

人工智能课程

期末报告

报告人：姚林丽

学号：2016202236

所在小组组员：

陈洁婷、薛 畅

苏吉雅、张 倩

目录

AI 智能小车项目报告

1. 项目内容介绍.....	3
2. 前期准备.....	3
2.1 硬件部分.....	3
2.2 软件部分.....	4
3. 第一阶段：实现小车基础功能.....	5
3.1 阶段目标.....	5
3.2 小车拼装.....	5
3.3 功能设计与程序设计.....	7
3.4 实验结果.....	10
4. 第二阶段：完善基础功能，为语音和图像识别做准备.....	11
4.1 阶段目标.....	11
4.2 实现超声波与红外线避障的结合.....	11
4.3 实现手机蓝牙控制小车运动.....	13
4.4 小车语音识别功能.....	14
4.5 小车图像识别功能准备.....	17
4.6 存在的问题.....	17
4.7 后期优化目标.....	18
4.8 小组成员分工.....	18
5. 第三阶段：基于深度学习实现语音识别和图像识别.....	18
5.1 阶段目标.....	18
5.2 实现电脑蓝牙向小车蓝牙模块传送信息.....	19
5.3 小车语音识别功能.....	19
5.4 小车图像识别功能.....	19
6. 个人工作和总结.....	20
课程学习笔记.....	21

AI 智能小车项目报告

1. 项目内容介绍

本项目基于 Arduino 这款便捷灵活的电子原型平台，使用 Arduino IDE 编写程序代码，利用 Arduino 电子板作为控制小车的核心。整个项目工程包括：采买基本硬件完成小车的搭建，实现小车前进、后退、左右转等基本行驶功能，实现小车红外线和超声波结合的避障方式，完善蓝牙控制小车运动模块，以及基于深度学习的小车语音识别和图像识别进阶功能。

语音识别和图像识别是应用深度学习非常广泛的领域，可以用已有的模型和知识在 AI 小车上实现新奇而有趣的功能。在语音识别方面，本组致力于应用相关知识实现智能小车的语音口令控制行驶和“识别主人”功能；在图像识别方面，本组优化了利用手机软件传图的模糊性、延时性，尝试了识别物体、倒车等一系列有趣的功能。本组同学通过自己动手实践，初步探索了人工智能应用领域知识、自动驾驶领域的知识，培养了利用深度学习等算法来解决实际问题的能力。

2. 前期准备

2.1 硬件部分

本项目使用到的硬件有：

- **核心控制板：**Arduino UNO R3 开发板。
- **组装小车的各个零部件：**包括 Motor control shield L293D 电机驱动扩展板，小车底板，四个轮子，电源，电机，USB 传输线，杜邦线，电焊铁，六角铜柱，充电器，螺丝螺母等等。
- **用来实现小车功能的各个硬件模块：**包括 BT06 蓝牙串口模块，超声波传感器 HC-SR04，超声波舵机云台，红外线传感器。

我们利用基本零件组装小车，用锂电池给小车稳定供电；将 Arduino UNO R3 开发板通过 USB 方头线连接电脑上传程序。利用超声波模块（包括舵机云台）、红外线探测模块

来实现避障功能,利用蓝牙模块来接收手机发出的控制信号、电脑发出的程序处理结果。

采买硬件清单如图 2.1、2.2 所示



图 2.1



图 2.2

2.2 软件部分

本项目所用的软件有如下几部分:

- **Arduino IDE:** Arduino 构建于开放原始码 simple I/O 介面版, 并且具有使用类似 Java、C 语言的 Processing/Wiring 开发环境。Arduino IDE 是计算机中的程序开发环境。你只要在 IDE 中编写程序代码, 将程序上传到 Arduino 电路板后, 程序便会告诉 Arduino 电路板要做些什么了。Arduino 能通过各种各样的传感器来感知环境, 通过控制灯光、马达和其他的装置来反馈、影响环境。板子上的微控制器可以通过 Arduino 的编程语言来编写程序, 编译成二进制文件, 烧录进微控制器。对 Arduino 的编程是通过 Arduino 编程语言 (基于 Wiring) 和 Arduino 开发环境 (基于 Processing) 来实现的。基于 Arduino 的项目, 可以只包含 Arduino, 也可以包含 Arduino 和其他一些在 PC 上运行的软件, 他们之间进行通信 (比如 Flash, Processing, MaxMSP) 来实现。
- **手机蓝牙串口 APP:** 通过手机蓝牙连接小车蓝牙模块 HC-05, 连接之后可以在软件中设置功能按键, 通过对应的功能按键发送二进制或者 16 进制的指令给小车, 从而实现蓝牙控制小车前进、后退、转弯等等基本功能, 还能通过蓝牙切换小车红外线、超声波、两者结合等避障模式。
- **USB 转串口驱动:** Arduino UNO R3 开发板直接通过接线连接电脑 USB 端口是不能被识别的, 需要安装 USB 转串口驱动后才能被 Arduino IDE 识别为正确的串口, 从而成功完成代码的烧制。

- **手机电脑实时传输图片软件 DroidCam:** 实现手机拍摄的图片或者视频实时传给电脑端，需要在手机安装 Android 软件 DroidCam，并在对应计算机上安装 Pc 端软件 DroidCam。
- **Anaconda:** 是一个开源的 Python 发行版本，其包含了 conda、Python 等 180 多个科学包及其依赖项。因为包含了大量的科学包，Anaconda 的下载文件比较大（约 531 MB），如果只需要某些包，或者需要节省带宽或存储空间，也可以使用 Miniconda 这个较小的发行版（仅包含 conda 和 Python）。
- **Tensorflow:** TensorFlow 是一个基于数据流编程（dataflow programming）的符号数学系统，被广泛应用于各类机器学习（machine learning）算法的编程实现，其前身是谷歌的神经网络算法库 DistBelief。

Tensorflow 拥有多层级结构，可部署于各类服务器、PC 终端和网页并支持 GPU 和 TPU 高性能数值计算，被广泛应用于谷歌内部的产品开发和各领域的科学研究。

TensorFlow 由谷歌人工智能团队谷歌大脑（Google Brain）开发和维护，拥有包括 TensorFlow Hub、TensorFlow Lite、TensorFlow Research Cloud 在内的多个项目以及各类应用程序接口。自 2015 年 11 月 9 日起，TensorFlow 依据阿帕奇授权协议开放源代码。

3. 第一阶段：实现小车基础功能

3.1 阶段目标

在第一阶段，本项目致力于完成如下目标：

- 完成小车的搭建
- 学习在 Arduino IDE 中编写代码，并烧制到 Arduino 主板中
- 实现小车的基本功能：跑动、变速、变向、避障

3.2 小车拼装

小车拼装步骤如下：

- 1) 组装小车四个轮子和底盘，每个轮子与双驱直流减速电机相连，保持连线方向的一

致，使得小车四个轮子朝同一个方向转动。

- 2) 将小车的四个电机连接到对应电机驱动板上的端口。
- 3) 电机驱动板在上，Arduino R3 板在下，将电机驱动板直接与 Arduino 板相连。
- 4) 连接电源到电机驱动板上，开启驱动板的电源开关，驱动板和 Arduino 板都会通电。
- 5) 安装超声波探测器和相应舵机平台，舵机使得超声波检测仪可以左右旋转探测，扩大了探测的范围。将超声波模块通过相应连线接到电机扩展板相应接口。
- 6) 在小车前侧左右安装两个红外线探测器，同样将红外探测器通过杜邦线连到电机扩展板相应接口。
- 7) 将蓝牙 HC-05 模块直接插到电机扩展板的相应接口。

