所以就变成了一个凸二次规划问题。这个问题可以用现成的 QP (Quadratic Programming) 优化包进行求解。简单来说就是在一定的约束条件下，找到目标的最优解。

但是上式的二次规划较为复杂，我们可由拉格朗日对偶性（Lagrange Duality）将原始问题转换成等价的对偶问题，从而通过解决对偶问题来得到线性可分条件下支持向量机的最优解。这样做的优点有两个：对偶问题往往更易于求解；通过内积的形式引入核函数的概念，进而可以推广到非线性分类问题。

拉格朗日对偶问题，简单来说就是给每个约束条件加上一个拉格朗日乘子（Lagrange multiplier），来定义拉格朗日函数。我们看下面的问题：

$$min_w \ \ f(w)$$

$subject\ to$:

$$h_i(w) = 0 \ \ i = 1,2,3 \dots n$$

对于上述的目标问题及其约束条件，我们可以构造拉格朗日公式：

$$L(w,\beta) = f(w) + \sum_{i=1}^{n} \beta_i h_i(w)$$

在这里β_i被称为拉格朗日算子，然后我们分别对 w 和β求偏导，使偏导数分别等于 0，然后求解出 w 和β；

$$\frac{\partial L}{\partial w} = \vec{0}, \frac{\partial L}{\partial \beta_i} = 0$$

因为在实际运算中 w 往往是 n 维向量(和训练集维度一样)，所以对向量求偏导，结果也应该是向量。

接着我们再引入线性不等式约束条件,从而完整我们的原始优化问题：

$$min_w \ \ f(w)$$

$subject\ to$:

$$g_i(w) \leq 0$$

$$h_i(w) = 0 \ \ i = 1,2,3 \dots n$$

转化为一般化拉格朗日公式：

$$L(w,\alpha,\beta) = f(w) + \sum_{i=1}^{n} \alpha_i g_i(w) + \sum_{i=1}^{n} \beta_i h_i(w)$$

在这里α_i和β_i都被称为拉格朗日算子。 因为不等式约束条件是小于等于 0

的，我们要转化

$$y_i(w^T x_i + b) \geq 1, i = 1,2,3, \dots, n$$

为：

$$1 - y_i(w^T x_i + b) \leq 0, i = 1,2,3, \dots, n$$

所以针对我们在求解 SVM 的最优化问题中，拉格朗日公式为：

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i (y_i(w^T x_i + b) - 1)$$

然后我们令：

$$\theta(\mathrm{w}) = \max_{\alpha_i \geq 0} L(w, b, \alpha)$$

容易验证，当某个约束条件不满足时，例如$y_i(w^T x_i + b) < 1$，$\theta(\mathrm{w}) = \infty$(因为$\alpha_i > 0$,令$\alpha_i = \infty$即可)，而当所有条件都满足时，则$\theta(\mathrm{w}) = \frac{1}{2}\|w\|^2$，也就是最初要最小化的量。

因此我们将原始的最小化$\frac{1}{2}\|w\|^2$的问题转化，等价地转化为最小化 $\theta(\mathrm{w})$，$subject\ to\ \alpha_i \geq 0, i = 1,2,3 \dots n$。于是目标函数变为：

$$\min_{w,b} \theta(\mathrm{w}) = \min_{w,b} \max_{\alpha_i \geq 0} L(w, b, \alpha) = p^*$$

这里用p^*表示这个问题的最优解。如果我们要直接求解上式，则需要对 w, b 两个参数，而α_i又同时约束着不等式，所以求解显得没有头绪。我们不妨把求解的顺序交换一下，变为先求最大值再求最小值。

$$\max_{\alpha_i \geq 0} \min_{w,b} L(w, b, \alpha) = d^*$$

交换以后的新问题是原始问题的对偶问题。这个新问题用d^*来表示。一般地$d^* \leq p^*$

在满足某些条件下(KKT 条件在后文会有介绍) d^*和p^*两者相等，这时就可以通过求解对偶问题来求解原始问题。

我们先跳过这个 KKT 条件直接介绍对偶问题的求解过程，就是先对

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i (y_i(w^T x_i + b) - 1)$$

求关于 w, b 的极小，然后求关于α_i的极大。

我们首先固定α_i(将α_i视为常量)，对于 w, b 分别求偏导数：

$$\frac{\partial L}{\partial w} = \vec{0}, \frac{\partial L}{\partial b} = 0$$

可以推导出：

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

对函数的推导过程进行简单解释，$\frac{1}{2}\|w\|^2$对 w 求偏导为w而不是$\|w\|$，而后一项可以忽略与 w 无关的项，对于 w 求偏导为$-\sum_{i=1}^{n} \alpha_i y_i x_i$。关于$b$的求偏导则可以忽略所有关于 w 的项即可简单求出结果。

我们将上述两个结果带入$L(w,b,\alpha)$有：

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i (y_i(w^T x_i + b) - 1)$$

$$L(w,b,\alpha) = \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - b\sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i$$

$$L(w,b,\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

4.1.4.1 KKT 约束条件

我们回到之前的拉格朗日对偶问题的成立条件中，当满足 KKT 条件时d^*和p^*两者相等成立。对于一般的二次规划问题：

$$min_x \ \ f(x)$$

$subject\ to$:

$$g_j(x) \le 0 \ j = 1,2,3 \dots m$$
$$h_i(x) = 0 \ \ i = 1,2,3 \dots n$$

我们假设 f 是凸函数，在 f 上的凸优化为：找出满足的$x^*, x^* \in R^n$,并且对R^n中的每一个x有：$f(x^*) \le f(x)$。

在 Karush-Kuhn-Tucker(KKT)条件中，上面提到的x^*必须具有以下条件：

$$h_i(x^*) = 0 \ \ i = 1,2,3 \dots n$$
$$g_j(x^*) \le 0 \ j = 1,2,3 \dots m$$
$$\nabla f(x^*) + \sum_{i=1}^{n} \beta_i \nabla h_i(x^*) + \sum_{j=1}^{m} \alpha_j \nabla g_j(x^*) = 0$$

其中：

$$\beta_i \neq 0，\ \alpha_j \ge 0，\ \alpha_j g_j(x^*) = 0$$

这也是为什么我们在解决$\min_{w,b} \theta(w) = \min_{w,b} \max_{\alpha_i \ge 0} L(w,b,\alpha) = p^*$时，需要假设$\alpha_i$大于等于 0 的原因。一般我们在求解 SVM 最大间隔分类平面的原始问题都满足 KKT 分类条件。

4.1.4.2 最终分类平面的表示

通过之前的讨论，我们已经要把原始问题转化为如下问题：

$$\max(\sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j)$$

$subject\ to$

$$\alpha_i \ge 0$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \ \ i = 1,2,3 \ldots n$$

在我们求出最优的α_i之后，根据$w = \sum_{i=1}^{n} \alpha_i y_i x_i$，我们就可以求出 w 的最优值$w^*$，然后通过下式:$b = \frac{\max_{i,y_i=-1} w^{*T} x_i + \min_{i,y_i=1} w^{*T} x_i}{2}$。从直观上讲，因为我们已经确定了$w^*$所以我们的超平面法向量也就确定，也就确定了方向，接下来我们只需要将超平面在空间内平移，也就是修改 b 的值，让他到两边支持向量的距离相等，也就得到 b 的最优值。

不过虽然我们经过了一系列的优化，并且引入对偶问题，最后将原始问题化简到了一个较为明朗的形式，但是我们还是需要使用 QP 来求解问题。除了需要使用额外的求解二次规划问题的工具外(往往需要额外的开销)，而且会大大增加时间复杂度。尤其是当特征向量的维数增多之后，如上文中提到的 102 维特征向量，用二次规划工具求解的时间往往令人难以接受。下面我将介绍一种针对以上问题的解决方法，SMO 算法，从根本上解决了使用二次规划工具求解 SVM 的问题，并且成为了一种较快的 SVM 训练算法。

4.2 SMO 序列最小优化算法

4.2.1 初步了解 SMO 思想

序列最小优化算法(Sequential minimal optimization, SMO)，是由微软研究院的 Platt 发明于 1998 年的一种解决 SVM 优化问题的算法[9][10]。Platt 是基于分解算法上提出的这个算法，由于其在多方面的优点，如：可以避免传统 SVM 复杂的训练方法和使用昂贵的 QP(二次规划)工具等，而被广泛应用于 SVM 的训练之中。

我们将目光转回上一章最后我们的结论目标：

$$\max(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T \ x_j)$$

$subject\ to$

$$0 \le \alpha_i \le C$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \ \ i = 1,2,3 \ldots n$$

这里的 C 叫做惩罚项，是为了解决在训练样本中有个别坏点而引入的松弛参数，通过设置这个参数，可以减小所谓坏点对于训练的影响。

我们需要求解关于α_i，$i = 1,2,3,\ldots,n$关于 i 个变量二次线性规划问题，这个问题的数量是关于我们训练集的样本数目线性增加的。如果对于简单的训练样

本还是可以考虑使用二次规划工具来求解，但是当样本数量增加了以后二次规划算法就带来了巨大的代价。为了解决这个问题，SMO 算法的思路是在每次迭代的过程中，只选取训练集的两个样本点，对于其他的样本点将其视为常量数据。这样就将过多的样本集转化为简单的求解二元最优化问题。

