# 北京航空航天大学

2021-2022 学年第一学期

# 《Python 编程与智能车技术》
# 结课报告

班　级周五 8-10 节　　学　号21374230

姓　名 黄乙笑　　成　绩_____

所在小组组长：孔令琪

组　　员：黄乙笑，杨辰润，罗俊午，范宏伟

# 1 Python 语言知识体系

Python 的强大之处，在于他灵活简单的语法、多样的数据类型、面向对象，还有各种强大的官方库或者第三方库。

## 1.1 Python 的基础语法

与其他各种语言一样，python 也有是以顺序、选择、循环为基础的，逻辑都差不多。

## 1.2 多样的数据类型

### 1.2.1 一般的数据类型

Python 也有整形、浮点型、布尔型、字符、字符串等，但 python 对整形和浮点型的限制没有 c 和 c++ 这么严格，在某种意义上，可以把二者看作是一样的。

### 1.2.2 特殊的组合数据类型

列表 list、元组 tuple、集合 set、字典 dic，是 python 特有的几种组合数据类型，字符串也可以算入其中。通过这些数据类型，我们能较为方便地处理大量的数据。各个数据类型可以相互嵌套，组成更加复杂的数据类型。

列表通过 [ ] 或者 list 来创建, 类似于 c 语言中的数组。但列表可以进行索引、切片、遍历等操作，还能使用 append、clear、copy、insert、pop、remove、reserve 等方法进行复杂的操作。

元组通过 ( ) 或者 tuple 创建，与列表类似，但他们之间最大的区别是：元组是不可以改变的，列表是能改变的。所以 append、clear 等方法都不适用于元组。

集合通过 { } 或 set 创建，与数学中的集合一样，集合是无序、排他的，可以增增删元素。集合有四种操作：交 & ；并 |；差 - ；补 ^ 。也有 add、clear、len 等方法。常用集合来进行去重。

字典通过 { } 或者 dic 创建，每一个元素是一个键值对，即一个键及其对应的值。字典也是无序、不可重复的。对于字典 dict，我们可以用 dict[键] 来进行索引和添加新键。dict.get (key,default) 方法可以返回 key 所对应的值，如果键 key 不存在则返回 default。

## 1.3 面向对象的编程

Python,Java,C++ 是面向对象的语言的代表。面向对象即“万物皆对象”，通过对象能实现各种操作。

在 python 的面向对象编程中，主要有以下几个关键的概念：类 class，类变量，局部变量，实例变量，对象，实例化，方法，继承，方法重写。

类变量是在类中共有的变量，类中所有函数均可访问。局部变量是类中某个函数的变量，其他函数不能访问。实例变量以“self. 变量名”来进行定义，之作用于调用此方法的对象，只能通过对象名访问，而不能通过类名访问。

对象是通过类定义的数据结构实例，包括两个数据成员（类变量和实例变量）和方法。实例化是创建一个类的实例，类的具体对象。

方法是类中定义的函数。继承是从父类中继承字段和方法，而方法重写是当父类中的方法无法满足需求时，对其进行改写。

## 1.4 第三方库

除了 python 自带的 math、random、turtle 等库，全球的开发者还共同开发了很多第三方库，我们一般需要通过 pip 来安装。

在我们的学习中，就用到了 jieba、worldcloud、numpy、pandas、matploit、pillow、sklearn 等库，为我们做项目提供了极大的方便。这也是 python 受到大家喜爱的原因之一吧。

# 2 机器学习

机器学习从上世纪中叶开始，经过了几十年的发展，已经较为成熟。机器学习可以粗略分为有监督学习、无监督学习、半监督学习和强化学习。其中，较为基础的是有监督学习。

有监督学习是既有特征，又有标签的学习。常见的有神经网络、线性回归、逻辑回归、支持向量机 (SVM)、朴素贝叶斯、决策树等，可分为分类和回归两大类。特征是输入的变量, 一般用向量 X 表示，标签就是要输出的结果, 一般应用向量 Y 表示。在训练集中，每个训练样本都有 X 和 Y。