各个模块在电机驱动扩展板上相对应的接口如图 3.1 所示。Arduino R3 板具体的端口介绍如图 3.2 所示。

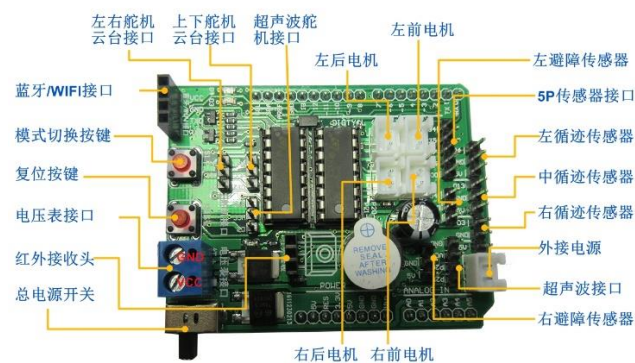


图 3.1 电机驱动扩展板的具体接口示意图

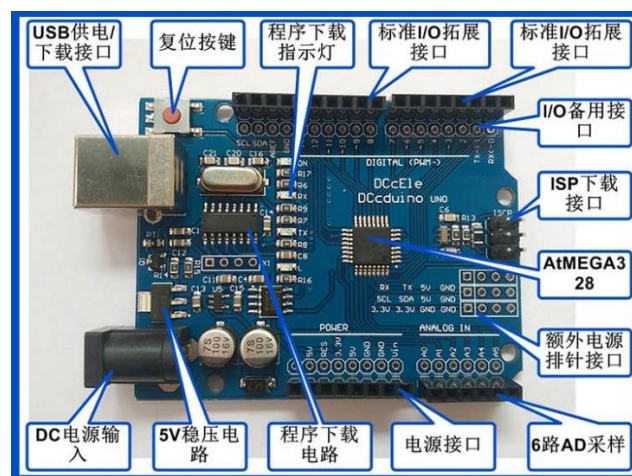


图 3.2 Arduino R3 板具体端口名称介绍示意图

理论上，小车拼装完成后各个角度的视图如下图 3.3 所示：

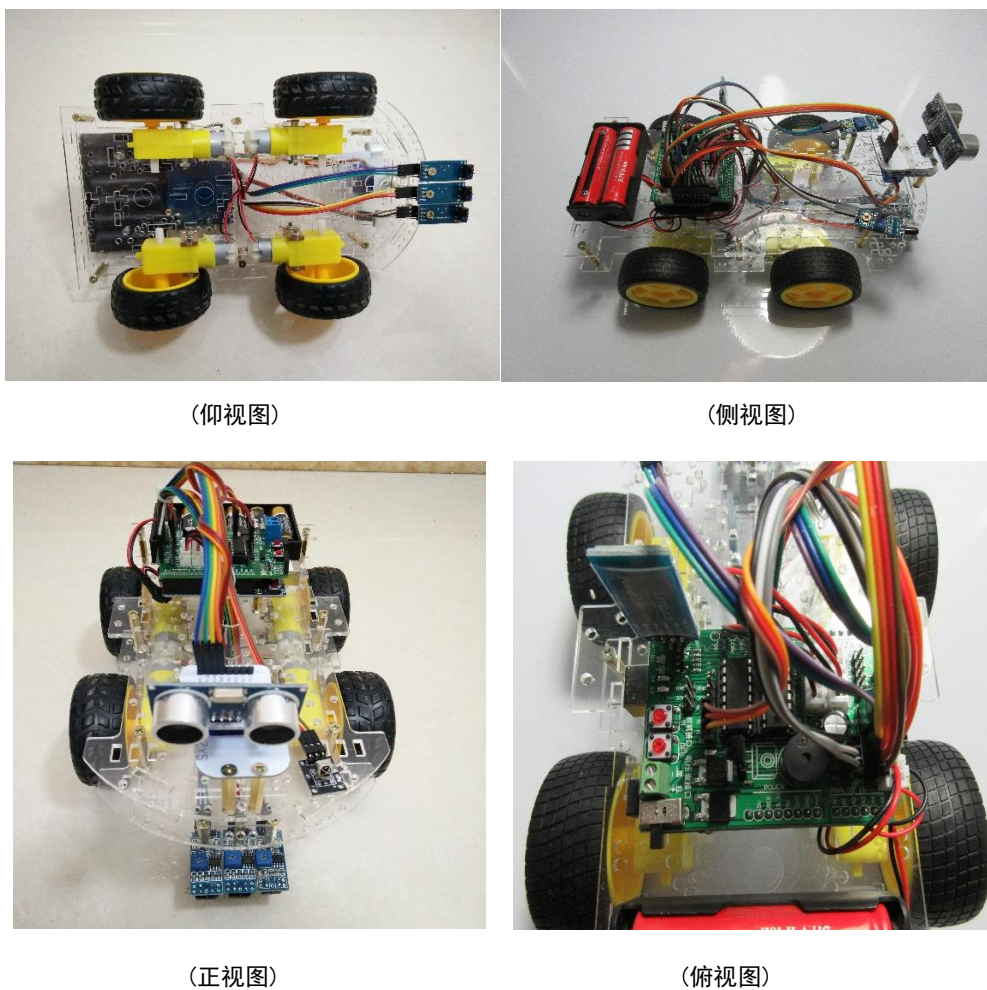


图 3.3 小车拼装完成后各个角度的视图

3.3 功能设计与程序设计

初始化引脚程序：

```
int Left_motor=8;    //左电机(IN3) 输出0 前进  输出1 后退
int Left_motor_pwm=9;    //左电机PWM调速

int Right_motor_pwm=10;    // 右电机PWM调速
int Right_motor=11;    // 右电机后退(IN1) 输出0 前进  输出1 后退

void setup()
{
    //初始化电机驱动IO为输出方式
    pinMode(Left_motor,OUTPUT); // PIN 8 8脚无PWM功能
    pinMode(Left_motor_pwm,OUTPUT); // PIN 9 (PWM)
    pinMode(Right_motor_pwm,OUTPUT); // PIN 10 (PWM)
    pinMode(Right_motor,OUTPUT); // PIN 11 (PWM)
}
```

1) 前进与 PWM 调速


```

void run(int time)    // 前进
{
    digitalWrite(Right_motor, LOW); // 右电机前进
    digitalWrite(Right_motor_pwm, HIGH); // 右电机前进
    analogWrite(Right_motor_pwm, 150); // PWM比例0~255调速，左右轮差异略增减

    digitalWrite(Left_motor, LOW); // 左电机前进
    digitalWrite(Left_motor_pwm, HIGH); // 左电机PWM
    analogWrite(Left_motor_pwm, 150); // PWM比例0~255调速，左右轮差异略增减
    delay(time * 100); // 执行时间，可以调整
}

```

2) 后退

```

void back(int time)    // 后退
{
    digitalWrite(Right_motor, HIGH); // 右电机后退
    digitalWrite(Right_motor_pwm, HIGH); // 右电机前进
    analogWrite(Right_motor_pwm, 150); // PWM比例0~255调速，左右轮差异略增减

    digitalWrite(Left_motor, HIGH); // 左电机后退
    digitalWrite(Left_motor_pwm, HIGH); // 左电机PWM
    analogWrite(Left_motor_pwm, 150); // PWM比例0~255调速，左右轮差异略增减
    delay(time * 100); // 执行时间，可以调整
}

```

3) 左转

```

void left(int time)    // 左转(左轮不动，右轮前进)
{
    digitalWrite(Right_motor, LOW); // 右电机前进
    digitalWrite(Right_motor_pwm, HIGH); // 右电机前进
    analogWrite(Right_motor_pwm, 150); // PWM比例0~255调速，左右轮差异略增减

    digitalWrite(Left_motor, LOW); // 左电机前进
    digitalWrite(Left_motor_pwm, LOW); // 左电机PWM
    analogWrite(Left_motor_pwm, 0); // PWM比例0~255调速，左右轮差异略增减
    delay(time * 100); // 执行时间，可以调整
}