不失一般性，假设我们选取的样本点是$(x_1, y_1), (x_2, y_2)$，这样一来我们需要选取一对(α_1, α_2)在固定$(\alpha_3, \alpha_4, \dots, \alpha_n)$的情况下。这样我们的优化目标，我们以 W 来表示就变成了关于(α_1, α_2)的函数，并且(α_1, α_2)满足：

$$0 \le \alpha_1, \alpha_2 \le C$$

$$\alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{n} \alpha_i y_i$$

为了方便起见，我们将等式右侧的值记为ξ。

$$\alpha_1 y_1 + \alpha_2 y_2 = \xi$$

y_1, y_2都属于{-1,1}集合中所以有

$$\alpha_1 + \alpha_2 = \xi$$
$$or \quad \alpha_1 - \alpha_2 = \xi$$

我们以后者为例，$y_1 = 1, y_2 = -1$,上式可以表示为坐标系中的一条直线，其斜率为 1。如下图：

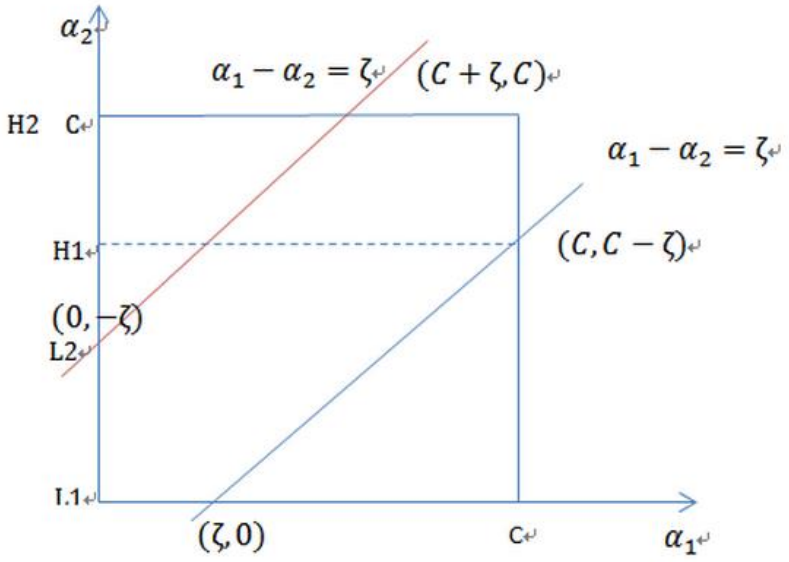


图 4-6 横坐标为α_1, 纵坐标为α_2

可以清楚地看到，α_1和α_2满足的约束条件可以在二维平面上表示。即α_1和α_2要在矩形区域内，也要在直线上。所以我们有：

$$L = \max(0, \alpha_2 - \alpha_1), H = \min(C, C + \alpha_2 - \alpha_1)$$

同样地我们有在y_1, y_2同号时，

$$L = \max(0, \alpha_2 + \alpha_1 - \mathrm{C}), H = \min(C, \alpha_2 + \alpha_1)$$

再根据约束条件 2：

$$\alpha_1 = (\xi - \alpha_2 y_2)\, y_1$$

再带入我们的目标函数 W 中:

$$W(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \dots, \alpha_n) = W((\xi - \alpha_2 y_2) y_1, \alpha_2, \alpha_3, \alpha_4, \dots, \alpha_n)$$

将其展开我们可以表示成二次函数的形式:$a{\alpha_2}^2 + b\alpha_2 + c$,其中 a,b,c 都是固定值，我们在$L \le \alpha_2 \le H$的域内求解二次函数的最值，用$\alpha_2$'来表示。我们可以根据$\alpha_2$'的情况，来推出最终$\alpha_2$的最优值$\alpha_{2_new}$，情况如下：

$$\alpha_{2\_new} = \begin{cases} H & if\, \alpha_2' > H \\ \alpha_2' & if\, L \le \alpha_2' \le H \\ L & if\, \alpha_2' < L \end{cases}$$

接着我们有$\alpha_{1_new} = (\xi - \alpha_{2_new} y_2)\, y_1$。

上述就是 SMO 算法迭代优化的大体思路。在下一小节中，我们将介绍 Platt 在 1998 年《Sequential Minimal Optimization：A Fast Algorithm for Training Support Vector Machines》一文中的 SMO 算法推导。

4.2.2 SMO 的推导

我们之前的最终结果是$f(x) = w^T x + b$，在这里我们将它改为：$u = \vec{w} \cdot \vec{x} - b$。实际上和我们之前用的公式$f(x)$所表达的内容是一致的，只不过换了一种表达方式。我们将原来的目标函数

$$\max(\textstyle\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T\, x_j)$$

转化为：

$$\min_{\alpha} \Psi(\vec{\alpha}) = \min_{\alpha} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i\, y_j \left(\overrightarrow{x_i} \cdot \overrightarrow{x_j}\right) \alpha_i \alpha_j - \sum_{i=1}^{n} \alpha_i$$

Subject to:

$$\alpha_i \ge 0$$

$$\sum_{i=1}^{n} \alpha_i\, y_i = 0$$

我们引入松弛变量 C 后，最终我们的目标问题变为：

$$\min_{\alpha} \Psi(\vec{\alpha}) = \min_{\alpha} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i\, y_j K\left(\overrightarrow{x_i}, \overrightarrow{x_j}\right) \alpha_i \alpha_j - \sum_{i=1}^{n} \alpha_i$$

Subject to:

$$0 \le \alpha_i \le C$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

其中我们用$K(\overrightarrow{x_i}, \overrightarrow{x_j})$来表示$(\overrightarrow{x_i} \cdot \overrightarrow{x_j})$的内积，也就是提前引入下一章将要介绍的核函数的概念。

与之前相同地我们选取一对(α_1, α_2)，然后固定$(\alpha_3, \alpha_4, \dots, \alpha_n)$。不断地从拉格朗日乘子中选取一对乘子，然后固定其余乘子，来迭代求解。

回忆上一节的 KKT 条件，我们可以得出取不同值，α的不同意义：

$$\alpha_i = 0 \Leftrightarrow y_i u_i \geq 1$$
$$0 < \alpha_i < C \Leftrightarrow y_i u_i = 1$$
$$\alpha_i = C \Leftrightarrow y_i u_i \leq 1$$

a) 说明α_i是普通的分类，对应的样本是正常的分类，在边界的内部。

b) 说明α_i是在边界上，对应的样本是支持向量，在分类的边界上。

c) 说明α_i在间隔边界之间，对应的样本属于坏点。

除此之外的情况是属于不满足 KKT 条件的拉格朗日乘子。这些拉格朗日乘子就是我们需要更新的α_i，同时也还要满足$\sum_{i=1}^{n} \alpha_i y_i = 0$的约束条件。

考虑我们之前推出的α_i的边界条件：

在y_1, y_2异号时，

$$L = \max(0, \alpha_2 - \alpha_1), H = \min(C, C + \alpha_2 - \alpha_1)$$

在y_1, y_2同号时，

$$L = \max(0, \alpha_2 + \alpha_1 - \mathrm{C}), H = \min(C, \alpha_2 + \alpha_1)$$

我们将α_2和α_1带入 Ψ 中，有：

$$\Psi = \frac{1}{2} K_{11} {\alpha_1}^2 + \frac{1}{2} K_{22} {\alpha_2}^2 + s K_{12} \alpha_1 \alpha_2 + y_1 \alpha_1 v_1 + y_2 \alpha_2 v_2 - \alpha_1 - \alpha_2 + \Psi_C$$

解释一下上式，$K_{ij} = K(\overrightarrow{x_i}, \overrightarrow{x_j})$是特征向量的内积，

$$v_i = u_i + b^* - y_1 {\alpha_1}^* K_{1i} - y_2 {\alpha_2}^* K_{2i}$$

${\alpha_1}^*, {\alpha_2}^*$是迭代前的数值，是固定的值。$\Psi_C$是常数。由第二个约束条件我们有：

$$y_1 {\alpha_1}^* + y_2 {\alpha_2}^* = -\sum_{i=3}^{n} \alpha_i y_i = \alpha_1 y_1 + \alpha_2 y_2$$

上式左右两边同时乘以y_1,令$s = y_1 y_2$：

$${\alpha_1}^* + s{\alpha_2}^* = \alpha_1 + s\alpha_2 = w$$

其中$w = -y_1 \sum_{i=3}^{n} {\alpha_1}^* y_i$,将上式带入

$$\Psi = \frac{1}{2} K_{11} {\alpha_1}^2 + \frac{1}{2} K_{22} {\alpha_2}^2 + s K_{12} \alpha_1 \alpha_2 + y_1 \alpha_1 v_1 + y_2 \alpha_2 v_2 - \alpha_1 - \alpha_2 + \Psi_C$$

我们有：

$$\Psi = \frac{1}{2}K_{11}(w - s\alpha_2)^2 + \frac{1}{2}K_{22}{\alpha_2}^2 + sK_{12}(w - s\alpha_2)\alpha_2 + y_1(w - s\alpha_2)v_1 + y_2\alpha_2 v_2 - (w - s\alpha_2) - \alpha_2 + \Psi_C$$

于是将Ψ 转化为了关于α_2的二元一次函数。关于Ψ 对α_2求导数，当导数值为 0 的时候， α_2的值使上式取得最小值。

于是推导出了α_2的更新公式：

$$\alpha_2{}' = \alpha_2 + \frac{y_2(E_1 - E_2)}{K_{11} + K_{22} - 2K_{12}}$$

公式左侧的为更新后的值，右侧的α_2是原值，$E_i = u_i - y_i$

同样地我们还需要对α_2'进行约束：

$$\alpha_{2\_new} = \begin{cases} H & if\alpha_2{}' > H \\ \alpha_2{}' & ifL \le \alpha_2{}' \le H \\ L & if\alpha_2{}' < L \end{cases}$$

所以α_1的更新公式为：$\alpha_{1\_new} = \alpha_1 + s(\alpha_2 - \alpha_{2\_new})$。

我们接下来更新 b 的值，b 的推导公式如下：

$$b = \begin{cases} b1\ if\ 0 < \alpha_{1\_new} < C \\ b2\ if\ 0 < \alpha_{2\_new} < C \\ \dfrac{b1 + b2}{2} & other \end{cases}$$

其中，$b1, b2$ 的值可以表示为：

$$b1 = E_1 - y_1(\alpha_1 - \alpha_1{}')K_{11} + y_2\left(\alpha_{2_{new}} - \alpha_2\right)K_{12} + b$$
$$b2 = E_2 - y_1(\alpha_1 - \alpha_1{}')K_{12} + y_2\left(\alpha_{2_{new}} - \alpha_2\right)K_{22} + b$$

这样我们所有需要的值的更新公式就推导出来了。需要注意的是每次更新完一对拉格朗日乘子后我们都需要重新计算一遍 b 的值，以及对应E_i的值。

综上所述最后的$f(x) = w^T x + b$为：$f(x) = \sum_{i=1}^{n} \alpha_i y_i < x_i, x > + b$。

4.2.3 SMO 算法步骤

伪代码：

```
While(不收敛) do
    选择一组(α_i, α_j),并更新相应的;
    在固定其余α的情况下，更新W(α)的值
end
```

