

****

**J I A N G S U U N I V E R S I T Y**

**本 科 毕 业 论 文**

基于Java的视频动作识别系统

**Java - based Video Action Recognition System**

学 院 名 称： 计算机科学与通信工程学院

专 业 班 级： 计算机1302班

学 生 姓 名： 曾 彪

指导教师姓名： 詹永照

指导教师职称： 教 授

2017 年 5 月

基于Java的视频动作识别系统

专业班级：计算机1302班 学生姓名：曾彪

指导教师：詹永照 职称：教授

**摘要** 视频动作识别是目前计算机视觉领域发展的一个热点方向，在计算机视觉领域，很多技术都已经逐渐趋于成熟，比如说人脸识别，早在前几年各大公司就相继宣布人脸识别率达到99.9以上，标志着机器学习的应用已经不再是遥不可及的童话，与此同时，随着技术的发展和社会生活的需要，越来越多的学者开始转向于研究它的另一个分支——视频动作识别。

视频动作识别的基本原理和人脸识别一样，但由于人体动作的特征性，整个动作是由许多帧连贯而成的，每帧之间是上下文相关的，采取的特征提取方法等也就和人脸识别有所不同。

由于以往的论文基本着重于理论的介绍，很多初学者看完后往往是晕头转向，更多的细节实现更是一带而过，对于机器学习这一庞大的知识体系，很容易让人抓不到重点，以至于让后面的研究者花费大量的时间在环境以及平台兼容性上，而不是专心研究算法。基于各方因素，本文的主要研究内容主要分为以下几点：

1.采用了目前世界上比较流行，但在机器视频领域选用的较少的Java语言作为开发平台，并详细介绍了系统开发的结构和思路。

2.设计了一个协同训练算法，对两个分类器进行协同训练集成融合，选取置信度较高的视频数据作为训练样本进行迭代训练，以解决标记数据不容易获取、标记代价大等问题。

3.由于在视频里一个动作往往是可以分解成很多个小的动作的，以一定的时间间隔作为分割，每个时间范围内的视频序列往往已经足够用来表示一个动作，因此我们以此作为依据，一个视频就可以划分成若干个动作的集合，这样也可以用来解决视频标本数据不足的问题。

4.原理同上一点一样，既然训练的时候一个视频可以提取出若干个动作特征，那识别的时候同样也可以对一个待识别的视频数据提取，然后对每个动作特征分别进行识别，最后采用一定的投票算法选取出一个可能性较高的结果。

。

**关键词：**动作识别 机器学习 行为识别 机器视觉 Java

**Spark Model Optimization For Large Data Applications**

**Abstract** No

d.

**Key words** Big Data Machine Learning RNNLM Spark

# 第一章 绪论

## 1.1 课题背景及意义

视频动作识别是机器学习的一个分支，本质上都是运用大量的有标记的数据去训练分类器，然后再用训练出来的分类器来进行样本的识别，但是与人脸识别不同的是，和目前比较热门的人脸识别相比，无论是国内还是国外，视频动作识别这方面的研究都还比较少，难度也相对较高，而且目前比较流行的平台大多是基于Python、matlab、C++的，基于java的则更是少之又少了，人脸识别所需要的输入数据只是静态的图片，只要提取出当前图片的特征，再进行训练和比较，就很容易了。虽然也是视频，我们经常见到的通过摄像头扫瞄人脸，然后进行实时跟踪的例子也基本上是属于人脸识别的范畴，因为它相当于把视频拆分成一帧一帧的图片，然后只要在每一帧中分别识别就可以了。而视频动作识别则不仅仅需要识别出每一帧图片，而且每帧图片之间还是上下文相关的，它们之间存在着某种关联，只有特定的组合才能成为一个动作。另一方面，目前人工智能领域比较热门的动作识别方法主要是通过大量的有标记数据样本来进行训练，这样的话虽然最终的分类器的精度比较高，但是有标记的样本往往需要具体样本所属领域的专家进行手工标记，这样一来要获取大量有标记的样本，所要耗费的人力财力极高。