```

4) 急速左转

```

void spin_left(int time)    // 左转(左轮后退，右轮前进)
{
    digitalWrite(Right_motor, LOW); // 右电机前进
    digitalWrite(Right_motor_pwm, HIGH); // 右电机前进
    analogWrite(Right_motor_pwm, 150); // PWM比例0~255调速，左右轮差异略增减

    digitalWrite(Left_motor, HIGH); // 左电机后退
    digitalWrite(Left_motor_pwm, HIGH); // 左电机PWM
    analogWrite(Left_motor_pwm, 150); // PWM比例0~255调速，左右轮差异略增减
    delay(time * 100); // 执行时间，可以调整
}

```

5) 右转


```

void right(int time)          //右转(右轮不动,左轮前进)
{
    digitalWrite(Right_motor, LOW); // 右电机不转
    digitalWrite(Right_motor_pwm, LOW); // 右电机PWM输出0
    analogWrite(Right_motor_pwm, 0); // PWM比例0~255调速,左右轮差异略增减

    digitalWrite(Left_motor, LOW); // 左电机前进
    digitalWrite(Left_motor_pwm, HIGH); // 左电机PWM
    analogWrite(Left_motor_pwm, 150); // PWM比例0~255调速,左右轮差异略增减
    delay(time * 100); // 执行时间,可以调整
}

```

6) 急速右转

```

void spin_right(int time)      //右转(右轮后退,左轮前进)
{
    digitalWrite(Right_motor, HIGH); // 右电机后退
    digitalWrite(Right_motor_pwm, HIGH); // 右电机PWM输出1
    analogWrite(Right_motor_pwm, 150); // PWM比例0~255调速,左右轮差异略增减

    digitalWrite(Left_motor, LOW); // 左电机前进
    digitalWrite(Left_motor_pwm, HIGH); // 左电机PWM
    analogWrite(Left_motor_pwm, 150); // PWM比例0~255调速,左右轮差异略增减
    delay(time * 100); // 执行时间,可以调整
}

```

7) 倒车

```

void back(int time)           //后退
{
    digitalWrite(Right_motor, HIGH); // 右电机后退
    digitalWrite(Right_motor_pwm, HIGH); // 右电机前进
    analogWrite(Right_motor_pwm, 150); // PWM比例0~255调速,左右轮差异略增减

    digitalWrite(Left_motor, HIGH); // 左电机后退
    digitalWrite(Left_motor_pwm, HIGH); // 左电机PWM
    analogWrite(Left_motor_pwm, 150); // PWM比例0~255调速,左右轮差异略增减
    delay(time * 100); // 执行时间,可以调整
}

```

8) 停车

```

void brake(int time)          //刹车,停车
{
    digitalWrite(Right_motor_pwm, LOW); // 右电机PWM 调速输出0
    analogWrite(Right_motor_pwm, 0); // PWM比例0~255调速,左右轮差异略增减

    digitalWrite(Left_motor_pwm, LOW); // 左电机PWM 调速输出0
    analogWrite(Left_motor_pwm, 0); // PWM比例0~255调速,左右轮差异略增减
    delay(time * 100); // 执行时间,可以调整
}

```

9) 红外避障

```
void loop()
{
    val = analogRead(GP2D12);
    //读取红外测距传感器数据
    distance_float=2547.8/((float)val*0.49-10.41)-0.42;
    //转化为浮点型数值
    start();
    //开始执行主要函数
}

void start()
{
    //如果距离小于20，则避让
    if (distance_float<=20 && distance_float>0)
    {
        avoid();
    }
    else //否则继续前进
    {
        advance();
    }
}
```

其中，小车四个轮子的电机驱动真值表如表 3.3 所示。

	Left_motor = 8	Left_motor_pwm=9		Right_motor	Right_motor_pwm=10
前进	0	1		0	1
刹车	0	0		0	0
左转A	0	0		0	1
左转B	1	1		0	1
右转A	0	1		0	0
右转B	0	1		1	1
后退	1	1		1	1

表 3.3 电机驱动真值表

3.4 实验结果

编译通过了程序代码的正确性。理论上，小车可以完成前进、后退、倒车、停车、左转、右转、左急转、右急转等基础功能，并且在行驶过程中当左右轮速度因为硬件问题不匹配时可以自动调速达到一致。理论上初步实现了红外线避障，可以在很短的距离内判断小车左前方和右前方是否有障碍物。

3.5 存在的问题

因为购买的小车有质量上的问题，导致硬件出现不可知并且不可预测的错误。小车在实际运动中存在如下问题：

- 1) Uno 板和驱动板引脚对应标识模糊、杜邦线接触不良、电压不够。
- 2) 硬件的物理原因导致小车轮子之间力度与摩擦力的不稳定差异。
- 3) 在安装 USB 转换串口驱动后仍无法上传代码到 Arduino 板中。

对应解决方法：

- 1) 将普通电池换成续航更稳定的航模电池
- 2) 更换轮子硬件
- 3) 在上传代码给 Arduino 板时，打开小车电源，给小车通电。

4. 第二阶段：完善基础功能，为语音和图像识别做准备

4.1 阶段目标

解决第一阶段中的硬件问题。在第一阶段完成小车基本功能的基础上，继续完善小车的其他功能：将小车的普通红外线避障改进为红外线与超声波相结合的避障方式，扩大了小车避障的范围和提高了小车避障的精度。同时，新增了用手机蓝牙串口 APP 控制小车行驶的功能。

在改进小车的基础功能之外，开始尝试利用神经网络和深度学习的知识来探索小车的进阶功能，来增加小车的“智能”程度。考虑到真实场景下，有人驾驶或者无人驾驶都要涉及视觉和听觉方面的判断，本项目将目光放在图像识别和声音识别领域上，希望通过已有知识来实现一些新奇有趣的功能。具体地，在语音识别领域，本项目主要想实现两方面的功能：1. 语音口令控制小车行驶；2. 实现小车能识别“主人”的语音，只自动执行主人发出的语音口令，而忽略其他人发出的语音口令。

4.2 实现超声波与红外线避障的结合

4.2.1 设计思路

在第一阶段实现小车避障功能时，本组同学发现红外避障探测仪只能在很短的距离内感知障碍物，并且只能给出是否有障碍物的判断信号，不能给出障碍物的具体距离。而超声波

探测仪器能在更大范围内判断出障碍物的存在，并且能精确给出障碍物的距离，但是超声波避障也存在一个问题：探测障碍物的横向范围有限。并且超声波模块作为小车的“眼睛”部分，距离地面有 20 厘米的高度差，从而不能检测出贴近地面的障碍物。综合红外线和超声波测距的优势和劣势，本项目决定将两者结合作为避障的优化：即扩大了避障的横向和纵向范围，又提高了避障的精度。

图 4.1 分别为本项目小车的超声波模块和红外避障模块。

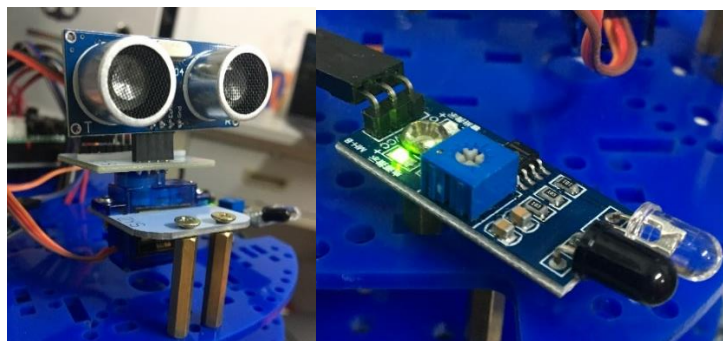


图 4.1

4.2.2 功能设计与程序设计

- 超声波测距代码，返回障碍物距离 Distance。

```
//超声波部分测距
Distance_test();//测量前方距离
Distance_display();//液晶屏显示距离
```