其中关于(α_i, α_j)的选择，我们需要根据上一章提到的首先选择一个不满足 KKT 条件的α_i，然后第二个拉格朗日乘子我们可以选择:$\max_{\alpha_j} |E_i - E_j|$。这个方法通常被称为启发式寻找方法。

4.2.4 SMO 算法小结

上述就是 SMO 算法的介绍，推导和算法描述。在现在基于 SMO 算法的 SVM 已经广泛地应用到了各个方面。比如文本处理，图像分类，和在本文中涉及到的版面分析和手写字符识别。

实际上在上述的 SMO 算法的推导中，虽然其中可能需要较多的迭代次数，但是每次的迭代量计算比较小，而且在实际操作过程中，我们可以发现真正的支持向量的数量是比较少的(即使样本数量很大)。这就使得最后大部分的α_i的值都为 0，只有少数支持向量对应的α_i的值是非 0 的，这就极大地减少了计算的复杂度。而且其中有一项$K_{ij} = K(\overrightarrow{x_i}, \overrightarrow{x_j}) =< x_i, x_j >$,这种处理方式引入了内积的概念，不仅避免了存储大量样本特征向量矩阵的存储，也没有矩阵的运输，而且引入了核函数的概念，这些我们在下一章节会介绍。

4.3 核函数的选择与最优参数选取

4.3.1 核函数

在上一章里我们介绍的是 SVM 的介绍和 SMO 算法的推导，所有这些问题都是建立在这样一个假设下的，那就是线性可分，简单点来说就是在 N 维空间中存在一个超平面，可以把我们需要分类的数据完美的分隔开来。虽然我们也引入了惩罚参数 C，针对在训练集中影响分类的坏点，但是这只是个别的情况。那么当满足上述条件的超平面不存在，也就是线性不可分时，我们应该如何设计我们的分类器呢？(实际上，在 SVM 分类的应用中，大部分处理的情况都是属于线性不可分的)。

我们的解决办法是，通过一个核函数 k，来把我们的 N 维的数据映射到更高维度的空间中去。

核函数的形式是$K(x, z) =< \varphi(x), \varphi(z) >$。$\varphi$是特征向量 x 到高维空间的一个映射，当然也可以是线性的。在上一节中，我们最后得到的分类器表达公式是：$f(x) = \sum_{i=1}^{n} \alpha_i y_i < x_i, x > +b$。在拉格朗日对偶问题的计算中，我们需要计算$< x_i, x_j >$也就是内积。其实这里的$< x_i, x_j >$也就是线性核函数的一个特例：$\varphi(x) = x$。

举个例子,如下图所示的数据：

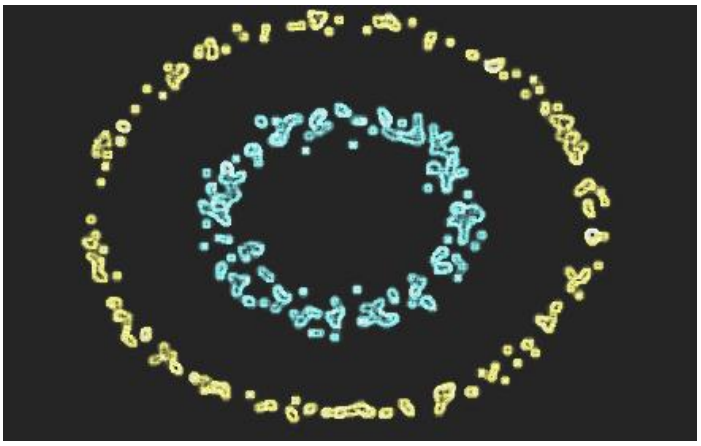


图 4-7 线性不可分例子

在此图中，特征向量维二维，显然在二维空间中不能找到一条线来分隔两类数据。但是一个椭圆环就可以完美地分隔开两类数据。

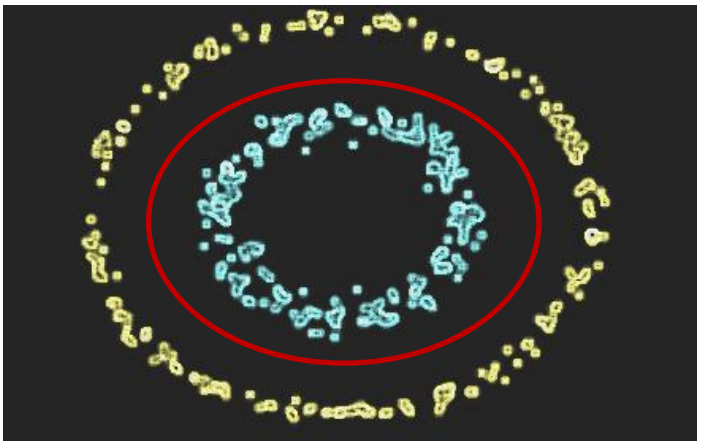


图 4-8 一种分隔情况

我们可以设在二维平面内这个椭圆曲线为:

$$ax_1 + bx_1{}^2 + cx_2 + dx_2{}^2 + ex_1x_2 + f = 0$$

所以我们将原来的二维特征向量(x_1, x_2)映射到五维空间中$(x_1, x_1{}^2, x_2, x_2{}^2, x_1x_2)$即可找到一个超平面能使其变为线性可分割的情况。

4.3.2 常用核函数

核函数不仅可以将原始特征向量映射到高维空间内，来解决在二维空间中线性不可分的问题。更重要的是，在特征向量映射到高维空间之后，其计算复杂度仍然很低[11]。

下面是几个我们常用的核函数：

a) 线性核：$K(x_i, x_j) = x_i{}^T x_j$

b) 多项式核：$K(x_i, x_j) = (\gamma x_i{}^T x_j + \mathrm{r})^d, \gamma > 0$

c) 径向基函数神经（RBF）：$K(x_i, x_j) = \exp\left(-\gamma\left\|x_i - x_j\right\|^2\right), \gamma > 0$

d) Sigmoid 函数：$K(x_i, x_j) = \tanh\left(\gamma {x_i}^T x_j + r\right)$

线性核就是我们在普通线性可分割的情况下使用的内核，这种情况是不多见的。我们上述的那个例子就是一种多项式核，维度为 5，d=5。当然这些不是全部的内核，还有许多如高斯核等[12]，在这里就不多介绍了。

那么我们如何选取自己的核函数呢？在本项目中我使用部分样本，对比了上述几种核函数分类的准确率。最终选取了 RBF 内核。其实在一般情况下，RBF 内核也是我们的一个合理选择。除了可以将特征向量映射到高维空间外，RBF 内核与多项式内核相比，需要更少的参数，这就减少了核函数的复杂程度，而且当特征向量维度较高时，RBF 核具有较少的数值困难。因为 RBF 内核的 $0 < K_{i,j} \leq 1$，相反当维数很大的时候，多项式内核可能达到无穷$\gamma {x_i}^T x_j + \mathrm{r} > 1$，或者接近于 0，，$\gamma {x_i}^T x_j + \mathrm{r} < 1$。

4.3.3 使用交叉验证和网格搜索选取最优参数

在 RBF 内核$K(x_i, x_j) = \exp\left(-\gamma\left\|x_i - x_j\right\|^2\right), \gamma > 0$中有两个参数，第一个为 RBF 自身的参数$\gamma$，第二个参数是引入松弛变量后的惩罚项参数 C。我们现在的问题就是如何选取最合适的(γ, C)来使得分类器可以正确预测未知数据。

一个通常的策略就是将数据集分割成两部分，其中的一个被认为是未知的。从“未知”获得的预测精度得到更精确反映了分类独立数据集的性能。此过程的改进版被称为交叉验证。在一个 k-折叠(k-fold)交叉验证中，我们首先将训练集分成 k 个相同大小的子集。接着一个子集作为测试集用于测试用 k-1 个子集作为训练集的分类器。因此，训练集中的每个实例都被验证了一遍，所以交叉验证的准确性就是实例被正确分类的比例。

而往往网格搜索和交叉验证是配合使用的。在下面的介绍中，我们用g来代替参数γ。网格搜索的思路是选取多对(g, C)来进行测试，然后选取交叉验证最准确的一项。简单来说就是让(g, C)在一个范围内进行测试，如$(\mathrm{C} = 2^{-5}, 2^{-4}, 2^{-3}, \ldots, 2^{5}; \mathrm{g} = 2^{-5}, 2^{-4}, 2^{-3}, \ldots, 2^{5})$，一般我们选取二的指数为基准，初始的步长为 1 或$\frac{1}{2}$。接下来我们找到准确率最高的位置，如果同时存在多对(g, C)拥有相同的准确率，那么我们就选取最小的一对。因为我们的惩罚项参数一般不可以选取的太高，太高的惩罚项参数会引起过拟合问题。接下来我们把(g, C)的范围缩小，把步长的大小也缩小，让(g, C)在更小的范围内测试，然后生成一个更好的分类器。

如我在本次项目中初始的(g, C)我选为$(\mathrm{C} = 2^{-5}, 2^{-4.5}, 2^{-4}, \ldots, 2^{5}; \mathrm{g} =$

$2^{-5}, 2^{-4.5}, 2^{-4}, \ldots, 2^{5}$)，步长为 0.5。选取的交叉验证为 5-fold。在全范围内的宽松网格如下图：

