《人工智能导论》课程期末总结

信息学院　张冰清 2016202126

所在小组：第一组

概述

人工智能在计算机学科中是一个重要的分支学科，人工智能的发展历史与计算机学科一样的悠久。1950 年，英国科学家艾伦图灵提出了图灵测试，开启了人工智能的发展方向。

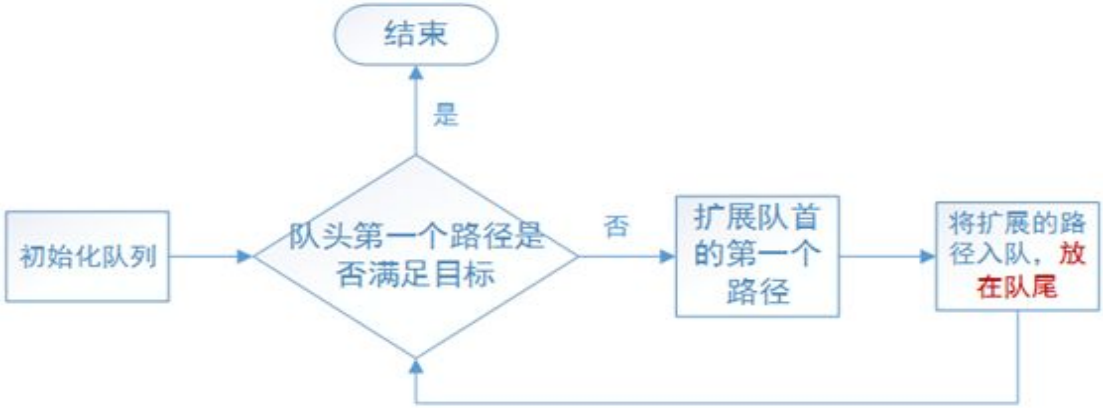
人工智能目前主要有三个学派，分别为 :符号主义、连接主义以及推理主义。不同的学派在历史的不同时期都发挥着重要的作用。

这门课程，从经典的推理形式的人工智能算法出发，讲到了目前主流的人工神经网络 DNN。并且学习了 Skit-learn 以及 Tensorflow 的使用。

一、经典搜索算法

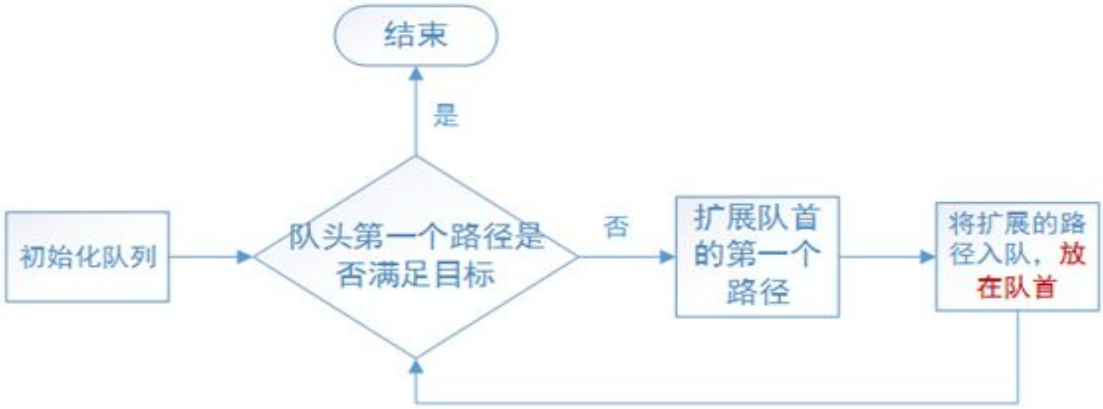
由于大量的人工智能问题最终可以归结为搜索问题，因此，使用搜索算法可以尝试解决人工智能问题。最简单的搜索方法就是枚举法，但是枚举显然是 NP-Hard 问题，因此难以推广在大网络中。使用网络中的搜索算法可以将规模进行减少。网络搜索算法主要有广度优先搜索以及深度优先搜索。

广度优先搜索主要流程下图所示



算法思想为从一个节点出发，找到其他临接节点，并将他们压入队列中。重复上述过程，直到队列为空。

与之对应的另一个算法为深度优先搜索算法，主要流程下图所示：



深度搜索的思想是贪心算法，每次找到一个未发现的节点，压入队列中，如此重复该过程， 直到搜索完所有的节点。

上述两种算法是最为经典的搜索算法，它们在小规模的网络上就有非常优良的性质。但是实际过程中，人工智能搜索网络为一个非常复杂的大规模网络，因此上述的两种算法很难在短时间内搜索完成。一种解决办法是使用并行的 BFS 和 DFS。在众核处理器上，每一个处理器核心从不同的节点出发，实现上述的串行算法。另一种解决方案就是对上述算法的改进。