- 红外线测距代码，返回左前方和右前方是否有障碍物：是，返回 HIGH；否，返回 LOW。

```
//红外线部分测距
//有信号为LOW 没有信号为HIGH
SR_2 = digitalRead(SensorRight_2);
SL_2 = digitalRead(SensorLeft_2);
```

- 超声波测距距离长，所以当超声波探测前方障碍物距离<40cm 时，小车停车开始避障。小车首先判断此时前方左右两个红外线探测仪的探测结果：如果左边探测到近处有障碍物，则小车右转一个小角度；右边有障碍物同理；如果两边都有障碍物，则小车只能先倒车。进行一次操作后，小车重新进行超声波和红外线的障碍物测距，如果仍有障碍物则重复上述过程，直到小车完全绕开障碍物，恢复正常行驶。在结合超声波和红外线避障后，通过实验可以发现小车能绕开比较复杂的充满障碍物的地形。

```

while(Distance < 40)//再次判断是否有障碍物，若有则转动方向后，继续判断
{
    brake(1); //先刹车
    back(2);
    //下面根据红外线来判断调整方向
    if (SL_2 == HIGH & SR_2 == HIGH) //红外线没有检测到障碍物
    {    right(1.5); }
    else if (SL_2 == HIGH & SR_2 == LOW) // 右边探测到有障碍物，有信号返回，向左转
    {    left(1.5); }
    else if (SR_2 == HIGH & SL_2 == LOW) // 左边探测到有障碍物，有信号返回，向右转
    {    right(1.5); }
    else { // 都是有障碍物，后退
        back(5); //后退500MS
        right(1.5); //调用右转函数 延时500ms
    }
    Distance_test(); //测量前方距离
    Distance_display(); //液晶屏显示距离
}

```

4.3 实现手机蓝牙控制小车运动

4.3.1 设计思路

利用小车的蓝牙串口模块 HC-05，并在手机端安装蓝牙串口助手 app。利用手机蓝牙搜寻到小车的蓝牙模块，配对成功后，可以通过手机软件向小车发送二进制的字符串信号，小车根据收到的字符串信号，完成对应的功能。所以在手机端 app，可以自己定义不同的按键功能，从而实现将已有的功能都集成到手机蓝牙控制端。

4.3.2 程序设计

Serial 库模块封装了对串口的访问，支持二进制传输。借助 Serial 库，初始化蓝牙串口。小车的蓝牙模块用 Serial.read() 来读取手机端给它发送的二进制字符型指令，读取指令 act 后，执行函数 go(act)，来执行对应的功能。

```

void loop()
{
    Serial.begin(9600); //初始化蓝牙串口
    if(start==1){
        if(Serial.available()){
            act = Serial.read();
            Serial.println(act);

            start=0;
            go(act);
            digitalWrite(beep, HIGH);
        }else{
            go(0);
        }
    }else{
        go(act);
    }
}

```

执行指令函数 go(act)内部具体设计如下。本项目共实现了手机蓝牙控制的 7 个功能：前进、后退、左急转、右急转、左转、右转、倒车。

```
void go(int cmd) {  
    switch(cmd) {  
        case 1: run();break;  
        case 2: back();break;  
        case 3: spin_left();break;  
        case 4: spin_right(); break;  
        case 5: left(); break;  
        case 6: right(); break;  
        case 7: brake(); break;  
        default: brake();  
    }  
}
```

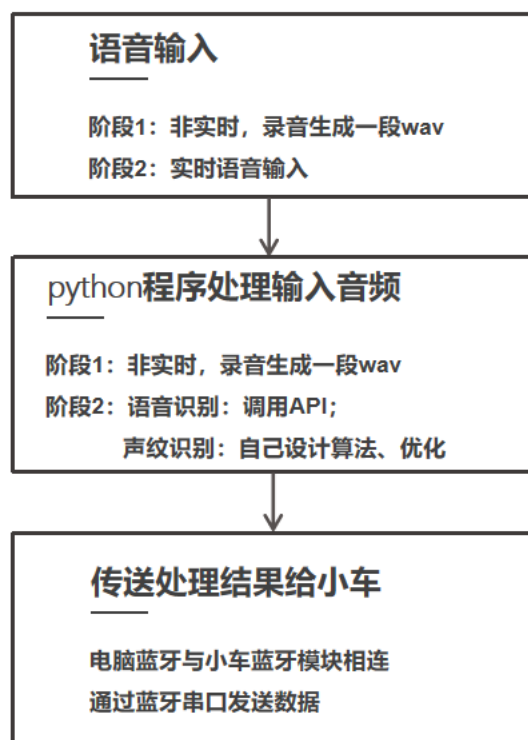
4.4 小车语音识别功能

4.4.1 设计思路

在语音识别部分，本项目预期实现以下两个功能：

- **功能 1：**语音口令控制小车行驶。语音口令包括“前进”“后退”“左转”“右转”“停车”等。实现的技术主要是语音识别
- **功能 2：**小车能识别“主人”语音。实现技术主要是 SVM 支持向量机分类，声纹识别

4.4.2 功能实现流程



4.4.3 “语音口令控制小车”功能已完成的工作

1) 录音

调用 python 语言 pyaudio 库，即可在计算机上进行录音，用户可以自由控制录音时长。输入一段语音口令，输出一段 wav 音频。

```

pa=PyAudio() ;
stream=pa.open(format = paInt16,channels=1, rate=framerate, input=True,
frames_per_buffer=NUM_SAMPLES);
stream.read(NUM_SAMPLES);
  
```

2) 语音口令识别

输入一段 wav 音频，调用百度 api 的语音识别库进行语音识别，输出识别结果，并且根据识别结果转换为指令。

```

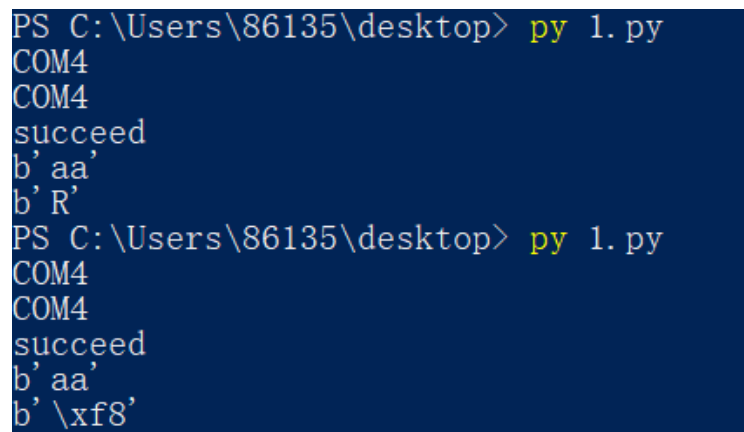
client = AipSpeech(APP_ID, API_KEY, SECRET_KEY);
res=client.asr(get_file_content(filepath), 'wav', 16000, {'dev_pid':
1536,})
re=res['result']
  
```

3) 电脑蓝牙传送指令结果给小车蓝牙串口模块

电脑蓝牙配对小车蓝牙，计算机上出现虚拟串口。调用 python 语言 pyserial 模块进行指令的传输，其中 pyserial 模块封装了对串口的访问，支持二进制传输。图 4.2 是电脑端通过虚拟串口 COM4 向小车蓝牙发送字符串，并且成功接收到小车蓝

牙端返回信息的截图。

```
import serial
ser = serial.Serial('COM4', 9600, timeout=0.5)
//打开串口 COM4, 设置波特率
ser.write(data) //从串口读数据
st=ser.read()   //从串口写数据
```



```
PS C:\Users\86135\desktop> py 1.py
COM4
COM4
succeed
b'aa'
b'R'
PS C:\Users\86135\desktop> py 1.py
COM4
COM4
succeed
b'aa'
b'\\xf8'
```