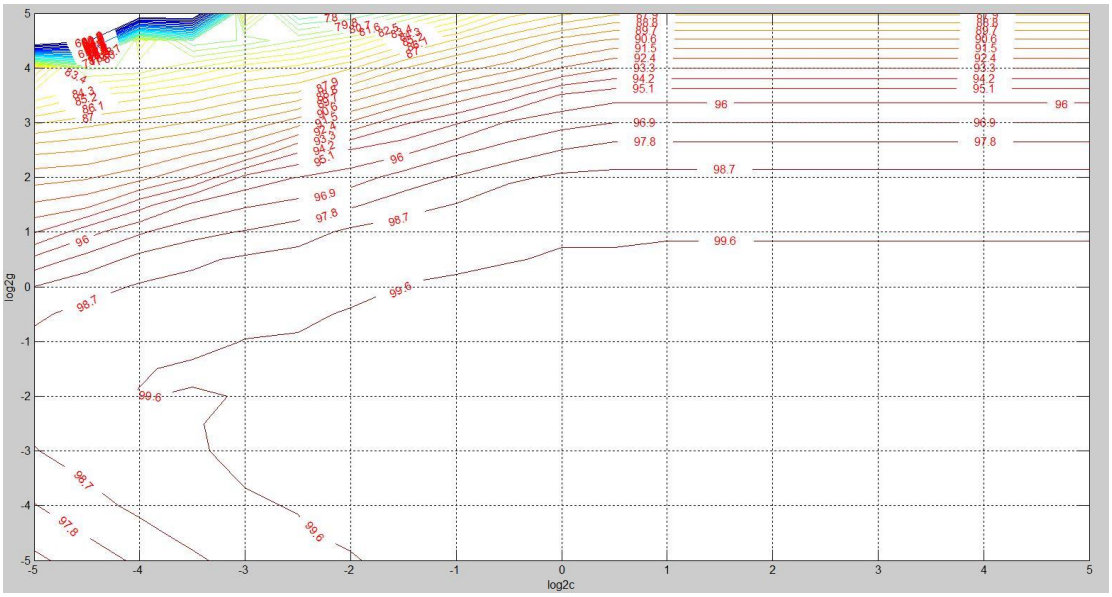


图 4-9 宽松网格在$(C = 2^{-5}, 2^{-4}, 2^{-3}, \ldots, 2^{5}; g = 2^{-5}, 2^{-4}, 2^{-3}, \ldots, 2^{5})$

此时的 bestC=$2^{-0.5}$，bestg=$2^{-1.5}$, 也就是在横坐标-0.5，纵坐标-1.5 附近。此时的准确率为：99.9455%。

此时我们设计一个更优化的网格在(-1.5,-0.5)附近，我们选取$(C = 2^{-1.5}, 2^{-1.25}, 2^{-1}, \ldots, 2^{0.5}; g = 2^{-2.5}, 2^{-2.25}, 2^{-2}, \ldots, 2^{-0.5})$的一个优化网格来继续选取。

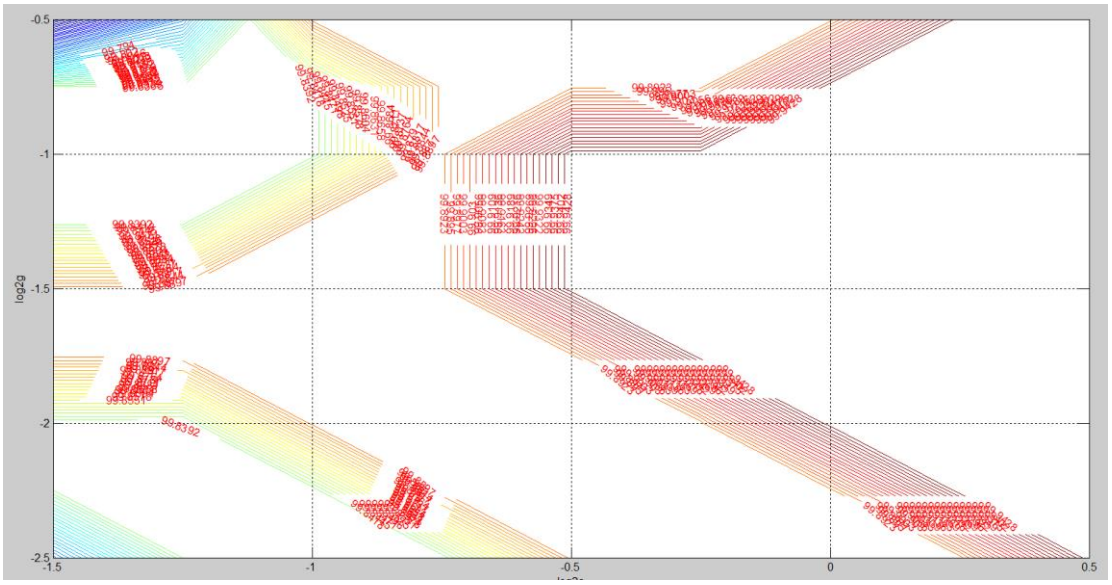


图 4-10 优化网格在$(C = 2^{-1.5}, 2^{-1.25}, 2^{-1}, \ldots, 2^{0.5}; g = 2^{-2.5}, 2^{-2.25}, 2^{-2}, \ldots, 2^{-0.5})$

我们可以看到在这个优化网格中，所有等高线所显示的百分比都相对较高，在一个更好的范围内。这时的准确率还为 99.9455%，这说明我们之前选取的(g, C)参数已经是一个较优的参数了。我们的训练样本为 1835 个，99.9455%说明有 1834 个点被正确地分类了，剩下的一个点应该是所谓的坏点，99.9455%的准确率就是一个可以接受的准确率。另外在优化网格中有多对(g, C)出现了同样的准确率，我们选取最小的一个即可。此时的 bestC=$2^{-0.5}$，bestg=$2^{-1.5}$。

```
function [bestpre,bestC,bestG] = SVMcg(train_label,train,minC,maxC,minG,maxG,v,Cstep,Gstep,step)
%使用网格搜索(Grid Search)和交叉验证(Cross Validation)
%搜索最优参数 c(惩罚项)和 g(rbf 核)
if nargin < 10
    step = 1.5;
end
if nargin < 8
    step = 1.5;
    Cstep = 1;
    Gstep = 1;
end
if nargin < 7
    step = 1.5;
    v = 3;
    Cstep = 1;
    Gstep = 1;
end
if nargin < 6
    step = 1.5;
    v = 3;
    Cstep = 1;
    Gstep = 1;
    maxG = 5;
end
if nargin < 5
    step = 1.5;
    v = 3;
    Cstep = 1;
    Gstep = 1;
    maxG = 5;
    minG = -5;
end
if nargin < 4
    step = 1.5;
    v = 3;
```

```
    Cstep = 1;
    Gstep = 1;
    maxG = 5;
    minG = -5;
    maxC = 5;
end
if nargin < 3
    step = 1.5;
    v = 3;
    Cstep = 1;
    Gstep = 1;
    maxG = 5;
    minG = -5;
    maxC = 5;
    minC = -5;
end
%% X:c 的参数范围 Y:的参数范围 cg:准确率
[X,Y] = meshgrid(minC:Cstep:maxC,minG:Gstep:maxG);
[m,n] = size(X);
cg = zeros(m,n);
%% 根据准确率 cg 选取最优参数，如果同样的准确率选取最小参数
bestC = 0;
bestG = 0;
bestpre = 0;
basenum = 2;
for i = 1:m
    for j = 1:n
        cmd = ['-v ',num2str(v),' -c ',num2str( basenum^X(i,j) ),' -g
',num2str( basenum^Y(i,j) )];
        cg(i,j) = svmtrain(train_label, train, cmd);

        if cg(i,j) > bestpre
            bestpre = cg(i,j);
            bestC = basenum^X(i,j);
            bestG = basenum^Y(i,j);
        end
        if ( cg(i,j) == bestpre && bestC > basenum^X(i,j) )
            bestpre = cg(i,j);
            bestC = basenum^X(i,j);
            bestG = basenum^Y(i,j);
        end

    end
end
```

```
%% 画出根据不同 c,g 的准确率等高线图
[C,h] = contour(X,Y,cg,60:step:100);
clabel(C,h,'FontSize',10,'Color','r');
xlabel('log2c','FontSize',10);
ylabel('log2g','FontSize',10);
grid on;
```

综上所述，通过 SMO 算法的优化，RBF 内核的选取，和最优参数的调整，我们已经确定了最终我们所使用的分类器。与目前手写字符的分类器相比，或神经网络算法相比，该分类器显示了非常强大的性能，对不同手写字符样本都能表现出准确的识别率。在下一章具体分析。

4.4 学生个人成绩管理系统

在完成了核心算法之后。我完成了学生个人成绩管理系统。其内容有：UI 设计，试卷的录入和显示，试卷信息的统计，分题目显示每到试题，其题号和判别情况，以及每套统计完成试卷的保存和提取功能。这部分内容会在第六章中具体展示。

第5章 分类器的实验与结果分析

5.1 样本收集

我们搜集了在不同条件下的不同同学手写对号√和错号×，共 1835 个训练集，104 个测试集。这些样本包括正常情况下的，也包括在不同光照影响下(改变图像的 RGB 值)的，对号√和错号×与试卷文字相互重叠的，对号√和错号×具有很高特异性的。为了使训练样本具有很高的普遍性，能使分类器在有光照，噪声等影响因素下也可以具有高准确率。