第一种算法是爬山算法，这种算法是对深度优先搜索的改进。改进之处在于引入了距离的概念，每一次搜索寻找距离最近的节点。

第二种算法是束搜索算法，束搜索算法的思想就是设定最大的队列数，防止某一层的扩展过大导致搜索效率的下降。

其他的搜索算法还有分支限界、A* 算法。

在棋类游戏中，在搜索的过程中使用博弈过程，常用的算法为 MINMAX 算法。Minimax 算法普遍用于博弈游戏中，一方要将自己的利益最大化，而另一方要让对方的利益最小化。

但是在国际象棋这一类稍微复杂的游戏中，需要进一步的进行剪枝操作，使用了 Alpha-Beta 剪枝操作。大大的降低了算法的复杂度。

二、传统机器学习算法

2.1 机器学习概述：

机器学习最原始的解释就是通过输入大量的数据来让计算机学会解决问题的流程而不用人为的去编写规则。更为准确的定义为：

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

简单来讲，就是在任务 T 中学习经验 E，最终获得 P 的概率对该任务进行评估。 因此机器学习可以不用人在程序中添加系列的规则而取得好的成果。

目前，机器学习主要可以分为 4 类：

监督学习、无监督学习、半监督学习以及增强学习。

2.2 机器学习常用的算法概述：

构造间隔理论分布：聚类分析和模式识别

- 人工神经网络
- 决策树
- 感知机
- 支持向量机
- 集成学习 AdaBoost
- 降维与度量学习
- 聚类
- 贝叶斯分类器

构造条件概率：回归分析和统计分类

- 高斯过程回归
- 线性判别分析
- 最近邻居法
- 径向基函数核

通过再生模型构造概率密度函数

- 最大期望算法
- 概率图模型：包括贝叶斯网和 Markov 随机场
- Generative Topographic Mapping

近似推断技术

- 马尔科夫链
- 蒙特克罗算法
- 变分法

2.3 部分机器学习算法简介

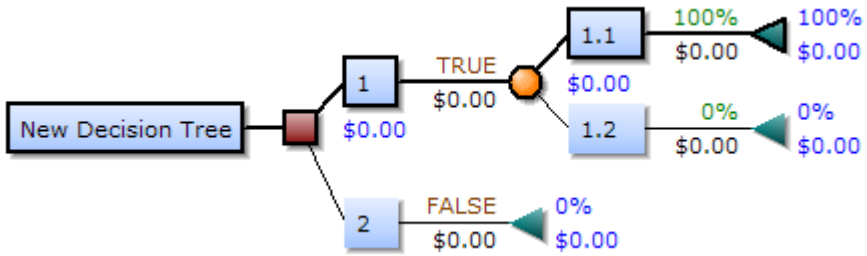
2.3.1 决策树

决策树本质上是一个数据结构。他代表的是对象属性与对象值之间的一种映射关系。树中每个节点表示某个对象，而每个分叉路径则代表某个可能的属性值，而每个叶节点则对应从根节点到该叶节点所经历的路径所表示的对象的值。决策树仅有单一输出，若欲有复数输出，可以建立独立的决策树以处理不同输出。

一个决策树包含三种类型的节点：

1. 决策节点
2. 机会节点
3. 终止节点

如下图所示



决策树模型的优点有：

解释性强

易于理解

在相对短时间内对大型数据做出较为优良的结果

决策树的缺点：

容易出现过拟合现象

当数据集过大的时候，相应的决策树模型将会非常的庞大

2.3.2 支持向量机

是在分类与回归分析中分析数据的监督式学习模型与相关的学习算法。给定一组数据，每一个数据可以被标记为属于两个类别中的一个或者另一个数据类型。SVM 通过表示空间中的点，这样映射就使得单独类别的实例被尽可能宽的明显的间隔分开。然后，将新的实例映射到同一空间，并基于它们落在间隔的哪一侧来预测所属类别。

支持向量机的优点：

数学理论的基础较为完善
在分类问题中，一般能够取得十分优良的成果

支持向量机的缺点：
SVM 仅仅试用与二元分类问题，多元分类问题中，通常将多类任务减少为二元分类问题。
解出模型的参数很难理解
2.3.3 线性回归
机器学习算法中最为简单的回归模型。最基本的二元线性回归模型为 y = Wx+b 给定一系列的数据点，通过调整参数 W、b 使得 y 与数据集的点的均方误差达到最小的方法。
线性回归的基础之上，有一些扩展模型，比如多项式回归，用来模拟数据点的分布不均匀并且呈现非线性的变化的规律。
线性回归的优点：
容易将数据进行向量化，使用矩阵运算可以优化代码
学习速度快，在极其短的时间内取得效果
参数简单，基本上不需要使用超参数来进行调整