## 1.2 技术的应用领域

在人们的日常生活中，信息可以说是无处不在，各种各样的信息充斥着我们的社交媒体，一方面，信息时代给我们的生产生活带来了极大的便利，手指随便一动，各种我们想要的信息就出现在屏幕上，另一方面，信息的增长速度也逐渐超出了人们的控制，各种我们想要的不想要的信息，都展现在我们面前，尤其是一些虚假信息的产生，不仅浪费了人们大把大把的时间，还会对人的判断产生误导。如果是文字的信息还好，基本上随便一瞥，大概的信息便了然于心，即便是交给计算机云处理，也极为的方便，只需要设置一些关键字，便可以很便捷地过滤出我们想要的信息。但是对于视频信息，情况则又不同了，互联网上视频信息也是成千上万，鉴于视频的特征性，如果想要知道视频里有什么信息的话则需要把视频完整的播放一遍，这样一来，如果视频很长的话，浪费的时间可是不可忽视的。

基于此，一种能够自动识别出视频里面动作的系统就应运而生，我们可以把视频动作识别这种特别耗时的操作交给计算机去处理。计算机系统首先从一大堆有标记的视频数据中提取出相关视频中人体动作的特征，然后根据分类器的训练方法用这些特征去训练出一个强大的分类器，后面的事情就方便了，当我们需要知道一个视频里面的动作类别时，就不用人工地把整个视频全看完，直接把视频提交给计算机，计算机从视频中提取出动作特征，运用分类器进行识别，把视频的类别告诉用户，当视频非常多时，也可以交给计算机去批量处理，从而省下了大量宝贵的时间。

应用人工智能使计算机也像人类一样拥有“视觉”，从计算机的角度去分析视频，把所需要的信息展现在我们面前，初期的目标是设计一个基于java的软件，用户通过手动导入相关视频，软件系统使用本课题中的相关方法进行分析，给出结果，使计算机能够帮我们人类快速识别出视频所包含的信息，并把结果反馈给用户以供决策，尽管这些步骤看上去非常粗糙、繁琐，但是我相信在不久的将来，我们甚至可以进一步地把这一成果应用到互联网领域，到那时候互联网上的每台计算机可以对识别出的视频贴上“标签”，而用户可以预先设置好想要看到的标签和不想看到的标签，计算机可以根据用户的需求进行推送和屏蔽，完全不需要人工的干预。

当技术发展到一定阶段时，还可以往更深的方向探索。比如我们可以建立一个庞大的服务器，对互联网上的视频进行批量的识别，并贴上视频的类别标签，方便用户进行查找和鉴别，当好友向你发送一个视频时，系统会先进行扫描，把视频的内容信息通过简要的文字描述给你，你再选择查看或者忽略。

## 1.3 课题结构及研究内容

本文工作。

# 第二章 相关技术研究现状

## 2.1 相关技术研究现状

视频动作识别是一个具有重大经济效益和社会价值的研究项目，具有广阔的应用领域，能够大幅度提升各种监控系统的智能程度。最早的研究是从西方开始的，一开始是应用于监控领域，设计出一个能够检测人体轮廓和动作轨迹的系统，以提升安保能力。后来学者们又尝试分析人体的行为模式，试图从一系列的动作序列中解读出动作的规律，让计算机理解视频中人体动作的含义。

随着模式识别相关技术的不断发展与成熟，视频动作识别的相关研究也在逐渐深入，近些年来，国家和社会对科研项目越发的重视，投入了大量资金，鼓励各高校、科研机构等加入科研领域。于是越来越多的人开始加入视频动作识别领域的研究，并取得了大量的成果。刘博等人[1]研究了基于关键姿势的人体动作识别，从视频中过滤掉视频中的背景，提取出人体的轮廓，再用模版匹配的方法去进行预测。吴文军等人[2]提出了一种提取人体的各个关节间的位置信息的变化关系，不同动作的各关节点的变化是一同的，通过这一特点进行特征提取，从而进行识别。Schuldt等人[3]研究了应用时空特征进行识别，首先提取出较为稳定的几个时空关键点，再选取一个适当的时空区域进行特征提取，再用支持向量机的分类器SVM进行识别。阵渊博等人[4]提出了一种具有判别性的图模型，充分利用有标记数据和大量无标记数据，通过共享稀疏系数剔除无关项，提升识别率。