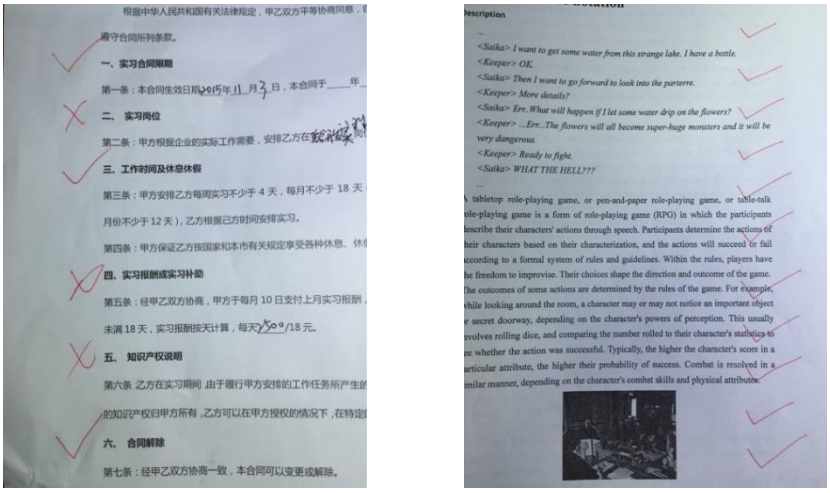


图 5-1 训练样本 1

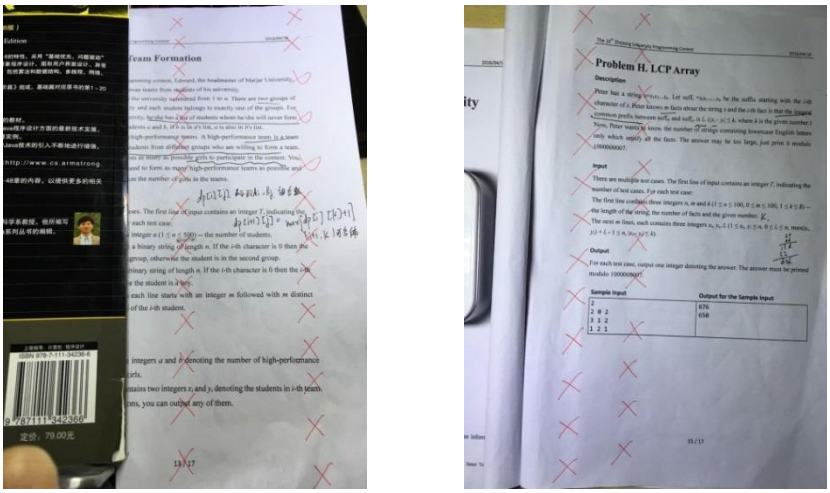


图 5-2 训练样本 2

从图中可以看到，样本的提取环境是受到较大影响的，试卷内容很杂乱有较

多干扰，试卷受到不同光照强度，明暗的影响。但是在这种情况下也可以很好地将首先字符提取并进行识别。

5.2 分类器测评

5.2.1 特征向量提取对准确率的影响

在实验过程中，我比较了基于 13 点特征提取的分类器，和基于改进的 102 维特征向量的分类器的识别率，如下表：

	1835 训练集,5-fold 交叉验证		1835 训练集，104 测试集	
	识别率	错识率	识别率	错识率
基于 13 点特征提取	98.4196%	1.5804%	98.0769%	1.9231%
基于改进特征提取	99.9455%	0.0545%	100%	0%

表 5-1 特征向量提取对准确率的影响

上表的测试结果都是基于分别选取最优参数的 RBF 内核，SVM 分类器的。我们可以看到传统的 13 点特征向量提取已经表现出了非常好的识别效果，对于 1835 训练集,5-fold 交叉验证可以达到 98.4196%的识别率。但是使用 104 测试集实际测试过程中，我们发现有 2 个点属于错识点。在我们实际识别试卷当中，大概 5 份试卷，会误识别两道题左右，这个效果就不是特别理想。但是使用改进的 102 维特征向量提取，可以做到对 1835 训练集,5-fold 交叉验证可以达到 99.9455%的识别率(只有 1 个点误识)，对于 104 维测试集，可以做到 100%的识别率。这样在实际识别试卷的时候就具有很高的可靠性了。

5.2.2 选取最优参数的 RBF 内核与其他核函数的对比

在分类器的设计过程中我还对比了选取不同核函数对训练结果的影响，最终选取 RBF 内核为我们分类器的核函数。测试环境同样是基于 1835 训练集，104 测试集，所使用的特征向量是优化过的 102 维特征向量，结果如下表：

	1835 训练集,5-fold 交叉验证		1835 训练集，104 测试集	
	识别率	错识率	识别率	错识率
线性核	95.8583%	4.1417%	97.0588%	2.9412%
多项式核，d=2	86.5940%	13.4060%	86.5384%	13.4616%
多项式核，d=3	54.8774%	45.1226%	59.6154%	40.3846%
Sigmoid 核	98.0926%	1.9074%	98.0392%	1.9608%
RBF 核	99.9455%	0.0545%	100%	0%

表 5-2 不同核函数的比较

从上表我们可以看出很多问题。首先我们所选取的 RBF 内核具有最高的识别率，而且唯一一个对测试集达到 100%的识别率。接着也可以印证具有一定参数的 Sigmoid 内核与 RBF 内核有一定的相似性，也可以达到较高的识别率。然后我们分析线性核，多项式核(d=2), 多项式核(d=3)，我们可以发现识别率是呈明显减低的，说明选取的核维度越大，反而识别效果越不好。实际上线性核就是 d=1 的多项式核。这说明了我们的特征向量空间，在一定程度上是线性可分割的，只用线性核就可以带来较高的准确率。这就直接说明我们的特征向量提取方法，可以很好地把不同手写符号很好地分割开来。横向比较如下：

我们都使用线性核对不同特征提取方法进行测评：

	1835 训练集,5-fold 交叉验证		1835 训练集，104 测试集	
	识别率	错识率	识别率	错识率
线性核，102 维	95.8583%	4.1417%	97.0588%	2.9412%
线性核，13 维	85.6131%	14.3869%	92.3077%	7.6923%

表 5-3 线性核下不同特征向量提取的比较

之前的比较是基于 RBF 内核的，13 点特征向量显示的识别率有一部分原因是因为 RBF 最优参数选取的结果。在最朴素的线性内核下比较，就直接反应出我们改进的 102 维特征向量的优势所在，即能更好的体现不同手写字符的特征，使其近似于线性可分割。

5.2.3 总体测评

5.2.3.1 大规模数据下的 ROC 分析

ROC（Receiver Operating Characteristic）曲线经常被用来评价一个二值分类器的性能。如下图就是一个二值分类器的 ROC 图：

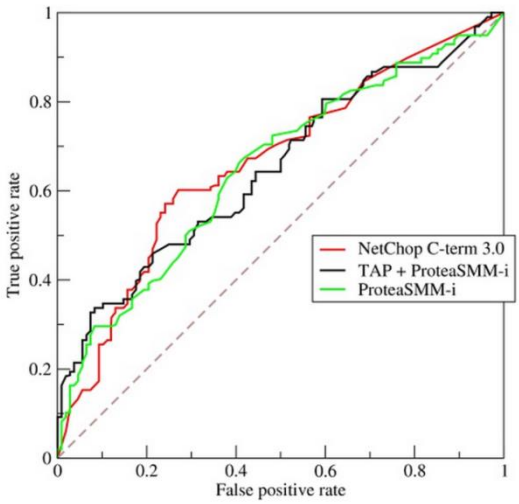


图 5-3 ROC 测试图

图中横坐标为 false positive rate（FPR），纵坐标为 true positive rate（TPR）。左上角的点(0, 1)表示的是 FRR=0, TRR=1，即所有样本都被分到了正确的类别，是一个最优的分类器；右下角的点(1, 0)表示 FRR=1, TRR=0，即所有样本都被分到了错误的类别，是一个最差的分类器；中间的虚线表示的是一个随机分类器。所以一个二值分类器在 ROC 图中呈现的曲线越靠近左上角，表明这个分类器具有更高的性能。

我对基于 RBF 核选取最优参数后的 SVM 分类器，对 1835 个样本进行测试后的结果，绘制成了 ROC 图，如下：

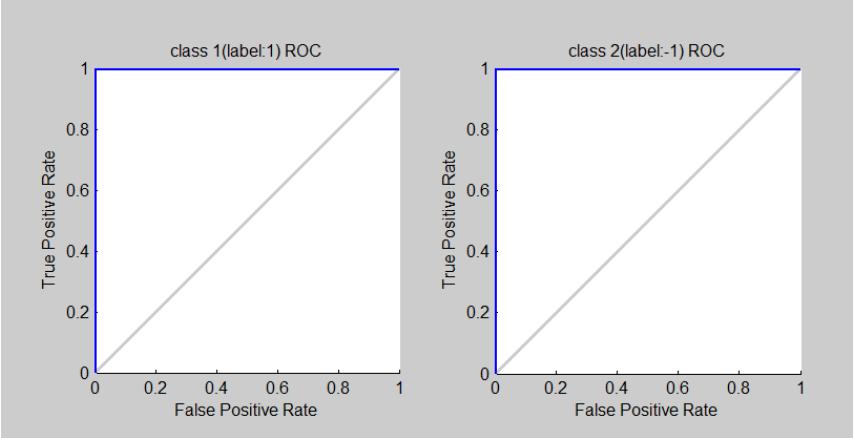


图 5-4 1835 样本的 ROC 图

从图中我们可以看到两类样本(标签-1 和 1)在 ROC 图中的结果都位于左上角(0,1)点，说明我们的 SVM 分类器表现出了较高的性能，几乎对所有点都做出了正确的识别。

5.2.3.2 与其他算法的识别比较

我将本文中所使用的优化后的 SMO 算法 SVM 分类器，与基于 BP 神经网络，基于 HMM 模型和基于 K-邻近算法分类器做了比较。如下图是平均识别时间的比较：

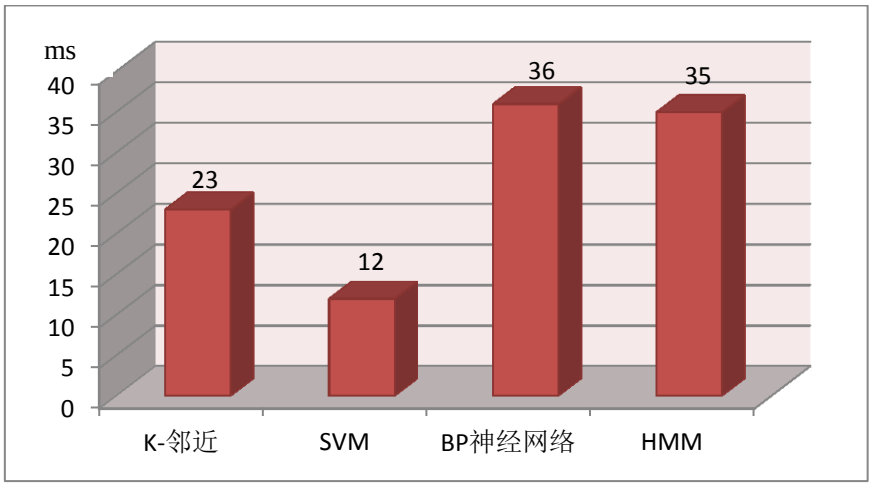


图 5-5 不同算法平均识别时间的比较

我们可以看到本文采用的基于 SVM 的分类器的平均识别时间是最优的，它继承了传统 SVM 算法分类的高效性。同时在不降低时间复杂度的情况下，使用了选取最优参数的 RBF 内核，从而大大提升了算法的识别效率。下图为优化过的 SVM 算法识别效率与传统 SVM 及其他识别算法的比较，我们可以看到使用 102 维特征向量提取与参数优化 RBF 内核所表现出的识别率上的明显优势：