线性回归的缺点
只能够处理极其有限的数据类型，对于稍微复杂的情况，线性回归的效果并不是很好

2.3.4 感知机模型
感知机模型是最早的人工神经网络模型之一。由 1957 年提出的一种前馈神经网络，是一种二元线性分类器。
感知机模型是生物神经细胞的一种简单的抽象。神经细胞大致可以分为：树突、轴突以及细胞体。单个神经元可以视为如下的两种状态：激动的时候为是，不激动的时候为否。神经元从多处树突收到传入信号，最后在轴突处传出信号。
因此，人工神经网络有着类似的处理的流程，有多个输入自变量代表着神经元的多维的树突结构，经过激活函数门处理之后产生一个输出 y。感知机模型在本质上仍然是一种线性回归的模型，优点在于：通过每一部分都相对较为简单

的结构综合产生较好的性能。缺点在于：没有进行反向传播，因此无法对参数进行有效的调整。

三、目前主流机器学习算法

3.1 人工神经网络概述

人工神经网络在机器学习以及认知科学领域，是一种模仿生物神经网络的结构和功能的数学模型以及计算模型。神经网络由大量的人工神经元联结进行计算。大多数情况下，神经网络能在外界信息的基础之上改变内部结构，是一种自适应系统。即具备学习的能力

3.2 深度学习与深度神经网络

深度学习是机器学习主流的一个分支，是一种试图使用包含复杂结构或由多重非线性变换构成的多个处理层对数据进行高层抽象的算法。

深度学习算法主要的数据结构为深度神经网络（DNN）。DNN 的结构主要包括输入层，多层的中间隐藏层以及最后的输出层。

深度神经网络是一种判别模型，使用反向传播算法进行训练。权重的更新通常是使用随机梯度下降的方法进行求解。并且通过学习率来控制神经网络的更新速度。常见的神经网络有卷积神经网络、循环神经网络、深度信念网络、生成对抗网络。

3.3 卷积神经网络概述

卷积神经网络（CNN） 是一种前馈神经网络，它的神经元可以响应一部分覆盖范围内的周围单元。

卷积神经网络由一个或者多个卷积层以及顶端的全连接层组成，同时也包括关联权重和池化层。这一性质使得卷积神经网络能够利用输入数据的二维结构。与其他深度学习相比，卷积神经网络在图像以及语音识别方面能够给出更好的结果。这一模型可以使用反向传播进行训练，相比于其他的深度、前馈神经网络，卷积神经网络需要考量的参数更少。

卷积神经网络的结构主要有卷积层，线性整流层、池化层、损失函数层以及最后的全连接层。

卷积层：卷积神经网络的每层由若干的卷积单元组成，每个卷积单元的参数都是通过反向传播算法的最优化得到的。卷积运算的目的是提取不同特征的输入。

线性整流层： 使用激活函数如 RELU、SIGMOID 来增强判定函数和整个神经网络的非线性特征。

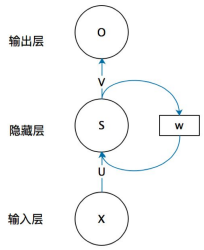
池化层：它实际上是一种降采样方式，有多重不同形式的非线性池化函数，最为出名的是最大池化方式。通过池化层后，可以减少特征的数目，也在一定程度上防止了过拟合的现象

损失函数层：用来惩罚网络训练过程中出现的预测结果与真实结果的差异。它通长是整个网络的最后一层。

卷积神经网络的经典模型有 LeNet, AlexNet, GoogleNet, VGG, ResNet

3.4 循环神经网络概述

循环神经网络（RNN） ，又名递归神经网络。主要用来处理时间序列信息，时间递归神经网络可以描述动态时间行为，因为和前馈神经网络（feedforward neural network）接受较特定结构的输入不同，RNN 将状态在自身网络中循环传递，因此可以接受更广泛的时间序列结构输入。手写识别是最早成功利用 RNN 的研究结果



四、第一实验报告： Arduino 小车避障实验

4.1 Arduino 语言介绍

Arduino 语言是建立在 C/C++基础上的，其基础是 C 语言，Arduino 语言只不过把 AVR 单片机（微控制器）相关的一些参数设置都函数化，不用我们去了解他的底层，让不了解 AVR 单片机（微控制器）的朋友也能轻松上手。