无监督学习与有监督学习相比，没有标签。在实际应用中，无监督学习比有监督学习应用范围更广，但技术上更加困难。无监督学习大多应用在聚类和 k-means 中。

在 python 中，常用的机器学习库有 sklearn 等。可以实现数据预处理、数据集、特征选择、特征降维、模型构建 (分类、回归、聚类)、模型评估 (分类、回归、聚类)。值得一提的，sklearn 自带很丰富的数据库，有手写数字数据集、波士顿房价数据集，还有乳腺癌和糖尿病数据集等。

功能更加强大的还有 Tensoflow,Pytorch,Paddlepaddle 等开源的机器学习库，通过这些库，即使不太理解机器学习算法的实现原理，我们仍然能写出性能很好的机器学习程序。

# 3 车辆识别项目

## 3.1 基于全连接神经网络的车辆识别

全连接神经网络，又叫深度神经网络 (Deep Neural Networks)，是最基础的神经网络之一。

正如其名字所描述的，DNN 是一种全连接的前馈神经网络。

神经网络就是以我们的生物上的神经元为基础出发的。生物上的神经元，通过树突接受来自感应器或者上一神经元的信号，并通过轴突输出给效应器或者下一神经元；神经网络中的神经元，或者说一个节点，接收来自输入层或上层神经元的数据，并输出给输出层或者下层神经元。人脑有多层，神经网络也有多层。全连接，就是前一层的每个神经元与后一层的每个神经元都有连接。这使得全连接神经网络较为复杂，需要的数据多，计算量也很大，非线性的拟合能力较好。当我们的隐藏层较多时，才能称作全连接神经网络，否则只是浅层神经网络。研究单个的神经元，我们要研究它的输入、输出，更要研究从输入到输出的过程。简单而言，将所有输入按照一定的权重 (weight) 加和，再加上偏置 (bias)，经过激活函数之后，进行输出。如此完成一个从输入到输出的过程之后，模型会有一个输出值，与目标值之间有差距，我们就用损失函数来刻画这个差距。DNN 的最终目标就是：通过改变权重、偏置，使得损失函数最小。上述过程的数学表达如下：

$$z = a_1\omega_1 + a_2\omega_2 + ... + a_k\omega_k + b = A \times \Omega + b$$

$\omega$为权重，b 为偏置。A 为$a_1, a_2 \cdots$组成的向量，$\Omega$为$\omega_1, \omega_2 \cdots$组成的向量

$$y = \sigma(z)\sigma\text{为激活函数，}y\text{为输出}$$

$$\text{改变}A, \Omega, \text{使得}loss(Y)\text{最小,}Y\text{为}y\text{组成的向量}$$

如何选取激活函数、损失函数，如何让机器改变参数使得损失函数最小，是我们需要思考的三个问题。常用的激活函数有四种：

$$Sigmoid\text{函数}: \frac{1}{1+e^{-x}}$$

$$Tanh\text{函数}: \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$Softsign\text{函数}: \frac{x}{1+|x|}$$

$$ReLU\text{函数}: max(0, x)$$

如果不使用激活函数，那么输入与输出仍然会保持线性关系，无法拟合非线性关系。激活函数有各自的优缺点，如 Sigmoid 函数与 Tanh 函数，在自变量很大或者很小的时候会出现梯度消失的情况，即函数值随自变量变化不大。我们一般在输出层采用 Softmax 函数，即归一化指数函数，将每一个结果以其概率表现出来，总和为 1。损失函数较常用的有平方差和交叉熵两种。平方差函数：$loss(Y) = \frac{1}{n}\sum_{i=0}^{n}(y_i - y)^2$，刻画输出与标准值的差距，常用于回归。交叉熵函数：$H(p, q) = -sump(x)\log q(x)$，其中 q 是经过 Softmax 激活之后的输出概率分布，p 是实际的概率分布，x 是每个概率分布对应的分量。它刻画预测的概率与实际的概率的差值，输出的常用于分类。如在智能车分类中，输入是一辆摩托车，的概率：摩托车 0.7，汽车 0.2，货车 0.1。即 $p = (1, 0, 0); q = (0.7, 0.2, 0.1);\ H(p, q) = -(log0.7 + 0 + 0) = -log0.7$。如何改变参数？最常用的是基于梯度下降法的反向传播算法。即让参数值沿着函数的梯度