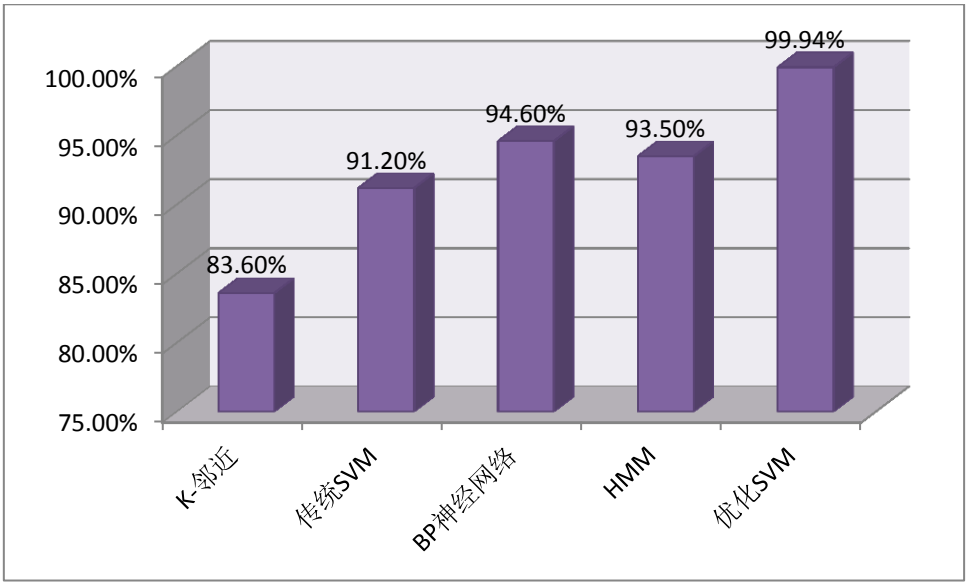


图 5-6 不同算法平均识别率的比较

5.2.3.3 在不同数据规模下的表现

下两图显示了分类器随着样本空间数量增长时，识别时间与识别率的变化，我们对不同规模的样本同样采用网格搜索和交叉验证(5-fold):

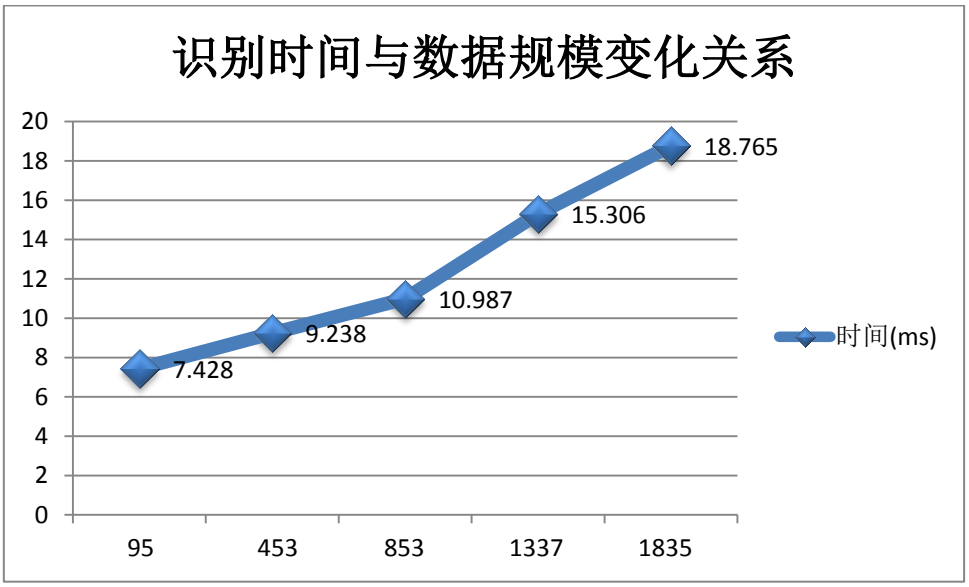


图 5-7 识别时间与数据规模的关系

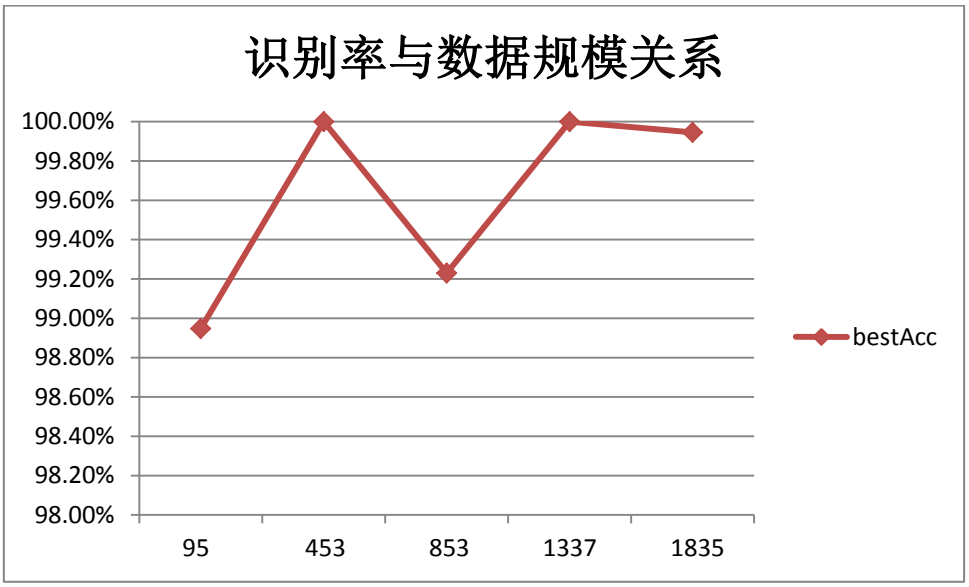


图 5-8 识别率与数据规模的关系

我们可以从图表看出，算法的识别时间是随数据规模呈线性增长的。但是 bestAcc(对于数据集进行不同的 5-fold 交叉验证所选出的最佳识别率)却与数据规模不是正比关系。这是因为交叉验证的 fold 数会直接影响到样本的识别率，如果对于数量过小的样本采用较高的交叉验证 fold 数，反而会降低其识别率。

不过总体上我们可以看到该分类算法的时间复杂度为 O(n)，是线性的复杂度。空间复杂度为训练时为 O(c)，因为训练使用的核函数的内积，所以只需要每次读取一个 102 维特征向量然后做点乘就可以，为常数复杂度；识别时空间

复杂度为样本的数量数，所以为 O(n)。对于大量数据也可表现出较高的效率，并且准确度不受数据量的影响。

第6章 学生试卷识别和管理系统展示

6.1 基本功能与使用

图 6-1 基本界面

如上图所示是我设计的学生成绩管理系统的基本界面。包括如下基本功能：

6.1.1.1 打开试卷

打开需要处理的试卷页面。每次只处理一套试题中的一张页面，系统会自动合并整套试卷。

点击打开按钮，如下图是打开试卷示意图。

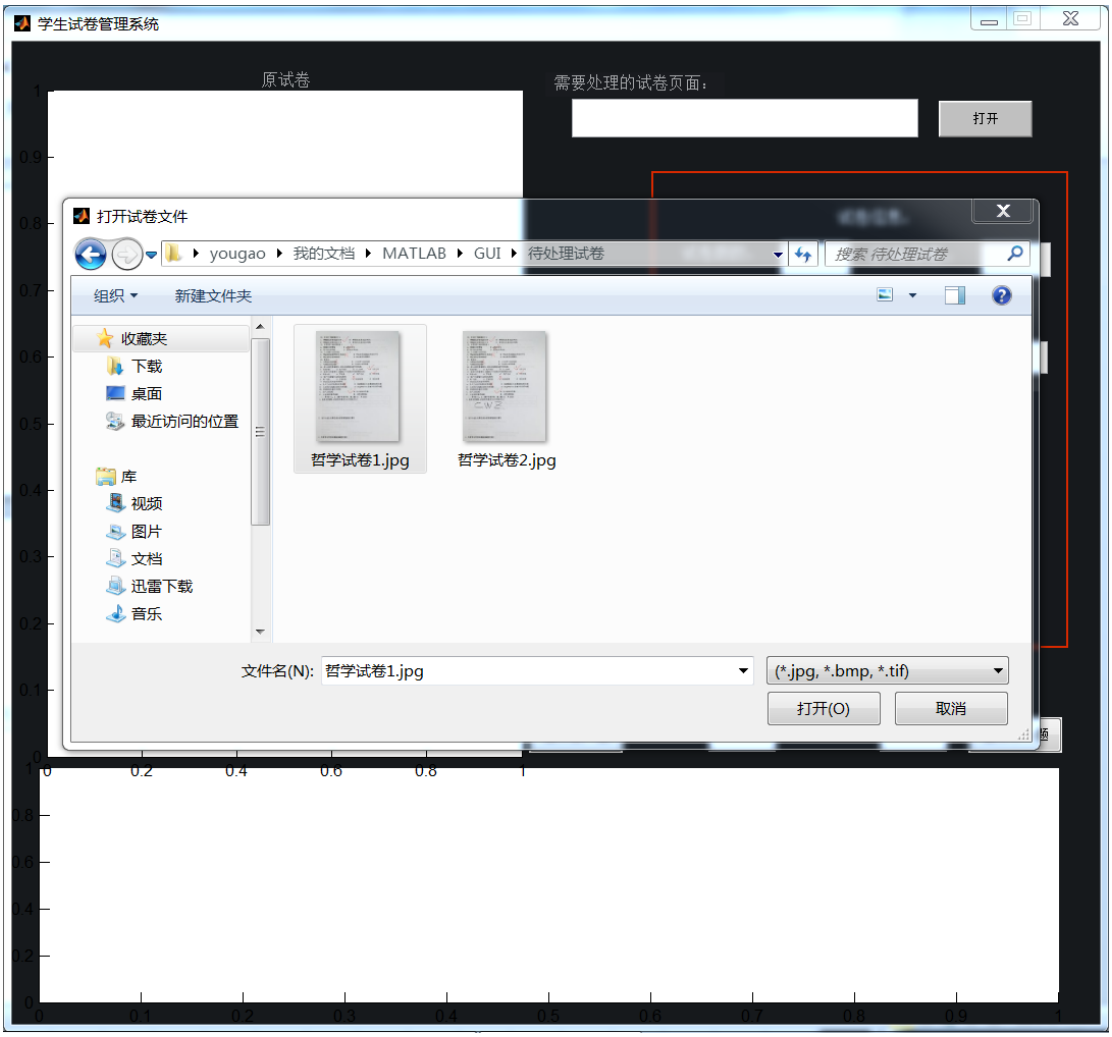

学生试卷管理系统
原试卷
需要处理的试卷页面：
打开
打开试卷文件
yougao ▸ 我的文档 ▸ MATLAB ▸ GUI ▸ 待处理试卷
搜索 待处理试卷
组织
新建文件夹
收藏夹
下载
桌面
最近访问的位置
库
视频
图片
文档
迅雷下载
音乐
哲学试卷1.jpg
哲学试卷2.jpg
文件名(N): 哲学试卷1.jpg
(*.jpg, *.bmp, *.tif)
打开(O)
取消