（1）结构

void setup() 初始化变量，管脚模式，调用库函数等

void loop() 连续执行函数内的语句

（2）功能

- 数字 I/O

pinMode(pin, mode)　数字 IO 口输入输出模式定义函数，pin 表示为 0 ~ 13，mode 表示为 INPUT 或 OUTPUT。

digitalWrite(pin, value)　数字 IO 口输出电平定义函数，pin 表示为 0 ~ 13，value 表示为 HIGH 或 LOW。比如定义 HIGH 可以驱动 LED。

int digitalRead(pin)　数字 IO 口读输入电平函数，pin 表示为 0 ~ 13，value 表示为 HIGH 或 LOW。比如可以读数字传感器。

- 模拟 I/O

int analogRead(pin)　模拟 IO 口读函数，pin 表示为 0 ~ 5（Arduino Diecimila 为 0 ~ 5，Arduino nano 为 0 ~ 7）。比如可以读模拟传感器（10 位 AD，0 ~ 5V 表示为 0 ~ 1023）。

analogWrite(pin, value) - PWM　数字 IO 口 PWM 输出函数，Arduino 数字 IO 口标注了 PWM 的 IO 口可使用该函数，pin 表示 3, 5, 6, 9, 10, 11，value 表示为 0 ~ 255。比如可用于电机 PWM 调速或音乐播放。

- 扩展 I/O

shiftOut(dataPin, clockPin, bitOrder, value)　SPI 外部 IO 扩展函数，通常使用带 SPI 接口的 74HC595 做 8 个 IO 扩展，dataPin 为数据口，clockPin 为时钟口，bitOrder 为数据传输方向（MSBFIRST 高位在前，LSBFIRST 低位在前），value 表示所要传送的数据（0 ~ 255），另外还需要一个 IO 口做 74HC595 的使能控制。

unsigned long pulseIn(pin, value)　脉冲长度记录函数，返回时间参数（us），pin 表示为 0 ~ 13，value 为 HIGH 或 LOW。比如 value 为 HIGH，那么当 pin 输入为高电平时，开始计时，当 pin 输入为低电平时，停止计时，然后返回该时间。

- 时间函数

unsigned long millis()　返回时间函数（单位 ms），该函数是指，当程序运行就开始计时并返回记录的参数，该参数溢出大概需要 50 天时间。

delay(ms)　　延时函数（单位 ms）。

delayMicroseconds(us)　延时函数（单位 us）。

- 数学函数

min(x, y)　　求最小值

max(x, y)　　求最大值
abs(x)　　计算绝对值
constrain(x, a, b)　约束函数，下限 a，上限 b，x 必须在 ab 之间才能返回。
map(value，fromLow，fromHigh，toLow，toHigh)　约束函数，value 必须在 fromLow 与 toLow 之间和 fromHigh 与 toHigh 之间。
pow(base, exponent)　开方函数，base 的 exponent 次方。
sq(x)　　平方
sqrt(x)　　开根号

- 三角函数

sin(rad)
cos(rad)
tan(rad)

- 随机数函数

randomSeed(seed)　　随机数端口定义函数，seed 表示读模拟口 analogRead(pin)函数 。
long random(max)　　随机数函数，返回数据大于等于 0，小于 max。
long random(min, max)　随机数函数，返回数据大于等于 min，小于 max。

- 外部中断函数

attachInterrupt(interrupt, , mode)　外部中断只能用到数字 IO 口 2 和 3, interrupt 表示中断口初始 0 或 1，表示一个功能函数，mode：LOW 低电平中断，CHANGE 有变化就中断，RISING 上升沿中断，FALLING 下降沿中断。
detachInterrupt(interrupt)　中断开关，interrupt=1 开，interrupt=0 关。

- 中断使能函数

interrupts()　　使能中断
noInterrupts()　　禁止中断

- 串口收发函数

Serial.begin(speed)　　串口定义波特率函数, speed 表示波特率, 如 9600, 19200 等。

int Serial.available()　判断缓冲器状态。

int Serial.read()　　读串口并返回收到参数。

Serial.flush()　　清空缓冲器。

Serial.print(data)　　串口输出数据。

Serial.println(data)　　串口输出数据并带回车符。

- Arduino 语言库文件

①官方库文件

EEPROM　　- EEPROM 读写程序库

Ethernet　　- 以太网控制器程序库

LiquidCrystal - LCD 控制程序库

Servo　　- 舵机控制程序库

SoftwareSerial - 任何数字 IO 口模拟串口程序库

Stepper　　- 步进电机控制程序库

Wire　　- TWI/I2C 总线程序库

Matrix　　- LED 矩阵控制程序库

Sprite　　- LED 矩阵图象处理控制程序库

②非官方库文件

DateTime - a library for keeping track of the current date and time in software.

Debounce - for reading noisy digital inputs (e.g. from buttons)

Firmata - for communicating with applications on the computer using a standard serial protocol.