的反方向下降。为减少计算量，一般采用随机梯度下降，使用一个 batch 的数据进行参数更新。为了防止过拟合，减小计算量，我们也可以使用 drop out，在每一轮训练中随机忽略一些节点。

## 3.2 基于卷积神经网络的车辆识别

卷积神经网络 (Convolutional Neural Networks, CNN) 有三个特点：局部连接、权重共享、下采样，这写特性使得它摆脱了全连接神经网络的缺点：结构不灵活，参数太多，训练速度慢。

局部连接就是选取局部进行识别，关注某些特殊的区域。权重共享就是各个局部的神经元参数相同。下采样就是让尺寸变小，像素变少。三个方法一起在保证准确度的同时减少了卷积神经网络的计算量。

卷积神经网络一般的步骤是输入 $\rightarrow$ 卷积层 $\rightarrow$ 池化层 $\rightarrow \cdots$ 卷积层 $\rightarrow$ 池化层 $\rightarrow$ 全连接 $\rightarrow$softmax$\rightarrow$ 输出

卷积层是用卷积核对图像进行卷积，可以理解为：输入图像 (input image) * 卷积核 (kernel) = 特征 (feature map)，是卷积操作，即矩阵之间的数乘。这是最基础的单核单通道卷积。一般的图像是三通道的，我们就需要把每个通道卷积之后的特征进行进行合并，此处每个通道的卷积核是不一样的，因为一个卷积核只能表示一个特征。还有多核卷积，即每个通道也采用不同的卷积核进行卷积，最终叠加。

池化层 (pooling) 又叫做下采样层，用来压缩数据, 缩小 feature map 尺度。常用最大池化 (max pooling) 或者平均池化 (average pooling) 两种池化层进行池化。

损失函数等其他参数与全连接大致相同。

# 4 对本课程的学习总结

## 4.1 个人学习反思

在学习方法上，我并不认为这门课是一门一般的通识课，而是一门入门人工智能的课程。我在课上认真做笔记，重要的程序在课下要么自己打一遍，要么看把它看懂。但是由于课时安排节奏太快，课余事件也不是特别充裕，越到后面，特别是机器学习那一块，我就放低了对自己的要求。对 sklearn 和 paddlepaddle 库中的东西搞不太懂，调参数、加层数也不明白为什么要这样做，这样做有什么好处有什么坏处。

在小组作业中，我们组的选题是用 pygame 做一个游戏。有杨辰润学长搭建框架，写主要部分，我和罗俊午学长做细节。我主要负责的是写 other_thing.py，做出漂浮物和能量球。刚开始我真的是一头雾水，一是不知道我该写什么，二是不知道我该怎么写。需求不是很明确，只说了一个大概的方向；对 pygame 完全没有了解，面向对象也学的不是特别扎实。后来经过自己的思考、与学长的讨论、阅读写好的程序框架，对游戏的逻辑以及需要我去实现的功能有了大致的了解。然后再在百度的帮助下，跌跌撞撞写出了一个大概。但是程序逻辑很混乱，我又自己进行了一边 code review，杨辰润学长把我们的程序整合到一起之后，又改了一些细节，才使得整个程序看起来很清爽。写代码其实是一门艺术。但是也有遗憾之处，本来我想的是这两个东西会动，但是我写着写着就把这事忘了，在我看最终版的时候才发现 speed_x,speed_y 根本没用上，遂作罢。

## 4.2 对课程的总结和建议

在寒假选这门课的时候，就看到了学长学姐们对这门课非常一致的评价：“很硬，但值得一上。”当时仰仗着自己有一丢丢 python 基础，对 python 语言又比较感兴趣，就选了这门课。现在这门课结课了，如果要我给一个评价，我也会说那句话：“很硬，但值得一上。”不仅仅是在知识层面上得到了很大的提升，在个人学习能力上、小组合作能力上，我都感觉我有进步。

我是一个很爱接受新鲜事物、什么都想去尝试一下的人（这篇总结便是用才学的 latex 写的）。我可能什么东西都知道一点，但不专精，在 pyhon 这件事上也是如此。以后有了更多的空闲时间，我应该会更深入地学习 python，学习机器学习相关内容。

对于这门课的建议，我觉得是否可以向教务处申请延长课时、增加学分，使得我们有更多时间去学习机器学习相关内容，做更多项目。