图 6-4 打开试卷

6.1.1.2 显示整张试卷

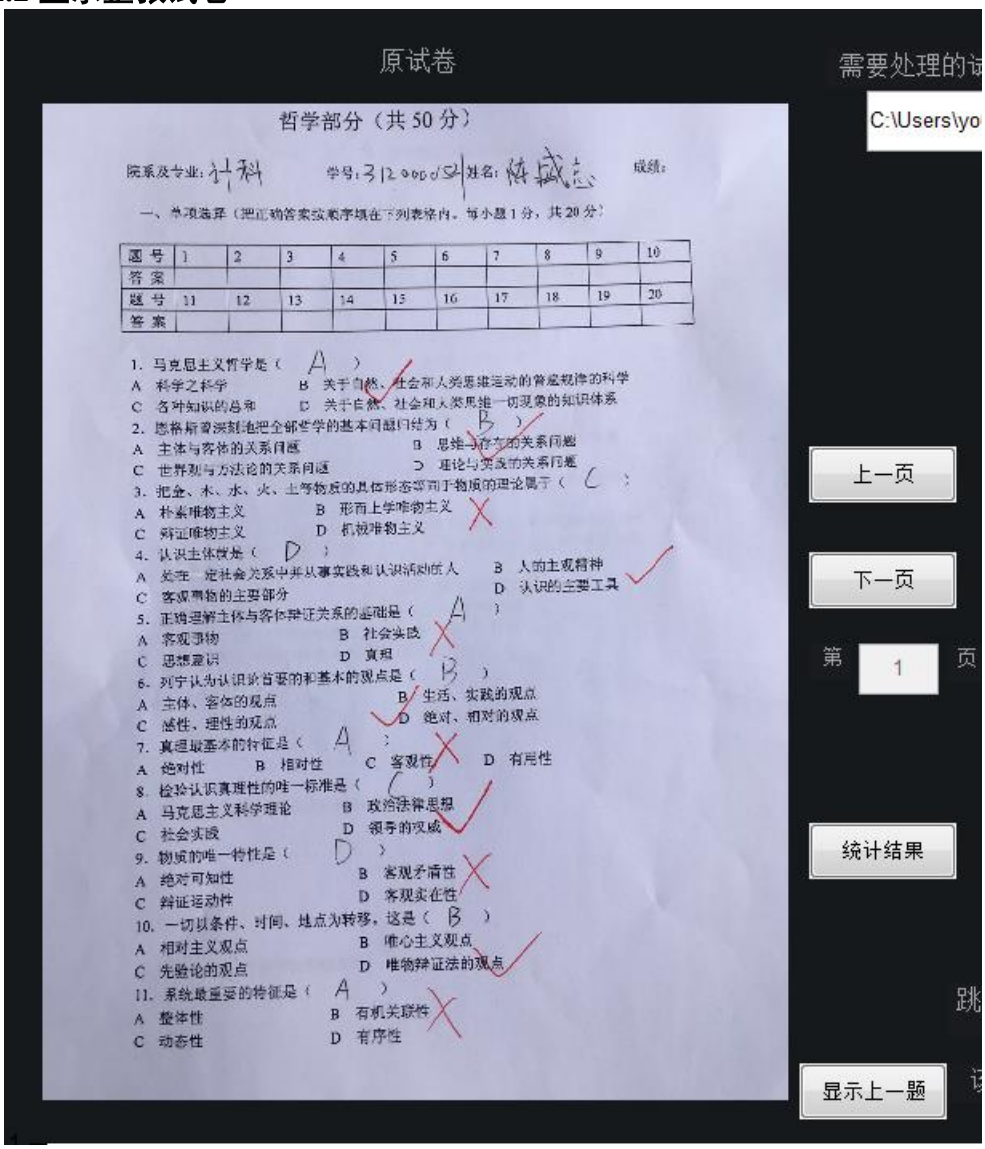


图 6-5 显示整张试卷

如上图是显示整张试卷功能，每打开一张新的页面会自动插入到整套试题中。选择上一页或下一页按钮可以翻页，并在下方显示页面编号。点击统计结果按钮统计该份试卷的信息。

6.1.1.3 试卷信息统计

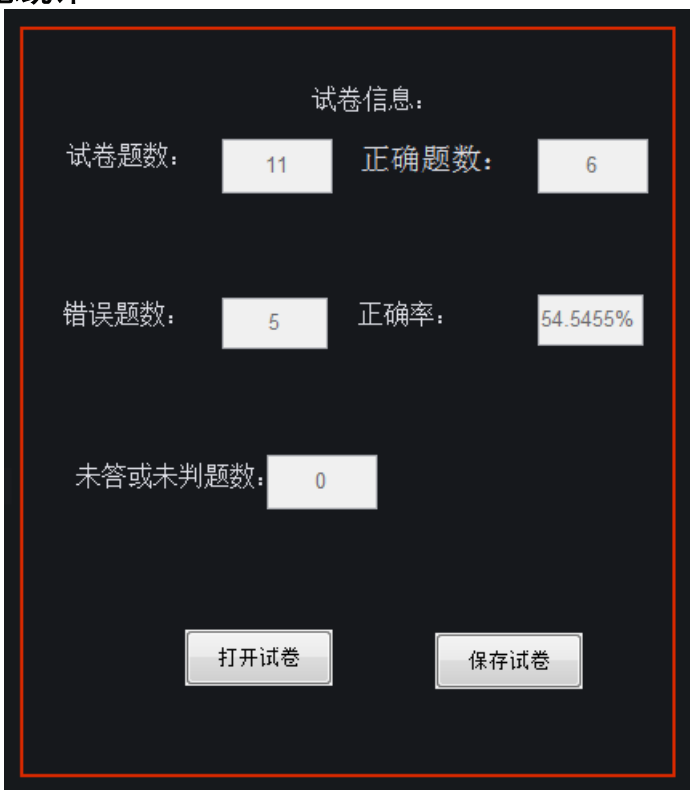


图 6-6 信息统计

如上图所示，点击统计结果按钮，学生试卷管理系统，自动进行版面分隔，处理，并识别出试卷上对应题目区域内的手写字符信息，然后进行统计。

6.1.1.4 试题分隔显示

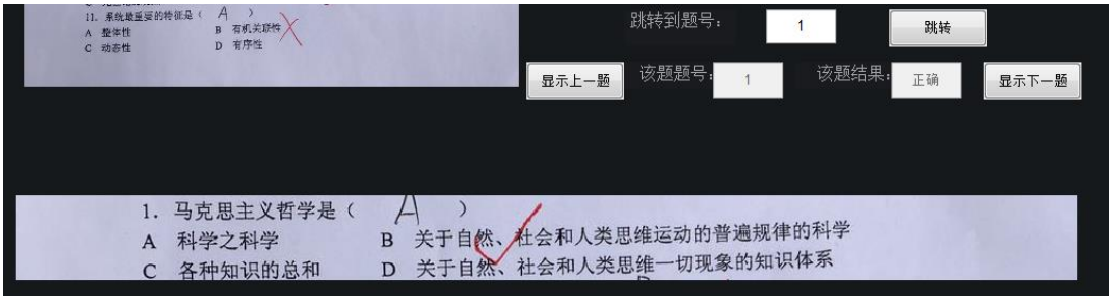
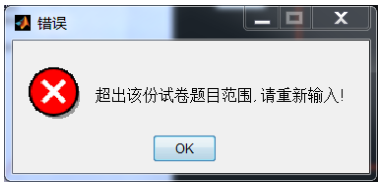


图 6-7 逐题显示

点击统计按钮后，在下面显示区域内会自动显示分隔好的试题信息。可以通过按钮选择上一题或下一题，在上方会显示试题编号，用户也可以输入题号自动跳转到制定题目。如果在试卷的范围内则会跳转到制定位置。如果超过题目范围会提示错误。



在下面的方框内会显示该题的识别结果。这个结果包括 4 种可能：正确，错误，未判，异常。如果系统检测到该题教师没有进行批改，则会显示未判，如果有异常情况：如在一道题区域内出现多个判别痕迹，则会显示异常。