## 2.2 本课题面临的问题与挑战

虽然目前国内外在动作识别领域的研究在不断深入，各种新的技术和方法相继被提出，值得注意的是由于人体动作的复杂性，各种动作千变万化，然而对于计算机而言，其识别模式是固定的，它所能看到的只是一串串的数据，对于每个动作的理解，却一无所知，目前世界上几乎所有的动作识别的相关研究，都是一种近似的模拟，和人类对于动作的判别方法是截然不同的，一个人体在空间整体的细微移动，在人眼看来，基本是没什么变化的，因为这种无关的信息自动地被过滤掉了，然而在计算机看来，这两帧的数据已经完全不同。正是由于计算机采用了模式识别的方法，需要通过计算不同动作间的差异来进行区别，而每个相同的动作，即使是同一个人，也可能存在很大差异，这就给动作识别带来了很大的困难。另外由于视频的标注成本很高，很难找到充足的标记数据来进行训练，而较少的训练样本必然会影响视频动作的分类器的识别效果。

## 2.3 小结

# 第三章 一种新的技术或方法

一种方法描述

相关实验结果比较分析

# 第四章 原型系统设计与实现

## 4.1 项目实施的对环境和社会的影响

视频动作识别系统投入商用以后可以广泛的应用到监控、安保、大数据分析等领域，由于系统是一个软件层次的概念，对环境没有任何的负面影响，且代码是友好的，没有任何的破坏和攻击性质。

现在的互联网越来越发达，人们都喜欢把视频上传到网上分享给朋友们，但是越来越多的视频堆积在服务器上，很难有序的进行管理和分类，只能通过上传者手动地加上一些视频的描述信息，但是很多人为了博取眼球，故意把描述写的很夸张，甚至有的时候是虚假的标题，有的更是一点描述都没有，只有单独的视频，这时候当我们想要找到目标视频时就会很费劲。还有一些不法之徒为了达到自己不可告人的目的，在互联网上大量传播违法的视频，诱导别人走向歧途。

有了视频动作识别的概念，对于上面这些案例，可以在视频上传的时候先检测一下，运用视频动作识别系统预测出视频里面都包含了哪些动作信息，并自动给视频打上标签，对于不合格的视频，则禁止用户上传，并给予警告。这样一来就可以营造一个安全干净的互联网环境，对于我们收到的视频也可以提前用系统作一个识别，通过了以后再选择接收。另外视频动作识别系统还能应用于手语翻译，先识别出手语者的每一个手势动作，然后把每个动作对应的意思翻译出来，帮助不能发声的人们进行正常的交流。

每一个技术的发展都不可能是孤立的，它不仅是多个学科门类知识杂糅的产物，也能在不同的领域中得到升华，从而促进相关产业的发展，相信在不久的将来，视频动作识别将成为一个社会生活中不可或缺的技术。

## 4.2 项目实施的技术和经济可行性分析

本课题是开发一个基于java平台的软件系统，用于视频的动作提取和识别，用到的技术有：

1.基于旋度的视频动作特征提取方法。通过计算出视频中的每一个特征点，选取出一帧中在设定的阀值范围内的所有特征点，计算出一个合适的时空区域，在这个区域中提取特征。

2.设计一个协同训练算法，选取置信度较高的样本作为训练数据。先利用两部分不相交的数据集分别训练两个分类器，再用这两个分类器去预测未标记视频数据的类别，并将置信度较高的数据加入到另一个分类器中进行训练，再对剩下的数据进行迭代，直到数据为空。

3.利用opencv中集成的一些比较经典的方法来简化系统的设计。OpenCV是一个开源的计算机视觉库，里面集成了许多模式识别、图像视频处理的常用方法，为我们系统的开发节省了很多的工作。

4.基于java的用户操作界面的开发技术。一个完整的系统不仅要实现应有的基本功能，还需要一个漂亮、操作简单的用户界面，采用java开发的系统易于平台的移植，并且功能强大，简单易用。

经过众多学者的不断创新与探索，动作识别的各方面的技术都已逐渐趋于成熟，项目的实施的技术难题也不在话下，至于经济方面，该项目的应用领域广阔，拥有无限的商机，随着人们生活水平的提高，对科技的需求也越来越强烈，这将会吸引一大批的企业投资，且软件系统的部署几乎不需要多少成本，一次开发成功，多个计算机系统均可快速部署，可见其经济可行性还是很高的。

## 4.3 系统需求分析

本课题设计的是一个基于java的视频动作识别系统，开发平台主要是eclipse，视频动作识别系统要求是要能够对输入的视频进行相应的处理，能够判断出视频中含有的人体动作的类别，在本次研究中，我们采用的是比较经典的kthdata人体动作视频库，里面含有6种不同的人体动作，系统所要实现的是给用户提供一个友好可视的用户操作界面，在界面上通过按钮的点击完成视频的识别功能，并在具体的位置显示出处理的结果。