图 4.2

4.4.4 “小车识别主人”功能已完成的工作

1) 数据集采集

采集多个人的声音录音，每个人录音多次，得到多个 wav 文件构成数据集。

2) 数据处理与训练

抽取 MFCC 特征，目前使用支持向量机 SVM 给音频分类。应用 SVM 的主要流程是：给 wav 音频加窗、分帧，使用 mfcc 得到特征向量，再利用 pca 主成分分析法降维，从 12 维度变成 8 维度，再把每帧的 mfcc 拼接得到一整段音频的 mfcc，对不齐的末尾补 0，最后用 SVM 支持向量机分类。图 4.3 是提取 MFCC 特征向量的流程图。

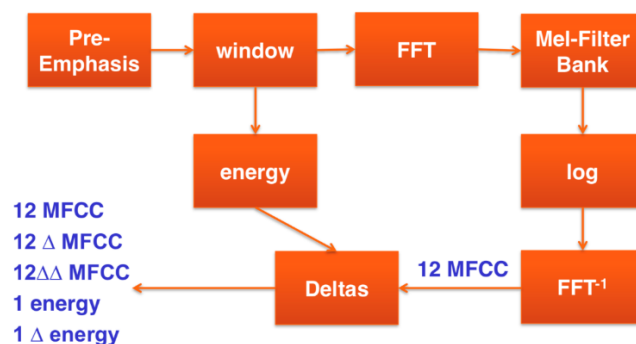


图 4.3

3) 得到识别结果

目前得到的效果不太好，分类识别准确度不高，分析原因有 svm 的参数需要优化、

数据集太小（只收集了 3 个人每人 10 个 wav 文件）、噪音太大、每个人的音频未对齐等等。

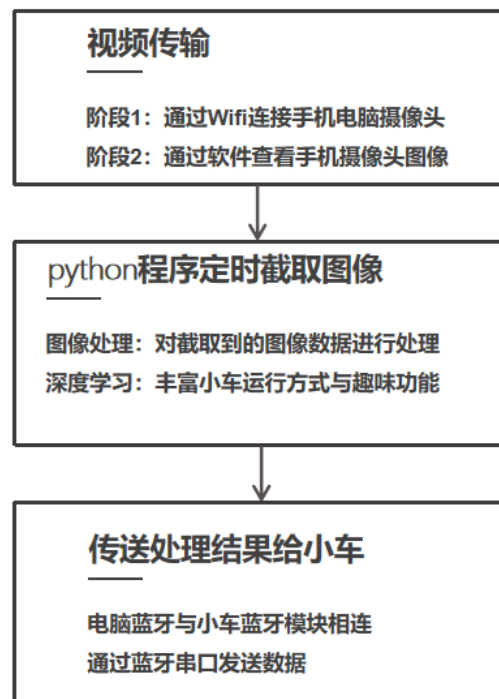
4.5 小车图像识别功能准备

4.5.1 设计思路

在第二阶段先实现将手机拍摄的图片或者视频实时传送给电脑，为第三阶段的图像识别做准备，可以分为如下两个功能：

- **功能 1：** 将手机拍摄的图片或者视频传入电脑端程序
- **功能 2：** 用 python 处理电脑端图片并保存

4.5.2 功能实现流程



4.6 存在的问题

1) 硬件问题：

- 舵机无法转向、小车右前轮发生故障
- 调用 pyserial 库要提供虚拟设备端口，但在 mac 电脑上找不到相应的端口号
- 电脑蓝牙在给小车蓝牙串口模块发送数据出现莫名的编码错误

2) 语音识别部分的问题：

- 现阶段还不能做到实时语音控制，只能通过录音对音频进行处理
- 小车识别主人的功能效果不太好，SVM 分类结果不太准
- 电脑蓝牙在给小车蓝牙串口模块发送数据出现莫名的编码错误

3) 图像识别部分的问题：

- 影像通过 Wifi 网络进行传输，存在延迟问题，且网速较慢的情况下延迟加重
- 在小车快速行驶的情况下，可能会产生拍摄模糊的情况

4.7 后期优化目标

- 提高程序运行效率，尽量降低延迟造成的影响
- 考虑减少截取图像之间的时间间隔，考虑自动筛除部分图像
- 逐一排除软件、硬件方面的 bug

4.8 小组成员分工

陈洁婷（组长）：实现手机蓝牙控制小车运动模块；实现了小车语音口令识别、主人识别部分

薛 畅：手机端图像识别 app 的调研与制作

张 倩：手机电脑端视频传输与图像处理

姚林丽：实现红外线与超声波结合避障模块、PPT 制作，参与电脑小车蓝牙传输

苏吉雅：电脑传送数据给小车蓝牙串口

5. 第三阶段：基于深度学习实现语音识别和图像识别

5.1 阶段目标

解决第二阶段遇到的问题，并且在第二阶段的基础上完善小车的语音识别功能和图像识别功能。在语音识别方面，沿着第二阶段的思路，我们完成了小车的识别主人功能和识别语音口令功能。在图像识别功能，我们小组致力于实现一个有趣的功能：让障碍物在小车面前不断移动，小车原地判断障碍物的速度和车身距离障碍物的距离，经过算法判断后寻找合适的时机向障碍物方向行驶并能巧妙躲过障碍物。

5.2 实现电脑蓝牙向小车蓝牙模块传送信息

在第二阶段，我们已经实现电脑蓝牙配对小车蓝牙，利用 python 语言 pyserial 模块将指令传输给小车。但在第二阶段，电脑端通过虚拟串口 COM4 向小车蓝牙发送字符串，小车可以接受到信息，但不能接受到正确的字符串指令，所以相应也不能执行正确的动作。在第三阶段，我们发现这是由于蓝牙传输过程中不正确的编码造成的，所以进行了改进，最终实现了电脑和小车之间正确的字符串传输。

电脑端传输字符串的代码如下：

```
data = "1" ;  
  
Serial.write(data.encode()); //电脑发送 data 给小车  
  
st=ser.read();           //电脑接收小车返回的信号
```

5.3 小车语音识别功能

在第二阶段，本组已经完成了小车语音识别大部分的电脑端工作：对于语音口令识别功能，调用百度 API 进行识别；对于小车识别主人功能，利用支持向量机方法进行分类。在第三阶段，我们解决了 python 录音百度 API 识别不准的问题，提升了支持向量机分类的准确率，并新增了一个功能。新增功能为让小车听到不同的声音作出相应的反应：听到雨声，减速行驶；听到动物叫声，停车；听到汽车鸣笛声，加快速度行驶；听到警报声，向右行驶让出车道。新增的功能是在模拟自动驾驶上路时可能会遇到的各种问题，让小车作出符合实际的反应，来实现其智能。

5.4 小车图像识别功能

根据第二阶段提出的设想和准备，实现小车可以做游戏的功能：即能判断面前不断来回移动的障碍物的速度和距离，智能地计算出要使自己能通过前方不被障碍物撞击的合适速度与时机，富有趣味性。

6. 个人工作和总结

在本组的项目中，我主要做辅助的工作：实现小车红外线与超声波结合避障模块、PPT制作，参与实现电脑小车蓝牙传输。在深度学习功能中，通过向同组成员学习，了解了支持向量机的实际应用，尝试了用循环神经网络 RNN 去解决实际语音分类的问题。参与这个项目最大的收获还是在小组同学的帮助下，学习到了很多知识的实际运用。以及在课堂上，从传统的人工智能方法到 CNN 等深度学习方法，再到最近很热的 GAN 等模型，系统地了解了人工智能领域的各种方法和方向，收益颇丰。

课程学习笔记

一. 传统 AI 方法

AI: 与思维有关, 与感知有关, 与行动有关

AI 模型

目的: 解释过去, 预测未来, 理解当前

核心是表示, 表示往往决定算法

生成测试法:

【例 1】农夫, 狐狸, 鹅, 谷物过河问题

四元组表示 (F, Fx, G, Gn)

取值: 0 表示左岸, 1 表示右岸

起始状态 (0,0,0,0) 目标状态 (1,1,1,1)