当我们处理完一张页面后可以继续用打开功能处理下一张页面。

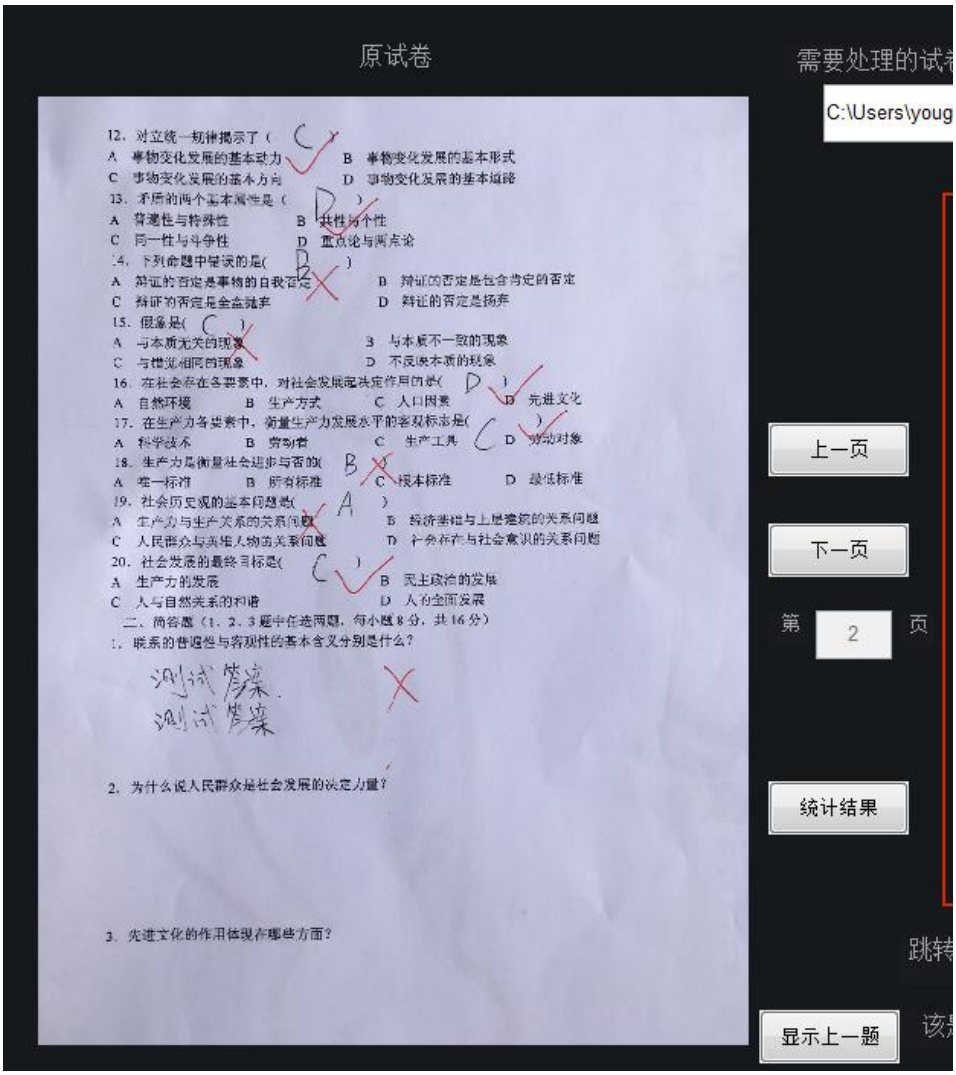


图 6-8 显示试卷第二页

此时显示的是第二页，点击统计后会和上一张页面的信息合并起来。并整体显示。

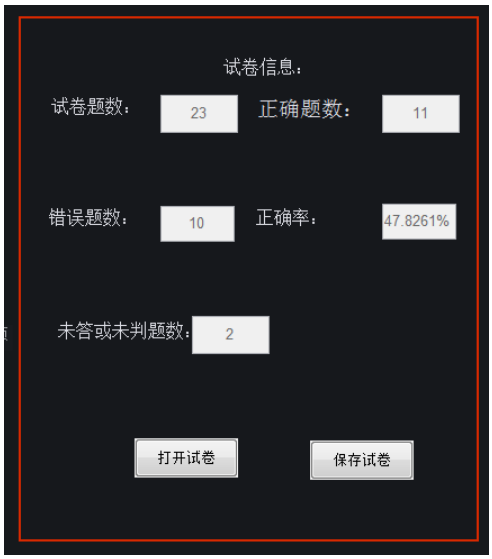


图 6-9 信息整合显示

这是显示的信息就是一共 23 道题的试卷整体信息。其中最后两道题未答，所以显示未答或未判题数为 2。这时使用显示下一道题功能就会连续显示整份试卷的所有题目。并且如果当前显示的题目所在页面为第一页，那么显示到下一题的时候，整张试卷也会随着翻到下一页显示。跳转功能也可以跳转到整套试卷内所有试题。

6.1.1.5 试卷的保存和提取

当一套试卷处理完成后可以点击保存试卷进行保存。

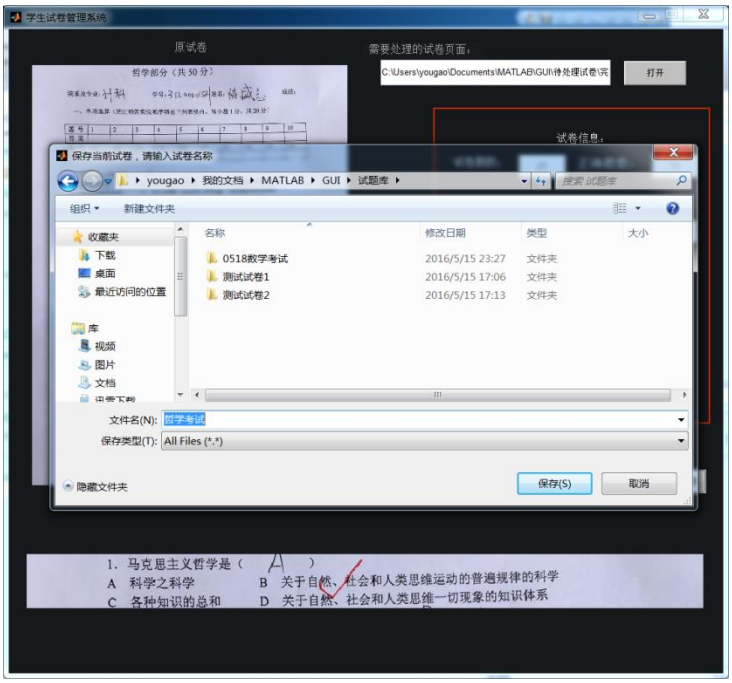


图 6-10 试卷的保存和提取

然后当用户需要再次查看该套试题是，选择打开功能就可重新调取整份试卷。

6.2 代码汇总

总体架构&用户界面	main.m
13 点特征向量提取	GetFeature.m
102 维特征向量提取	GetFeatureNew.m
特征向量性能比较	GetFeatureTest.m
读取数据&生成训练集和测试集	program1.m
网格搜索&交叉验证选取最优参数	SVMcg.m
SVM 训练和结果预测	Training1.m
核函数&训练器的性能比较	test.m

6.3 工作展望

目前的开发是在 PC 端的。我们试卷识别和管理系统相比于同种产品的一个优势是可以处理手机拍摄的试卷图片，而别的版面识别分析系统只能处理扫描仪扫描的照片。我们可以处理光照，环境等带来的影响。所以今后的一个主要工作就是基于现有的核心代码，开发手机端的 APP 应用，充分发挥产品的优势。

其次要完善现有的功能，比如识别出试卷的卷头信息，如什么时候进行的什么考试，考试人的姓名，学号是多少，这需要对卷头信息进行分隔，并且对其进行 OCR 识别。可以识别教师的评分，这需要准确分隔出评分位置，然后进行手写数字的识别。

总之，目前的系统是一个初步的实现。需要通过后续的努力，设计出功能更强大，更通用的试卷识别和管理系统。

参考文献

[1] 李宏峰 基于 OCR 的试卷版面理解 安徽工业大学.2009

[2] 许雁飞, 陈春玲, 陈夏梅基于 OpenCV 的脱机手写字符识别技术南京邮电大学计算机学院软件学院.2011

[3] 李辉熠 基于 HMM 的脱机手写体字符识别 长沙理工大学.2011

[4] 沈茜 基于神经网络与 GPU 的手写数字识别及其试卷管理 苏州大学.2011

[5] 常昌 图像特征提取方法研究及应用 华中科技大学.2009

[6] 董慧 手写体数字识别中的特征提取和特征选择研究 北京邮电大学.2007

[7] 乐海，胡伟 手写体数字识别系统中一种新的特征提取方法四川大学学报(自然科学版).2007

[8] 杨文涛 分数傅里叶变换在数字图像处理中的应用研究 华中科技大学.2007

[9] Platt J C.Fast training of support vector machines using sequential minimal optimization. . 1998

[10] Platt J C.Fast Training of Support Vector Machines using Sequential Minimal Optimization. Cambridge, MA: MIT press . 1999

[11] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin A Practical Guide to Support Vector Classification National Taiwan University, Taipei 106, Taiwan.2010

[12] 李琼，董才林，陈增照，何秀玲 一种快速的 SVM 最优核参数选择方法 计算机工程与应用.2010

致谢

首先非常感谢吴健老师对本次项目提出的意见和指导。感谢顾盼学姐对项目及论文的建议和帮助。也感谢同组成员李毅儒在本次项目中的认真合作，使我们可以共同顺利完成本次项目。最后感谢学院老师对项目的批准和支持。

本科生毕业论文（设计）任务书

一，题目：基于 SMO 算法 SVM 的试卷信息统计

二，指导教师对毕业论文（设计）的进度安排及任务要求：

需要同学调研国内外近几年试卷管理系统和手写字符识别相关研究成果，对手写字符识别的知识点进行重点研究，研究手写字符识别的现状和常用方法，与本项目进行结合，设计出具有针对性，创新性，具有高识别率的识别方法。与同组同学合作，完成试卷识别和管理系统。需要翻译图像识别相关领域的国外文献一篇。并且要求报告书写符合写作规范，表达清晰，字数在 10000 字以上。

起讫日期 **2016** 年 **3** 月 **25** 日 至 **2016** 年 **5** 月 **31** 日

指导教师（签名）＿＿＿＿职称＿＿＿＿

三，系或研究所审核意见:

负责人（签名）＿＿＿＿

年　月　日

毕 业 论 文（设计） 考 核

一，指导教师对毕业论文（设计）的评语：

论文针对手写笔迹开展研究，选题具有重要应用价值，论文通过轮廓特征、笔画密度特征等特征向量建立一个基于 SMO 的 SVM 分类器，得到良好的分类结果。

指导教师(签名）

年 月 日

二，答辩小组对毕业论文（设计）的答辩评语及总评成绩：

成绩比例	中期报告占（10%）	开题报告占（20%）	外文翻译占（10%）	毕业论文（设计）质量及答辩占（60%）	总 评成绩
分值	9	19	9		

答辩小组负责人（签名）

年 月 日