GLCD - graphics routines for LCD based on the KS0108 or equivalent chipset.

LCD - control LCDs (using 8 data lines)

LCD 4 Bit - control LCDs (using 4 data lines)

LedControl - for controlling LED matrices or seven-segment displays with a MAX7221 or MAX7219.

LedControl - an alternative to the Matrix library for driving multiple LEDs with Maxim chips.

Messenger - for processing text-based messages from the computer

Metro - help you time actions at regular intervals

MsTimer2 - uses the timer 2 interrupt to trigger an action every N milliseconds.

OneWire - control devices (from Dallas Semiconductor) that use the One Wire

protocol.

PS2Keyboard - read characters from a PS2 keyboard.

Servo - provides software support for Servo motors on any pins.

Servotimer1 - provides hardware support for Servo motors on pins 9 and 10

Simple Message System - send messages between Arduino and the computer

SSerial2Mobile - send text messages or emails using a cell phone (via AT commands over software serial)

TextString - handle strings

TLC5940 - 16 channel 12 bit PWM controller.

X10 - Sending X10 signals over AC power lines

以上库文件都需要下载到编译环境（如下目录：arduino-0022\hardware\libraries）中才能使用。

4.2 实验展示

基本行动设计部分

- run() 前进函数

digitalWrite 左右轮分别前进

analogWrite 调节左右轮，PWM 比例 0~150 调速，左右轮差异略增减

- brake() 停车函数

digitalWrite 左右电机 PWM

analogWrite PWM 调速为 0

- back() 后退函数

digitalWrite 左右电机分别后退

analogWrite PWM 比例 0~150 调速，左右轮差异略增减

- left() 左转函数(左轮不动，右轮前进)

digitalWrite 左电机不转，右电机前进

analogWrite PWM 比例左 0，右 150

- spin_left(int time) 急左转函数(左轮后退，右轮前进)

digitalWrite　左电机后退，右电机前进
analogWrite　PWM 比例左 150，右 150

三、避障简易程序设计部分

（1）小车程序模块

- 基于红外线探测传感器避障
- 基于超声波传感器避障
- 基于红外线超声波传感器的综合避障

①基于红外线探测传感器避障

- 元器件：红外线传感器两个，分别安装在小车前方两侧
- 工作原理：红外线传感器发出红外线信号，若没有障碍物，则不会传回任何信息；若有障碍物，传回信息。具体表现在硬件上是高低电平的变化情况。

红外避障基本思想：

1）基于单节点单步长的深度优先搜索

2）每一次程序循环开始时，读取 SR_2 SL_2 参数，获取红外传感器的信息。如果两个参数均为高电平，表示前方没有任何的障碍物。如果左高右低，说明小车右边遇到障碍物；如果左低右高，小车左边遇到障碍物；如果同时出现障碍物，小车首先向后退，然后随机选择一个方向，继续前进

示例程序(同时低电平）：

```
brake();
delay(300);
back();
delay(400);
int a = random(11,13);
if(a%2 == 0){
left();
delay(500);
```

```
9.  }
10. else{
11.   right();
12.   delay(500);
13. }
```

②基于超声波传感器避障

- 元器件：超声波（建立在舵机平台） 一个
- 工作原理: 与之前的红外线传感器不同的是：超声波传感器返回的信号为模拟信号，因此可以利用模拟信号较为精确的测量出相关的数据。具体做法如下：1. 发出超声波信号，遇见障碍物后反弹，收到信号。2. 计算两次信号出入的时间。 3. 用时间/58
- 58 的计算方法： T(秒） = （2* D(米)）/344 (假设声音速度为 344）———> T = 0.0058 * D -- 厘米 = 微秒 / 58

基于超声波传感器避障基本思想：

1）遇见障碍物之前，无任何变化，调用 run()

2）遇见障碍物思想同红外线

两种方案的比较：

- **红外避障：数字电路的输入输出，因此程序简单，小车反应速度灵敏。但是硬件基础的特性使得避障不够灵敏**
- **超声波避障：模拟信号的产生，程序复杂。但是避障效果好，但是盲区较大，因此边际会发生碰撞**

③基于红外线超声波传感器的综合避障

- 优先级顺序：红外线 > 超声波
- 可以在突发事件出现时，最快速度停下来。

后续优化设想：

①基于蓝牙连接的控制

②图像识别初步

4.3 源代码位置

见 github 第一组第一次提交

4.4 个人参与情况

Arduino 小车的组装

红外避障与超声模块结合避障的程序实现编写（源代码见 4.3）

课堂汇报展示

五、第二次实验——蓝牙连接 & 视频识别初步

5.1 实验背景：

计算机视觉研究的一个重要主题就是让机器看懂图片。在许多应用场景中，通过机器的视觉系统，可以对视频或者图片信息进行初步的判断。比如，围栏监控摄像头需要实时的进行自动判断是否有人翻越围栏。这其中，最重要的一点就是要让机器“看得懂”图片。

5.2 利用蓝牙连接来操控小车

实验步骤：

1.小车烧录蓝牙控制程序

2.手机安装蓝牙控制 APP

3.手机与小车进行蓝牙连接

4.APP 上完成连接，操控小车运动

实验原理：

接受字符，根据不同字符的不同，小车进行不同的运动。如下所示：

```
getstr=Serial.read();
if(getstr=='A')
{
run();
}
else if(getstr=='B'){
```

```
7.  back();}
8.  else if(getstr=='C'){
9.  left();}
10. else if(getstr=='D'){
11. right();
12. }
13. else if(getstr=='F'){
14. brake();}
15. else if(getstr=='E'){
16. brake();
17. }
```

5.3 openCV 简介

OpenCV 是一个 2000 年发布的开源计算机视觉库，有进行物体识别、图像分割、人脸识别、动作识别等多种功能，可以在 Linux、Windows、Android、Mac OS 等操作系统上运行，以轻量级、高效著称，且提供多种语言接口。

其中 openCV 3.3 新特性为：

对深度学习（dnn 模块）提供了更好的支持，dnn 模块目前支持 Caffe、TensorFlow、Torch、PyTorch 等深度学习框架。

新版 OpenCV 兼容以下热门网络架构：

- AlexNet
- GoogLeNet
- ResNet
- SqueezeNet v1.1
- VGG-based FCN
- ENet
- VGG-based SSD
- MobileNet-based SSD

使用 opencv 的一般处理流程

- 从硬盘加载模型；
- 对输入图像进行预处理；
- 将图像输入网络，获取输出的分类。

常用的一些方法

用“create”方法直接从各种框架中导出模型

- cv2.dnn.createCaffeImporter
- cv2.dnn.createTensorFlowImporter
- cv2.dnn.createTorchImporter

使用“读取”方法从磁盘直接加载序列化模型：

- cv2.dnn.readNetFromCaffe
- cv2.dnn.readNetFromTensorFlow
- cv2.dnn.readNetFromTorch
- cv2.dnn.readhTorchBlob

5.4 视频识别初步

一些关键技术

1 通过 DroidCam 将手机摄像头转化为网络摄像头

2 使用 open CV 捕捉摄像头信息

3 通过 c v 内置 dnn 进行评估

处理流程

1. 连接网络摄像头

VideoStream(src=1).start()

2. 转化为正确的格式，使用 openCV 的 dnn 模块进行分类（前向传播）

3. 判断预测好的标签，如果正确率大于 default 值（0.2），在图片中框起来

5.5 个人参与情况

openCV 初步使用（本次实验使用互联网上已经训练好的模型）

openCV 程序连接手机摄像头

课堂展示

六、第三次实验——自动驾驶的简单实现

Arduino 小车上进行自动驾驶

优点：

已经有大量的内置传感器，第二次展示使用的视频识别技术

缺点：

内置电池续航时间短，充电时间长，充电宝电压不够

使用场景不明确（可以在宿舍楼道，操场，教室等不同场景下进行使用)

很难进行深度学习

GTA5 上进行自动驾驶

- 优点：
- 使用场景明确（在公路上进行正常的行驶）
- 更容易模拟真实场景
- 缺点：
- 程序较为复杂

在 gta5 上进行自动驾驶

原因分析

1. 更好的模拟真实的场景
2. 很多自动驾驶公司都使用 gta5 进行驾驶模拟
3. 可以使用神经网络进行训练

在 gta5 上进行自动驾驶

- 基础的接口及其模拟
- 在正确的道路中行进
- 如何生成并标注数据
- 如何训练数据
- 如何进行推理
- 使用视频识别技术
- 综合的实现

基础的接口及其模拟

利用 gta5 的内存信息？

- 了解到 openAI 以及一些自动驾驶公司使用 GTA 5 提供的官方接口，可以获取到游戏运行的所有内存信息
- 我们尝试寻找 API，发现这种接口只对企业级用户合作开放

使用截屏库（比如 pillow）

- Pillow 库是一个图像处理里面常用的库，接口简单

```
printscreen =  np.array(ImageGrab.grab(bbox=(0,40,800,640)))
cv2.imshow('window',cv2.cvtColor(printscreen, cv2.COLOR_BGR2RGB))
```

使用模拟鼠标键盘控制（pyautogui）

使用 pyautogui 进行键盘控制

PyAutoGUI 是一个纯 Python 的 GUI 自动化工具，其目的是可以用程序自动控制鼠标和键盘操作，利用它可以实现自动化任务

```
pyautogui.keyDown('w')
time.sleep(3)
pyautogui.keyUp('w')
```

最终结果：没反应

使用 directX 接口进行控制

- GTA 的图形接口为 directX
- 程序来源：https://stackoverflow.com/questions/14489013/simulate-python-keypresses-for-controlling-a-game

```
if __name__ == '__main__':
    PressKey(0x11)
    time.sleep(1)
    ReleaseKey(0x11)
    time.sleep(1)
```

```
W = 0x11
A = 0x1E
S = 0x1F
D = 0x20
```

如何正确的在道路上正确的行进

利用传统的道路识别技术

步骤一：彩色图片转换为灰度图片

方式：采用加权和的方法：0.2126 红+ 0.7152 绿+ 0.0722 蓝

步骤二： 高斯模糊

使用高斯模糊算法，使上一步得到的图像更为平滑

方法：

1. 选择照片并确定像素值
2. 找到所选像素的相邻区域（小区域）
3. 取原始像素和相邻像素的值，并通过某些加权算法将它们平均。
4. 用输出的平均值替换原始像素的值
5. 对所有像素执行此操作

步骤三： Canny 边缘检测

一个图片中的边缘就是像素值突然发生跳动的部分

方法：

1. 选择照片中的像素点
2. 识别所选像素元左侧和右侧的像素元组的值
3. 计算两组之间的差异
4. 将所选像素的值更改为 3 中计算的差值
5. 对所有像素执行此操作

经过这般操作之后，图片中的非边缘部分应该为黑色，边缘部分为白色

步骤四： 缩减视野

防止把天际线识别为道路

方法：对图像进行裁剪

步骤五： 基尔霍夫变换

画两根线

输出两个方程式

但是上述的方法不可行

使用深度学习进行自动驾驶

第一部分：数据集的采集与标注

- 正确的数据集应该包含有当前的状况以及正确的决策。
- 本次实验中：数据集的一个条目应该包含有一张图片以及正确的输出
- 简化起见：汽车在道路上主要有前进、左转、右转（分别对应键盘上的 w a d
- 由于我们不能够使用 gta5 的内存信息，因此不知道机动车的速度，采用的办法是定期刹车

使用 win32api 库获取键入信息

- 注意：此 api 只能够在 windows 下使用
- 核心代码

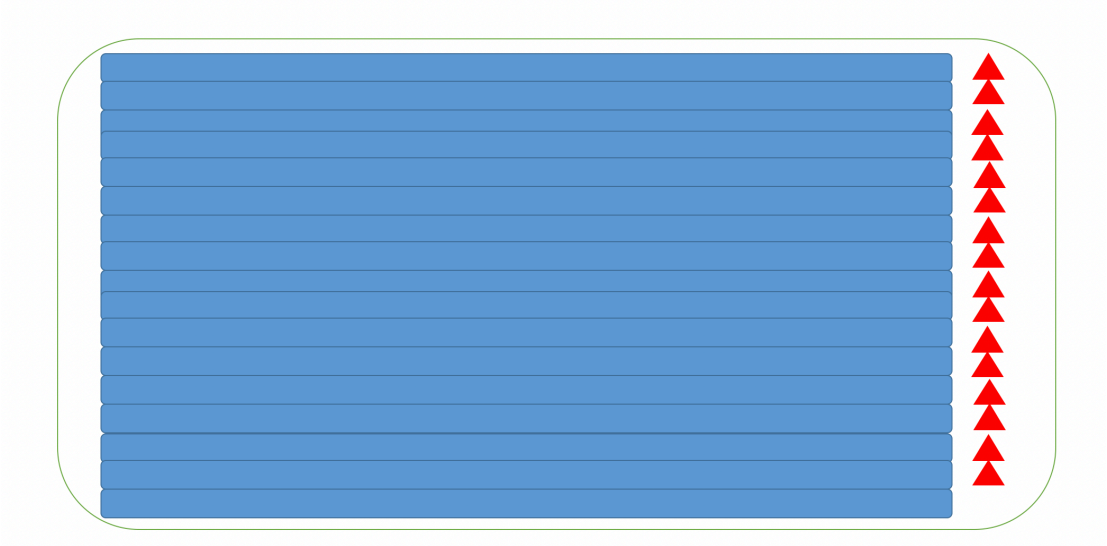
```
keyList = ["\b"]
for char in "WASD":
    keyList.append(char)

def key_check():
    keys = []
    for key in keyList:
        if wapi.GetAsyncKeyState(ord(key)):
            keys.append(key)
    return keys
```

标注数据集程序

核心代码

```
screen = grab_screen(region=(0,40,1600,940))
screen = cv2.cvtColor(screen, cv2.COLOR_BGR2GRAY)
screen = cv2.resize(screen, (160,90))
keys = key_check()
output = keys_to_output(keys)
training_data.append([screen,output])
```

数据集



失误

- 请使用数据库存储数据
- 如果数据量大的话，放在多个文件存储
- 减少在标注数据中的违规操作（比如闯红灯、骑线行驶）

结果

- 总共标注了 5 万张图片
- 总计耗时 5 个小时

第二部分：数据集的一些处理

- 最开始直接将标注好的数据集直接进行归一化然后训练，发现汽车只会走直线
- 分析数据集发现，五万张图片中，有 4 万多张图片标注为前进

数据的一些处理

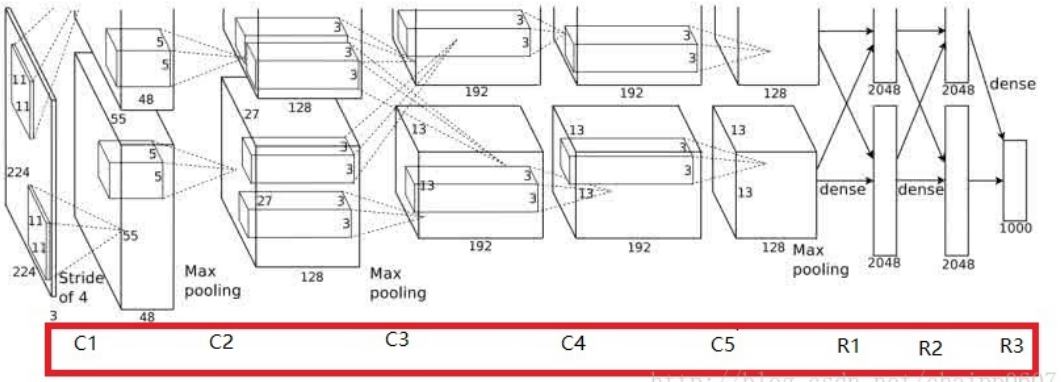
- 将数据集等份划分为 w a d
- 并且进行快速的随机重排，打乱数据集

- 最终数据集的大小 1 万多张图片（90M）

第一个大坑——tensorflowgpu 的安装

- 安装顺序(个人)
- Nvidia GPU 驱动程序（可选）
- CUDA
- cuDNN
- tensorflowgpu
- 之前一直用的是 cuda 10.0 浪费了一天的时间！！！

Alexnet 简述



Alexnet 的特点

- DropOut 实现鲁棒性
- 使用 ReLU 激活函数（引入了非线性性以及稀疏性，并且收敛块，解决了梯度消失
- LRN：对局部活动的神经元创建了竞争机制

使用 Alexnet 进行训练

- 最初的参数
- 训练轮次：EPOCHS = 10
- LR = 1e-3

训练效果不好

- 最终的参数
- EPOCHS = 120
- LR = 1e-4

一些反思

- 可以使用 Googlenet、ResNet 等更加强大的深度神经网络
- 训练集远远不够（至少需要 20 万张图片才能够见效果）
- 训练时请关闭游戏（Alexnet 大小）

第四部分：结果的推理(之后会再提)

实验平台

- CPU：Intel Core i7-8750H
- 内存：ddr4 2666 8GB*2（双通道）
- GPU：Nvidia Geforce GTX 1060

Tensorflow object dection API 概述

- Tensorflow models 的一部分

（见 github）

视频识别大升级

两次视频识别技术对比

- 第二次展示的视频识别
- 使用网络：已经训练好的未知名网络
- 使用数据集：未知名的数据集，只有 15 个类别
- 精准度：差
- 全新升级的视频识别
- 使用网络：轻量而又有效的 MobileNet
- 使用数据集：著名的 COCO 数据集（超过 200 中常见物品）
- 调用接口:Tensorflow Object Dection API
- 准确度：优

如何做

- 把内置的教程 objection_dection_tutorial.ipynb 更改一下就可以完成上述功能

实验的结论

深度学习的推理能力有待提高

- 真正的实验可能与教科书上的建议有所出入，比如 LR 的选择
- 数据集的人工标注真的很难受
- 使用更大的数据集以及更加强大的网络可以提高实验的准确性
- 有问题，上 google

个人在实验中的参与情况：

数据集的制作

模型训练

模型推理

视频识别（Tensorflow Object Dection API）

PPT 制作

课堂展示

七、结论：

1. 连接主义引导下的 AI 技术正在改变世界，通过机器学习，计算机可以实现许多之前没有办法实现的功能

2. 机器学习领域非常丰富，课后还需要大量的学习与大量的应用

3. AI 发展速度十分的快，我们还要不断的去了解 AI 领域的最新进展

源代码路径：见 README