移动约束 (x,y,z,k) \rightarrow (x',y',z',k')

移动约束表示: (每次农夫只能带一个物体过河) $|y-y'|+|z-z'|+|k-k'| \leq 1$

矛盾状态约束: (0,0,1,1) - 鹅和谷物在右岸, 不可行 (0,1,1,1) (0,1,1,0) (1,0,0,0) (1,1,0,0) (1,0,0,1)

【例 2】传教士与食人族问题

【不定积分的求解】

已有信息: 积分表 (简单函数的积分) \rightarrow 目标: 求任意一个复杂的不定积分的值

启发式变换 \rightarrow 求解流程: 与或树

一种推理结构: 与或树

前向链\可解释性

AI 搜索-八数码问题

任意给一个

2	1	5
6	3	4
7		8

用最少的移动格数变成

1	2	3
8		4
7	6	5

解决方法:

1. 大英博物馆 (穷举)

每一层按照词典序进行排列, 搜索不能打转

2. 深搜

3. 广搜

问题: 深搜广搜是无信息的搜索, 在选择路径的时候无法分辨是在靠近目标还是在远离目标

改进: 爬山算法和术搜索算法

爬山算法 (对深搜的改进)

选择距离最近的下一节点

术搜索算法（对广搜的改进）

搜索束是指：向下一层进行扩展时，能保留的最大的路径数

A*算法：

A*算法就是把启发式方法（heuristic approaches）如 BFS，和常规方法如 Dijkstra 算法结合在一起的算法。有点不同的是，类似 BFS 的启发式方法经常给出一个近似解而不是保证最佳解。然而，尽管 A* 基于无法保证最佳解的启发式方法，A* 却能保证找到一条最短路径。

Zero-Sum Games(零和博弈)：所有博弈方的利益之和为零，即一方有所得，其他方必有所失。在零和博弈中，博弈各方是对抗的。

Minimax 算法：普遍用于博弈游戏中，一方要将自己的利益最大化，另一方要让对方的利益最小化

时间复杂度： $O(b^d)$

-b 每一个结点的平均分支数

-d 搜索的深度

-在搜索树中就有 b^d 个节点

【例】以国际象棋为例 $b=15$ $d=100$ 时间复杂度为 $O(15^{100}) \approx O(10^{120})$

Alpha-Beta 剪枝：

α ：答案最小值

β ：答案最大值

$ans \in [\alpha, \beta]$

if $\alpha \geq \beta$ 剪枝

【总结】：搜索深度 d 越大，程序越聪明

搜索深度 d 越大，剪枝越多

剪枝可以使程序在相同时间内，最多多搜索一倍的深度

二. 神经网络

机器学习的定义：

Machine Learning is the science (and art) of programming computers so they can learn from data

Arthur Samuel, 1959 提出机器学习不需要显示编程，这是它最大的特点

1. Training set-训练集：机器用来学习的数据

传统的人工智能方法与机器学习的比较：

传统的人工智能需要手工去构建模型，手工调整

机器学习可以自动化地去适应数据，解放了人类的工作量。并且可以进行数据的不断更新，模型的不断改进，使算法更高效，但关键环节还是需要人们去做。可以帮助人们去学习，通过观察模型的表现，更好地理解 and 认识问题领域。

需要注意的是，机器学习并不是取代了人，而是将人的能力和机器结合起来。

机器学习适用的情况：

1. 对于需要大量人工调整的工作，或者具有很多规则的复杂问题

2. 现在用传统方法很难解决的复杂问题，用机器学习可能会更好。例如，对图像中对象的识别，若用传统的方法去做，一幅图像中包含的像素是巨大的，很难手动去分析并建立规则，而机器学习就可以做到
3. 对于一些有波动的环境，数据总是会变化，这时用机器学习可以自动适应这些变化
4. 现代社会，大量的数据积累，基于海量的数据，可以去解决新的更多的问题

机器学习的类型划分：

根据是否需要人类指导，分为模型：supervised（监督学习）,unsupervised（无监督学习），Semisupervised（半监督）and Reinforcement Learning(强化学习)

Supervised learning 监督学习：

- 分类
- 回归问题

Unsupervised learning 无监督学习：

- 聚类问题 【例】k-means 算法
- 关联规则的学习
- 可视化与降维

Reinforcement Learning 强化学习

一种以环境反馈作为输入的、特殊的、适应环境的机器学习方法。根据环境的变化，机器自己去做一些判断和学习，通过奖励或者惩罚的方式来告诉机器做的好还是不好。

Semi-supervised Learning 半监督学习是监督学习与无监督学习相结合的一种学习方法。它主要考虑如何利用少量的标注样本和大量的未标注样本进行训练和分类的问题

Batch and Online Learning

批量学习（batch learning）一次性批量输入给学习算法，可以被形象的称为填鸭式学习。

线上学习（online learning）按照顺序，循序的学习，不断的去修正模型，进行优化。

Instance-Based Versus:

“死记硬背”的方式，基于数据检索的方法——记住所有的邮件，对新来的邮件进行度量，判断和已有的邮件的距离

数据对机器学习的挑战：

- 训练集数据量不足，不够具有代表性，只是实际情况的一个子集—>训练出带有“偏见”的模型。
- 有噪音：非相关的数据。

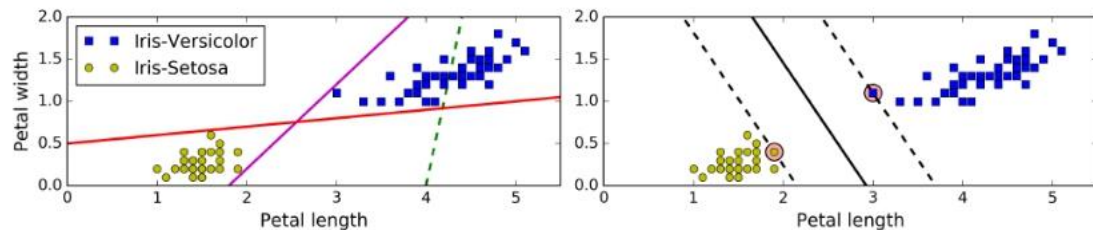
算法方面的挑战：

- Overfitting 过拟合，泛化性能不好导致的。解决方法：正则化 regularization
- Underfitting 欠拟合

三 . SVM (Support Vector Machine) –支持向量机

- 有严谨的数学证明
- 适合解决复杂分类问题，中小型数据集

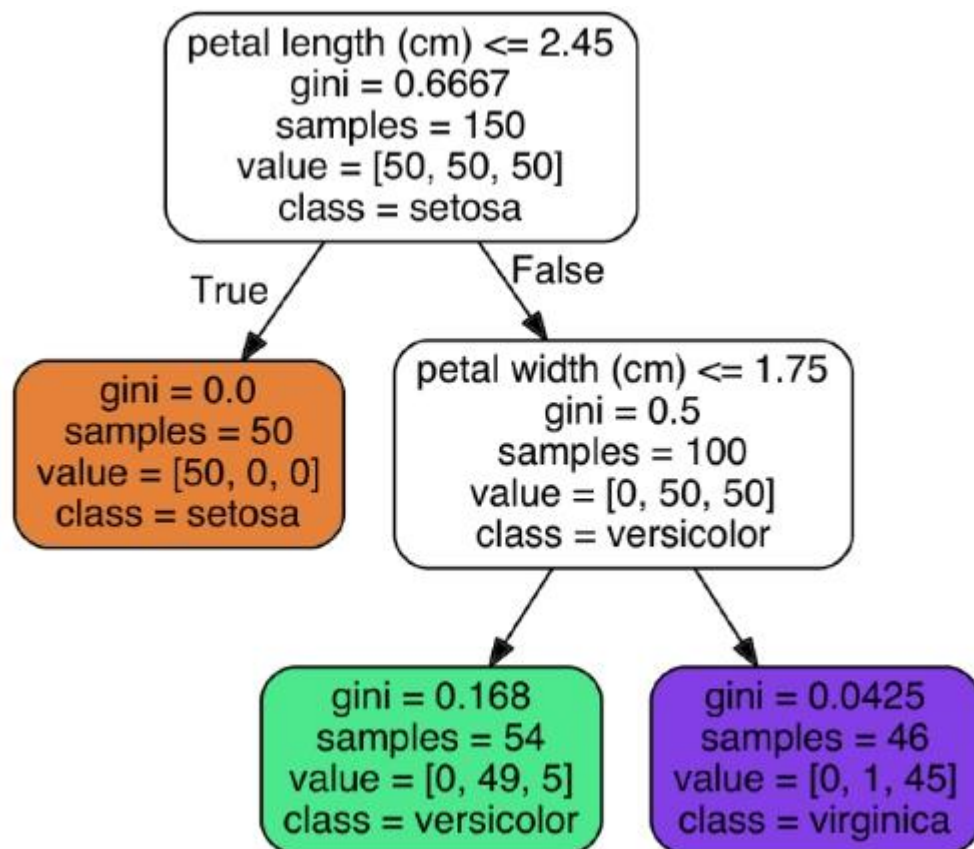
Linear SVM Classification 线性分类



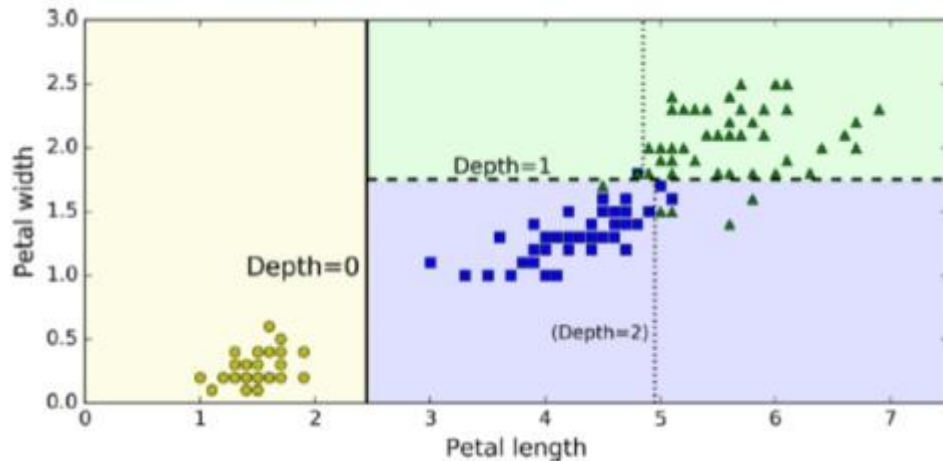
<https://blog.csdn.net/liugan528/article/details/79448379> hands-on5

四. 决策树–Decision Trees

- 无参模型，无超参数可以控制，但是若没有限制构建的树深度可以很长很长
- `Tree_clf = DecisionTree` (可以限定 depth)
- 结构：树根-整个数据集；然后选择一个最优特征，按着这一特征将训练数据集分割成子集，使得各个子集有一个在当前条件下最好的分类。



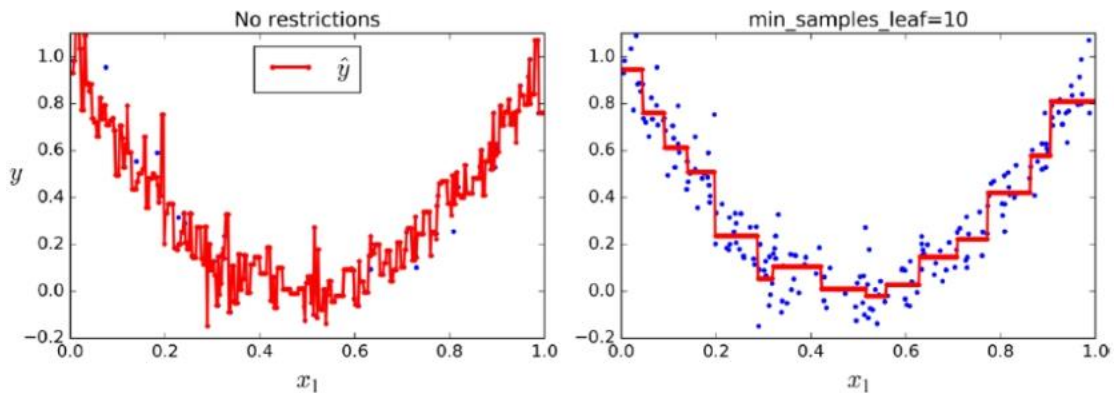
只能横竖划分，不能曲线



- Gini impurity(Gini 系数) 越小越好, 说明纯度更高

$$G_i = 1 - \sum_{k=1}^n p_{i,k}^2$$

- *Overfitting 问题*
不能一直划分下去, 树的深度很大, 使得模型完全契合训练集, 就没有实际意义了 à 故叶节点不能划分到一个
- 既可以做分类也可以做回归, 回归例子: 左图-过拟合-极度锯齿化; 右图: 加一些限制

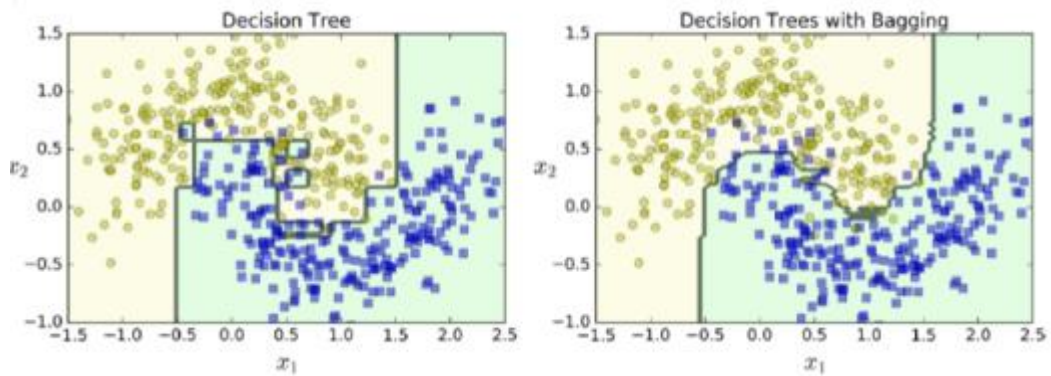


五. 随机森林-决策树的集合

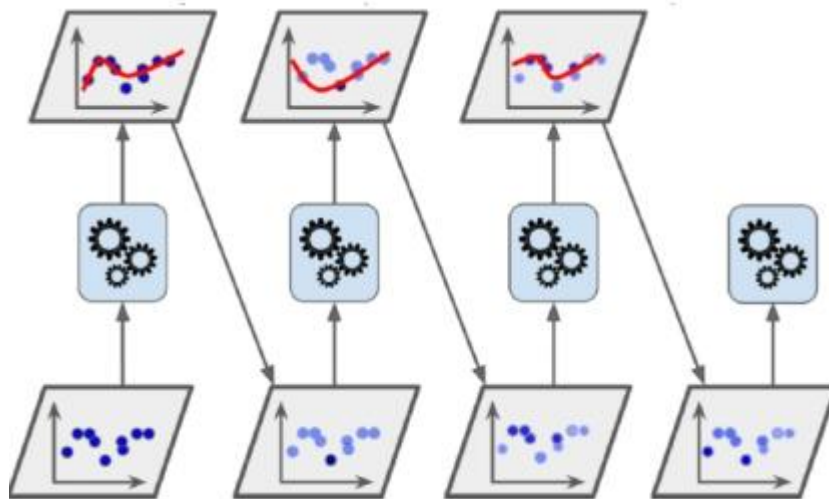
能够处理具有高维特征的输入样本, 而且不需要降维

随机森林就是通过集成学习的思想将多棵树集成的一种算法, 它的基本单元是决策树, 而它的本质属于机器学习的一大分支——集成学习 (Ensemble Learning) 方法。