在传统的动作识别的研究的基础上，本课题更加注重的是系统的实现，因为以往的动作识别的研究讲的大多都是原理，是一个比较笼统的概念，至于实现的细节，往往提的很少，然而由于视频动作识别的复杂性，很多细小的部分都是不可忽视的，有的时候一个小小的参数设置的不同，都可能会导致特征提取、识别结果的截然不同。鉴于此，本次课题选择了一个比较具体的语言java，一个在动作识别领域用的很少的语言，给后面的java动作识别研究者一个铺垫，少做一些麻烦的事。

根据视频动作识别流程的初步构思，系统需要实现以下几个功能：

1.特征提取功能。该模块用于将本地所有的视频文件进行以关键点为中心的时空特征提取，并将每个视频的特征保存在本地的固定目录下。

2.基于模版和基于概率统计的两个分类器的设计。设计两个不同的分类器对视频的特征样本进行训练，并将训练模型保存。

3.设计一个协同训练算法，先用有标记的视频样本去训练出两个不同的分类器，再用这两个分类器分别对无标记的数据进行预测，并将预测结果中置信度较高的视频样本加入到另一个分类器中进行训练，使用这样的方法来充分利用无标记数据提高分类器的识别率。

4.预测功能。用户选择一个分类器进行预测，弹出选择视频的窗口，通过浏览文件目录找到需要识别的视频，确定后系统进行特征提取，并调用相应的分类器进行预测，在提示区域显示预测结果。

5.计算一批视频的识别率。这个功能用来判定分类器的识别效果，可以批量地对一组视频进行识别，并自动把识别结果和真实类别比较，计算出各个视频类别的识别率以及分类器的总识别率。

6.用户界面除了有实现以上几个功能的控制按钮外还需要能够实时显示各个操作的提示信息以及视频特征提取过程的信息、关键点的监测（在视频中用蓝色点标注）等，以便于调试和监控系统的运行状态。

## 4.4 系统设计

本系统是一个基于java的纯客户端应用，所有的功能都集成在客户端上，不需要部署服务器，也不需要联网，用户直接通过客户端上的按钮等就可以进行相关操作。视频动作识别系统的流程如下：



视频动作识别系统按上面的几个主要的运行流程来看可分为特征提取模块、SVM分类器训练模块、SVM预测模块、KNN预测模块（由于KNN分类器是通过计算最近的K个特征得到预测结果，所以不需要训练，只需把所有训练样本特征融合起来就可以用了）、协同训练模块以及人机交互界面模块。系统的模块功能划分如下图所示：



为了方便对比，本系统设置了两种训练模式，一种是全监督训练模式，训练的样本全部采用的是有标签数据，执行流程如图所示：



还有一种是半监督协同训练模式，先训练一个SVM分类器和一个KNN分类器，然后分别对一组没有标记的视频数据进行预测，将各自识别出的置信度较高的样本加入到另一个分类器进行训练以提高分类器的识别率，其执行流程如图所示：



## 4.5 系统实现

在本节中，主要阐述的是本课题研究的视频动作识别系统的实现细节，我将在这节中详细介绍本系统的环境配置、数据集、用到的一些算法和技术、各模块和用户界面的实现等。

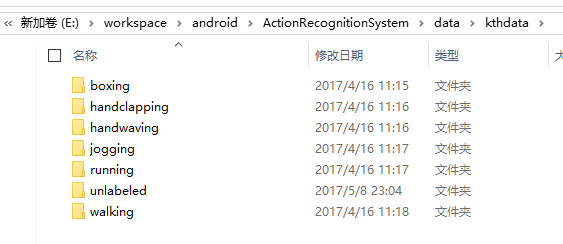
### 4.5.1 系统环境的配置

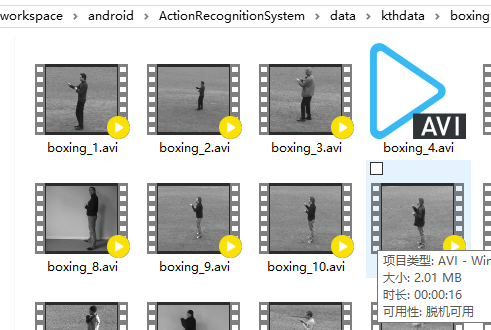
在本次课题中我使用的java运行环境是64位的jdk 1.8.0\_91 ，java开发环境用的Eclipse，在java平台上编写视频动作识别系统还需要一个工具Opencv，OpenCV是一个跨平台的计算机视觉库，有了它我们可以很方便地调用一些现成的方法来进行视频、分类器等的处理，OpenCV在java上的配置比较简单，直接找到OpenCV安装目录build\java里面的一个opencv-249.jar包导入到java项目里面就可以了。

本系统由于要对视频文件进行处理，还必须将OpenCV目录下build/bin目录下的opencv\_ffmpeg的dll文件（有2个，一个是32位，另一个是64位，比如我的是64位的环境，opencv2.4.9，就用了C:\mysoftware\opencv\build\x64\vc12\bin目录下的opencv\_ffmpeg249\_64.dll文件）复制到电脑上jdk的安装目录下的bin目录（以我的为例，就是C:\Program Files\Java\jdk1.8.0\_91\bin）下面才可以。这个问题真的很崩溃啊，在VisualStudio里只要配置好Opencv环境就一点问题没有了，但是按照网上的教程配置好java的Opencv的环境后，编译一点问题没有，但是死活就是识别不出视频文件，而且网上了好多文章都没有讲到还要配置这个dll文件的，对于以前从来没有接触过这方面知识的小白来说简直要抓狂了有没有，最后还是在看到国外一个有点类似问题的博客之后获得的灵感。

### 4.5.2 数据集的准备

本次课题选用了kthdata人体动作视频特征集，其中有boxing、handclaping、handwaving、walking、jogging、running等一共六种不同的人体动作，每一类动作各有大约100个视频，将种类视频归类到相应的文件夹下，另从各类视频中随机选取一部分放到unlabeled文件夹下，并将所有视频放到项目根目录下的data\kthdata文件夹下，数据集准备完成。

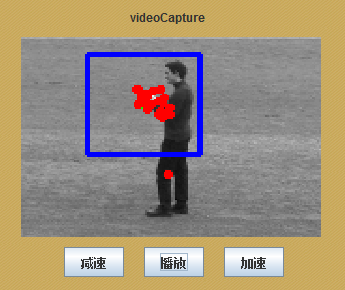




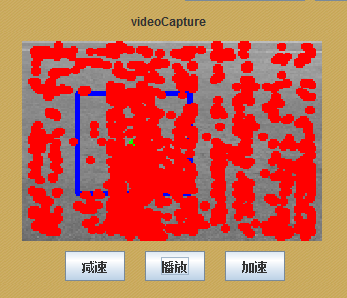
### 4.5.3 视频人体动作特征的提取

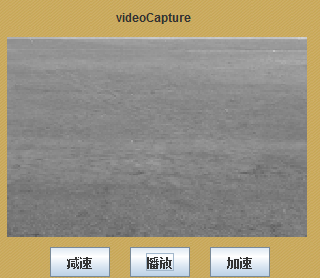
人体动作特征的提取是整个系统最为关键的一个步骤，我们所提取的特征在某种程度上就代表了这个视频的身份信息，特征信息提取的越准确、越完整，对后面的分类器训练和视频的识别，其效果也就越显著。在此我们提出了一种基于密集特征点的时空特征提取方法，它将最大程度地利用上视频中每个动作的关键信息。

首先我们计算出每一帧的光流信息，光流可以用来表示视频中物体的运动信息，使用OpenCV的Video.calcOpticalFlowFarneback函数可以计算出视频中的每帧图像中每个像素点的位置变化情况，在这里，我们将像素点位置在x方向和在y方向变化值的平方根称作旋度，然后找出每帧图像的旋度最大值，然后设置一个旋度百分比值0.2，用旋度的最大值乘以这个百分比就得到每一帧的旋度阀值，只有旋度大于旋度阀值的像素点才被登记为关键点（在图中用红点标记），这样做是为了消除一些噪声，因为有的时候由于相机的一些细微移动或光线的突然变化等就会产生一些额外的噪声，影响到关键点的识别，我们可以根据情况选取一个适当的阀值来把这些噪声过滤掉，留下那些主要的人体动作的特征点区域。



在这里要特别注意的是，由于有一些帧并不含有人体动作，但是却仍然可能含有很多噪声，如果按上面的处理方法的话就会把噪声中的最大旋度值的20%视作旋度阀值，从而提取出一大堆噪声的旋度值，并把这帧当成有关键点，这显然不是我们所希望的，由于噪声的旋度一般比较小，解决的方法是提前设置一个合适的固定旋度阀值，此处我们设置一个比噪声的旋度大很多的值200，当一帧中的最大的旋度比这个旋度阀值小时，就把最大旋度设为这个固定值，这样可以有效地把那些噪声过滤掉，效果如图。



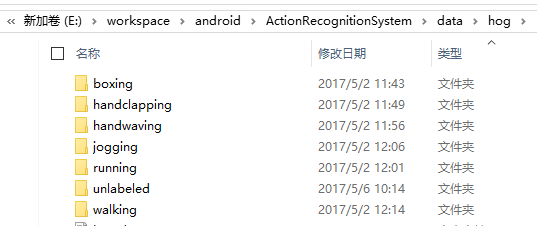


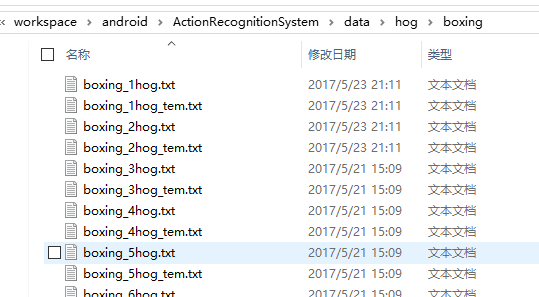
获得了每帧的关键点之后再把所有点的x、y值分别加起来除以关键点的数量，就可以算出中心关键点，在图中用绿色的点标记，从第一次检测出关键点的第一帧开始，我们选取一个以中心关键点为中心的固定大小的矩形区域（图中用蓝色线条标记），在这个矩形区域里提取每个像素点的HOG等特征值串联成一个向量，时间区域上我们选取十帧，每一帧的空间采样区域都和第一帧一样，不随目标物体移动，也就是说矩形区域只在第一帧里面计算，后面的九帧直接利用第一帧的矩形范围进行特征提取，这样在跑步的动作里每帧中人体在矩形区域中的相对位置是不一样的，可以把物体移动的速度信息也间接的表示进去。

考虑到相邻的两帧之间的相似程度较大，因此我们可以跳过其中一帧，按1，3，5，7，...，20帧的序号进行特征提取，总共提取出十帧，最后我们把这十个特征向量合并成一个，作为一个人体动作的特征向量表示。

在kth数据集中每个视频都是有一个人在一直不断重复地做着同一个动作，结合上面的方法，我们可以把这些动作分解开来，每20帧左右就可以提取到一个视频特征向量。这样一来可以增加训练分类器的样本数量，解决样本过少的问题；二来进行视频动作预测的时候可以对一个视频中的所有特征向量进行分别预测，最后进行一个结果统计，选取概率最大的那个预测结果，减少一个特征表示一个视频所带来的失误率。

由于视频特征的提取也比较耗时，特征提取完毕后我们直接把特征按视频的文件名和分类等保存到项目的data/hog目录下，后面分类器训练的时候直接读取即可。另外提取特征时如果不小心点击了停止或程序突然退出等，提取的特征就不完整，这样的数据用来训练会导致异常，所以提取时我们先保存到一个临时文件中（文件名用tem标识），顺利提取成功后再拷贝到另一个文件中，训练时用的是这个拷贝的文件。





### 4.5.4 基分类器的构建

在分类器训练之前，我们先进行特征数据的合并，根据需要选择一定数量的训练样本，读取图\*\*\*中的若干个文件，将其合并成一个文件，同时根据样本的类别生成对应的标签文件，将特征数据和标签数据分别读入到两个Mat矩阵中，作为分类器的参数，最后调用分类器的train函数完成训练，并将训练出来的模型保存到项目的data目录下。

本课题中我们每个类别分别用30个视频样本训练了两个基分类器，一个是SVM分类器，另一个是KNN分类器，用于对比两个分类器的识别效果以及协同训练算法的实现。

### 4.5.5 视频动作预测

该部分首先通过按钮调出的浏览窗口选择需要预测的视频，然后系统使用上面介绍的特征提取方法对该视频提取特征向量，每个视频可能会提取到很多个特征，我们在这里调用相应的分类器对每一个特征进行一一识别，再从所有的识别结果中进行统计，选择一个符合的结果最多的分类作为最终的预测结果。

### 4.5.6 半监督的协同训练算法

本模块模拟了在样本数量较少的情况下如何充分利用无标记数据来参与分类器的训练，从而扩充样本数量，提高系统识别率。

在生成标签数据之前，本系统对视频类别的判断都是通过视频所处的文件夹名字来识别的，因此我们从每个类别中各选取30个视频放到unlabeled文件夹下，模拟无标记数据，然后剩下的70个视频中30个用来训练SVM分类器，30个用来训练KNN分类器，最后的10用来预测，具体算法如下：

第一步：先分别训练出两个不同的基分类器SVM和KNN；

第二步：获得所有未标记视频的路径并保存在两个List数据结构中；

第三步：用SVM分类器对第一个List中的每个视频进行预测，把低于置信度阀值的所有视频从List中删除，并选取置信度最高的5个加入到另一个分类器中进行训练（更新另一个分类器的训练集和标签集并调用train函数），并把这5个数据从List中移除，同时KNN分类器也作类似的处理。

第四步：重新第三步直到两个List结构均为空，协同训练完成。

在这里我们计算置信度的方法公式为：

conf=(0.5\*( FeatureNum /N)+(0.5\*(FeatureLeveal/10)));

其中conf表示一个视频通过某个分类器检测后的置信度，FeatureNum为检测出来的最多的类别个数，N为所有结果的总数，当FeatureNum小于等于10时，FeatureLeveal等于FeatureNum；当FeatureNum大10时，FeatureLeveal的值固定为10。

### 4.5.7 用户界面的实现

本次界面设计采用了Java的JFrame实现，为每个功能绑定了按钮，注册监听器。系统提供了提取全部视频特征、SVM训练、SVM预测、KNN预测、协同训练、终止所有线程和计算识别率等按钮以供用户选择相应的操作，另外还提供了显示视频的区域VideoCapture和显示操作提示的区域Console，视频下方还提供了当前操作的实时详细信息，比如说当前操作、当前视频的路径、视频信息、当前操作的帧数和视频总帧数等。

视频的显示原理其实就是依次提取视频的每一帧，然后分别把每一帧以图片的形式显示到视频区域，中间设置一个合适的停顿时间然后刷新下一帧。视频特征提取的时候一边提取一边在相应的区域显示出来，每一帧中我们以红色的点标识过滤出来的特征点，用绿色的点表示计算的特征中心，用蓝色的矩形表示以此中心选取一个合适的范围进行特征提取。提取的过程中可以点击加速来减小停顿时间，加速播放，也可以减速，还可以点击暂停，观察特征点有无异常。

视频显示右边的Console窗口是系统向用户传达信息的通道，显示了所有与当前操作有关的状态、提示、结果、异常等信息，用户可以通过这些信息准确地定位到出现问题的地方，实时监控系统的运行状态、运行进度等。

在本系统中大多数操作都是及其耗时的，如果不采用线程的编程方式的话一旦点了某个操作之后就是漫长的等待，连关闭都做不到，更别说状态的实时更新了。因此系统中普遍采用多线程系统方式，每一个耗时的操作都分配一个线程进行单独控制，整个用户界面是一个主线程，我们可以在提取特征的同时进行视频的显示、各状态信息的实时更新提示等，各个不相干的线程之间互不干扰。此外，有很多操作之间是有先后顺序的，比如说提取特征和训练分类器、加载分类器和视频动作预测等，对于这些情况，系统中还设置了一些共享变量来进行各相关线程的同步和互斥控制，当两个有先后顺序的线程意外地被用户同时启动以后，系统通过对共享变量的判断，如果前一个操作还未完成，则后一个线程则需要持续等待，直到优先级高的线程完成再退出等待，开始执行。举个例子，分类器模型的加载可能会花费掉数分钟的时间，为了节约时间，当用户界面第一次加载的时候，系统会产生一个线程在后台自动加载之前训练好的分类器模型，因为用户从打开系统到找到并成功选择要识别的视频动作这段时间系统完全是空闲的，我们可以把这段时间充分利用起来，不用等视频选择好了之后才来加载分类器，而是先后台加载分类器，当用户选择好了视频文件以后根据共享变量循环判断分类器有没有加载完成，如果加载好就可以直接进行识别了。

## 4.6 系统测试

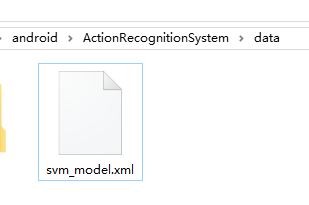
系统编码完成以后点击运行，加载出的用户界面如下：



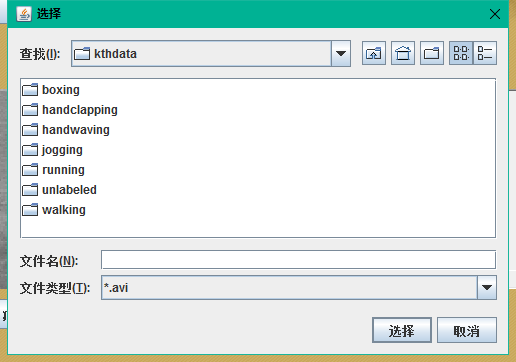
点击提取特征按钮，系统会对kthdata下的6个分类的600个视频以及unlabeled下的若干个未知分类的视频进行批量特征提取，各个视频的特征保存到hog目录下一个同名的.txt文件下，文件中每行表示一个特征向量。



单击训练按钮，系统先对预先设置好的样本数量个特征进行合并，生成标签文件，然后调用相应的分类器训练算法进行训练，训练完成后分类器模型成功保存到本地：



单击预测按钮，选择需要识别的文件，确定之后系统会对该视频进行特征提取，识别结束后根据结果按可能性从高到低排序，其中概率最高的类别即为最终的识别结果。如图所示，我们随机选取了一个boxing类别的视频进行识别，右侧的Console中显示了视频中有高达82.7%的特征向量被判定为boxing类，因此预测结果为boxing类，与视频真实的类别一致，识别成功。





## 4.7 小结

经过大半月的忙活，系统终于是成功地完成了，还是印证了那句古话，万事开头难，只要确定了一个方向，并为之不懈努力，接下来的事也就不那么困难了。在选择这个课题之前，从来都没有接触过这方面的知识，只是偶尔的在一些新闻里面看到过。真正开始做的时候真的是一头雾水，看了好多文献，尽管也提到过好多的算法和原理，无奈机器学习是一个很庞大的领域，分支发展错综复杂，直接以一个初学者的身份去看难免会出现知识断层，很多东西难以理解，直接去网上漫无目的地搜索很难知道自己的定位，甚至会误入到另一个相似的领域。

做系统也是，一定要有足够的耐心，说实话一开始的时候我真没有信心，不知道能不能坚持到系统成功运行，完全不知道设计的流程是什么，先做什么功能再做什么功能。后来终于静下心来，还特地去学了git版本管理系统的使用，准备放手大干一笔。于是配置好环境后我先做了个能读取视频和显示视频的界面，虽然中间遇到了些棘手的问题，好在最后终于成功解决了，这也给自己带来了很大的信心，有了更多的动力来迎接后面的各种挑战。

也是一个给自己的小经验吧，对于未知的任务，其实一开始不要太在意细节，先把大体的流程过一遍，对系统的总体有一个全局的掌握之后再来对系统的每个模块一一优化完善，这样效率其实是最高的。这次的系统也是，一开始只是按照稿纸上的设计做了个模型，编写了一个比较粗糙的系统，按照一种近乎理想的条件去设计，各种异常情况通通不考虑，只是保证这个粗糙的系统在正常情况下运行正确，这样系统的骨干也就完成了。下一步就是在稿纸上写出所有你能想到的异常情况，把这些情况一一完善，第三步就是不断测试，不断发现新的异常，不断修复。像用户界面、线程这些都是后来一点一点加上去的，因为如果一开始就把各种要实现的目标全部揣在脑子里，塞的死死的，这样头脑会特别乱，没法冷静的思考。

# 第六章 总结和展望

# 致谢

# 参考文献

1. 刘博. 基于关键姿势的人体动作识别[D]. 太原理工大学, 2014.
2. 吴文军. 基于光节点特征融合的人体行为识别[D]. 西安电子科技大学, 2015.
3. Schuldt C, Laptev I, Caputo B. Recognizing Human Actions: A Local SVM Approach[C]// International Conference on Pattern Recognition. IEEE, 2004:32-36 Vol.3.
4. 陈渊博. 视频序列中的人体动作识别[D]. 北京邮电大学, 2015.