- Bagging: 放回抽样, 不限制构建子集个数 --实际中一般用这种
(如果训练集大小为 N , 对于每棵树而言, [随机且有放回地](#)从训练集中的抽取 N 个训练样本; 每棵树的训练集都是不同的, 而且里面包含重复的训练样本)
- Pasting: 不放回抽样
- 组合模型会比单个模型效果好很多 左图-一颗随机树 右-随机森林



- Boosting——把出错的部分权重加大 à 修正错误的地方
Adaboost: 训练下一个分类器的时候, 把上一分类器出错的部分权重增大 (特别关注), 不断纠正错误



Gradient Boosting: 让后面的模型不断地去减少前面模型的残差, 使得残差越来越小

六. 维度缩减

1. 投影 projection 高维->低维
2. 主成分分析法 PCA
3. 流形学习 Manifold Learning
基于流形假设

七. Tensorflow

使用 tensorflow (谷歌开发):
并行计算
使用 tensorflow 的一些方法

八. 卷积神经网络 Convolutional Neural Networks

CNN - 图像识别, 复杂的视觉任务

应用:

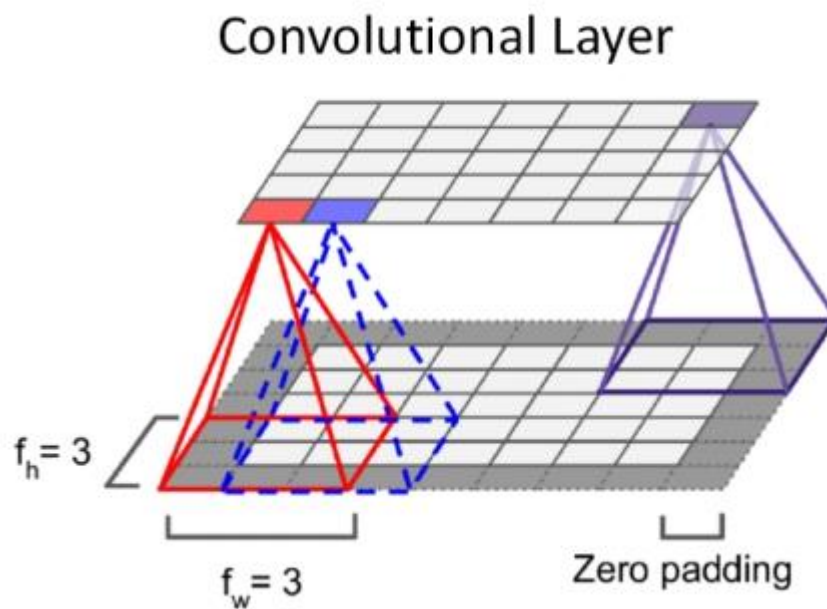
图像检索、自动驾驶、自动音频分类

结构:

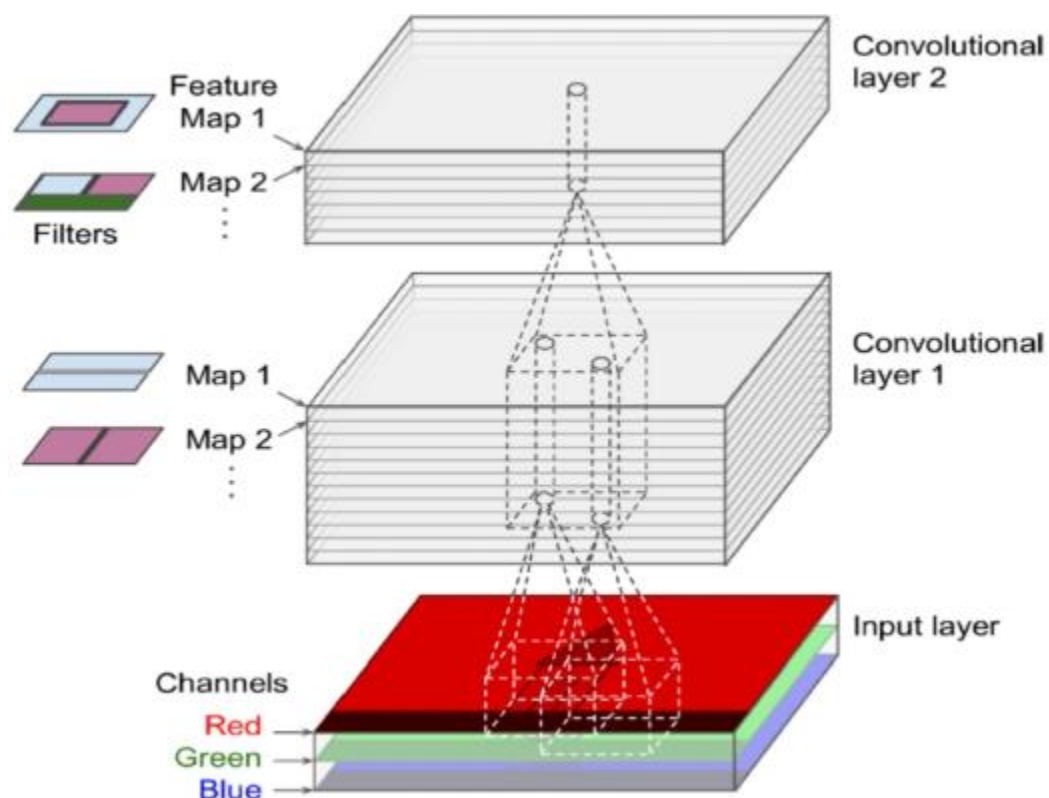
1. 卷积层

一些术语:

- 深度/depth
- 步长/stride
- 填充值/zero-padding: 补 0 可以不干扰原图, 相当于加了一个无信息点
- 过滤器/卷积核: filters



每一层用多个 filters, 会增加层数 (厚度)



在训练过程中，大小不断缩减，层数不断增大，变成一个类似一维的结构（“棍”），便于和输出层神经元做一个全连接。

2. 池化层

因为卷积网络在训练时图像的储存和训练会耗费大量的内存和 GPU，故引入池化层来缩减图片大小。

下采样：把尺寸变小

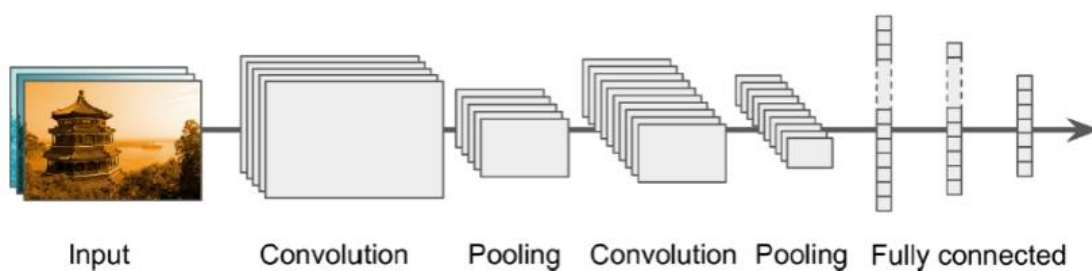
上采样：把尺寸变大

一般不会对原始图像做 pooling，而会对 feature map 做 pooling

3. 全连接层

和普通的神经网络中的结构差不多

一个典型的 CNN 结构：



九. 循环神经网络

RNN: 适合处理序列, 有“历史记忆”

训练公式:

$$\mathbf{y}_{(t)} = \phi\left(\mathbf{x}_{(t)}^T \cdot \mathbf{w}_x + \mathbf{y}_{(t-1)}^T \cdot \mathbf{w}_y + b\right)$$

其中 x 是输入, y 是历史输出 (“记忆”) 部分

结构类型:

