《人工智能》学习总结

刘一键　2014201886

《人工智能》这门课主要分为两部分，一部分以学习理论知识为主，同时辅以简单的代码讲解，另一部分就是动手实践，要求分组进行人工智能小车的开发，将软件与硬件相结合，在实际操作中应用机器学习、神经网络等知识。通过这一学期的学习，我认为自己收获很多，不仅逐渐深入的了解到人工智能这一研究领域，学习了相关的模型与算法，更是在实践过程中提高了自己应用知识的能力。在这份学习总结中，我将从理论知识体系和实践过程两部分进行阐述。

一、理论知识体系

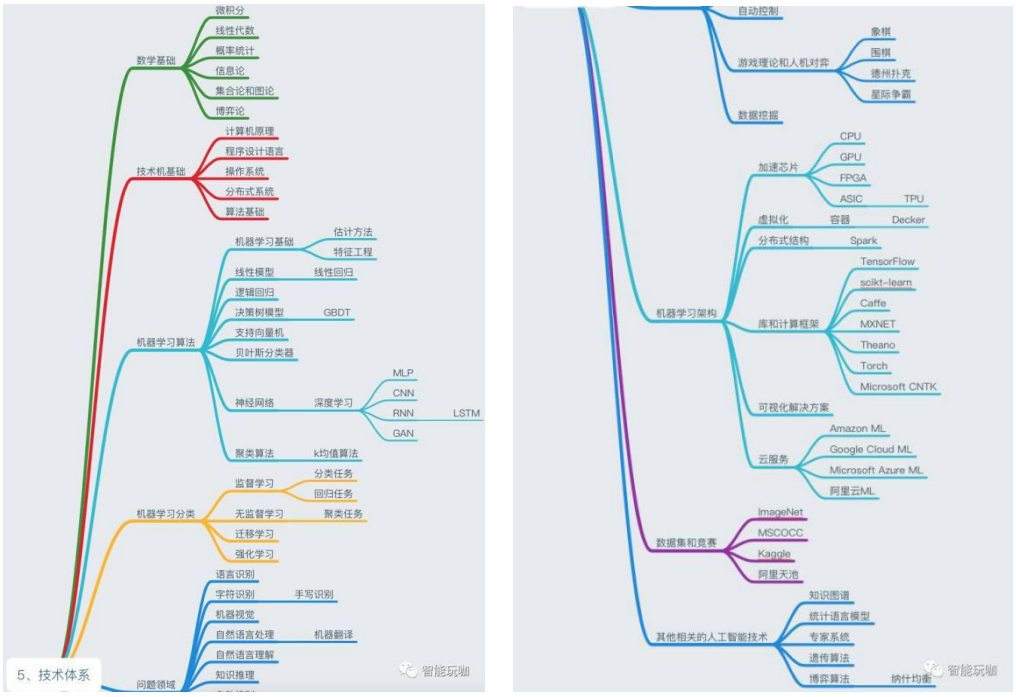
这门课的知识架构主要来自于教材《Hands-On Machine Learning with Scikit-Learn and TensorFlow》，书中的内容分为两个板块，第一板块是介绍机器学习，机器学习可以说是人工智能领域下一个最为重要的子类，另一个板块是介绍神经网络，神经网络是机器学习里面最有代表性的一种方法。回顾第一次课老师展示的课程大纲，通过这门课，我们将从“什么是机器学习”这样基本的问题，逐步深入到“如何使用 tensorflow 训练神经网络”，课程内容除了对基本概念和算法原理的讲解，还通过代码的演示让我们跟进一步的了解到这些模型与算法的工作机制。接下来我将首先阐述自己对于人工智能这一领域知识架构的理解，其次再谈谈我自己印象最为深刻的几个模型，并附以我调试代码的体会。

1、人工智能知识架构

机器学习是一种很特别的方法，它与传统的编程不一样，传统的编程需要通过人为给定明确的指令去工作，而机器学习却只需要人投入大量数据，它自己会完成直接编程无法做到的工作，因此有人说机器学习赋予了机器学习的能力。从实践的意义来说，机器学习就是利用数据训练出模型，再通过模型进行预测的方法。机器学习适用于解决或是优化这些问题：已有的解决方法需要大量手动调整工作的问题，传统思路下找不到解决方法的复杂问题，不稳定环境中的问题，数据集过于庞大的问题。通过机器学习，一方面是可以提高原来解决方案的工作效率，另一方案是针对原来找不到解决办法的问题给出解答。

人工智能领域的知识框架中包含了大量，，我在自己学习的时候参考了一张网络上的结构图，专业领域的诸多前辈们已经对人工智能的知识结构作了非常全面又清晰的划分，我通

过这些知识体系结构，也逐渐建立起自己对人工智能的理解。



在这张知识结构图中，可以看到，人工智能首先涉及到的是数学和计算机专业的基础知识，进一步的是机器学习的几大算法和模型，其中包括回归模型、决策树模型、支持向量机模型、贝叶斯分类模型、神经网络、聚类算法等。图中还清晰的展示了机器学习的分类，监督学习是已有训练样本，并有其对应的输出，通过这样的训练样本来训练模型，无监督学习是没有样本的，直接对数据进行建模，比如聚类算法，还有一种是半监督学习，是指综合利用有类标的和没有类标的数据来生成分类函数。其他的分类还有在线学习，指的是每输入一个样本都计算一次误差，进一步调整一下参数，而批量学习是指的一次性就输入所有的样本进行训练。除此之外还有基于实例的学习，基于模型的学习，迁移学习和强化学习。

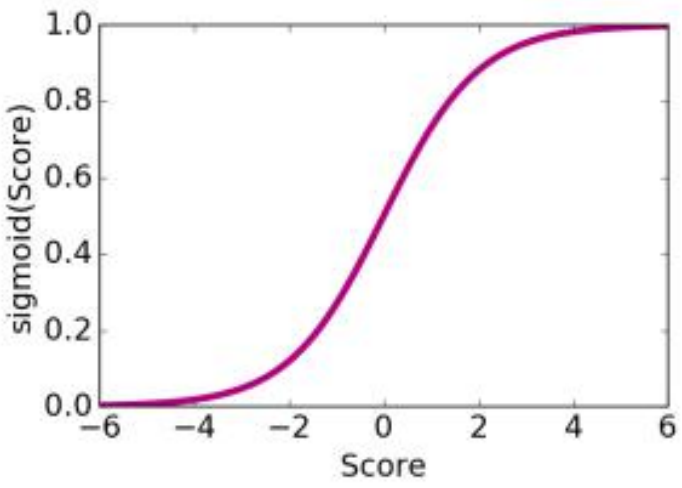
2、几个重要模型

一个典型的机器学习项目主要分为以下八个步骤：首先是观察到一个问题，再获取它的数据，讨论分析这些数据，甚至可以通过将数据可视化来深入理解这些数据，下一步就可以为机器学习算法准备数据集了，即将我们之前获取到的数据再加工，将其处理成标准的格式，便于算法直接引用，下一步就是进行模型的选择和训练，调整模型，接下来就可以显示出训练的结果，即预测结果，最后是发布、监视和维护这样一个智能系统。无论什么模型或者算法，机器学习项目基本上都是通过这样一个思路来展开工作的，有了这样通体的指导方针，我们就可以从发现问题开始一步步找到解决问题的方法。接下来我将详细介绍几个我印象比

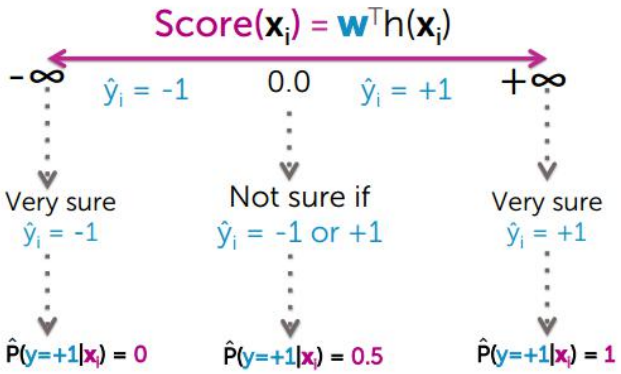
较深刻的模型以及我对代码的理解和调试代码的心得。

（1）回归模型

回归模型是机器学习里面比较常用也很基础的算法，通常情况下，我们都认为回归算法有两个重要的子类，即线性回归和逻辑回归，从它们的工作机制来看，线性回归处理的是数值问题，也就是最后预测出的结果是数字，例如在房价预测问题中，模型可以通过房屋的面积、地理位置等基本数据来预测出它的价格，最终得到的价格就是一个具体的数值。而逻辑回归可以说是属于分类算法，逻辑回归的预测结果其实是离散的分类，例如通过逻辑回归模型来判断一封邮件是否是垃圾邮件，它只有是或者不是这两种分类。表面上看两种回归模型有很大的不同，但实际上逻辑回归只是对对线性回归的计算结果加上了一个 Sigmoid 函数：$y = sigmod(x) = \frac{1}{1+e^{-x}}$。这个函数有这样的特性，即当 x 趋于负无穷时，y 的值趋近于 0，而当 x 趋近于正无穷时，y 的值趋近于 1，x=1/2 时，y 就正好等于 0。



为了更好地理解线性回归模型和逻辑回归模型之间的区别，我尝试着对其中的公式作了推导，分类模型实际上就是这样一个函数：$\hat{y}_i = sig\,\mathrm{mod}(Score(x_i))$，其中 $Score(x_i) = \sum_{j=0}^{D} w_i h_j(x_i) = W^T h(X_i)$，$\hat{y}_i$ 即为最后的预测值，取值 1 或-1，即代表是否属于某一分类。这里有一个比较重要的概念，即预测结果的准确度，我们可以定义：如果 $\hat{P}(y=+1|x) > 0.5$，则 $\hat{y} = +1$，否则 $\hat{y} = -1$，根据上面的公式，可以知道 $Score(x_i)$ 的值是分布到负无穷到正无穷的区间上，而对应的概率 P 的值是 0 到 1，而 sigmod 函数正好可以完成这一转换。



因此对于损失函数，我们可以知道：$ll(w)=\sum_{i=1}^{N}\ln P(y_i\mid x_i,w)$，又因为，已知 y 的值只有 1 或-1 两种情况，因此$ll(w)=\sum_{i=1}^{N}[1[y=+1]\ln P(y=+1)+1[y=-1]\ln P(y=-1)]$，而 P 我们已经确定为 sigmod 函数，进一步分析可以得到：$P(y=+1\mid x,w)=\frac{1}{1+e^{-w^Th(x)}}$，而$P(y=-1\mid x,w)=1-P(y=1\mid x,w)=\frac{1}{1+e^{-w^Th(x)}}$，代入以上公式可得到$ll(w)$的简化表达式：$ll(w)=-(1-1[y_i=+1])w^Th(x_i)-\ln(1+e^{-w^Th(x_i)})$，得到损失函数以后，我们将通过偏导来对模型中的参数进行调整：$\frac{\partial ll}{\partial w_j}=-(1-1[y_i=+1])\frac{\partial w^Th(x_i)}{\partial w_i}-\frac{\partial\ln(1+e^{-w^Th(x_i)})}{\partial w_i}$

$=h_j(x_i)[1[y_i=+1]-P(y=+1\mid x_i,w)]$。

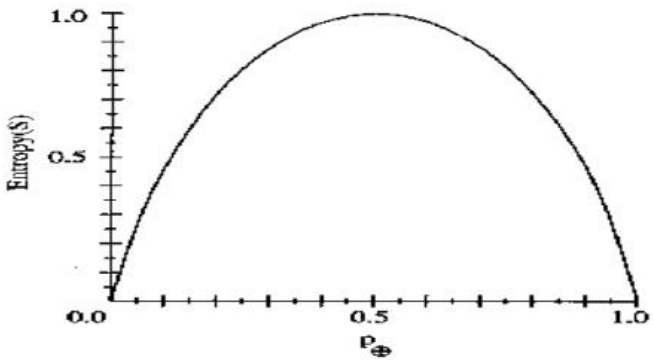
（2）决策树模型

决策树是一种利用概率分析来图解问题的方法，也有定义说是逼近离散值目标函数的方法。在机器学习中，决策树是一种做预测的模型，代表了对象属性和对象值之间的一种映射关系，它利用了信息学理论中信息熵的概念。决策树是一种监督学习，因为它给定了一堆样本，每个样本都有一组属性和一个类别，这些类别就是类标，我们通过训练模型可以得到一个树形结构的分类器。

顾名思义，决策树就是一种树形结构的模型，其中每个节点都分别代表了不同的属性，即对这一属性的测试，每个节点的分支代表了每种属性的不同取值，相当于是测试输出，最低端的叶节点就是我们最终想要得到的事物的类别。再具体执行的过程中，我们可以认为它

是一系列 if-then 规则的集合。决策树模型适用的问题有：实例是由（属性，值）这样的数据对表示的，目标函数具有离散的输出值，即有确定的几种分类，训练数据中可以容纳一些错误，训练数据中也可以包含缺少属性值的实例。构造决策树的过程实际上就是在选择逻辑判断和属性，对同一组实例，可能有多种树结构可以符合它的要求，但是为了让树的预测能力更强，我们要尽可能构造更小的决策树。构造一棵最小的树实际上是 NP-Hard 的问题，因此只能采用启发式的策略来做。

决策树中熵的定义为 $Entropy(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$，熵函数画出来如下：



另一个重要的定义是信息增益，一个属性的信息增益就是由于使用这个属性分割样例而导致的期望熵的降低。$Gain(S, A) = Entropy(S) - \sum_{v \in Value(A)} \frac{|S_v|}{|S|} Entropy(S_v)$，在每一步的属性选择时，即需要选择信息增益最大的一类属性，这样可以带来最大程度的信息熵的降低，进而提升模型预测的效率。

决策树模型是应用最广的归纳推理算法之一，它的工作机制很清晰简单，但是它在实际应用的过程中也会存在一些问题，比如会出现过度拟合的情况，最基本的决策树没有考虑数据中的噪声，因而生成的决策树会与训练数据完全拟合，这就意味着它对测试数据是过拟合的，如果测试数据中含有噪声，这样的完全拟合反而不具有很好的预测能力。解决过拟合问题的方法有例如对决策树进行剪枝的技术，有包括向前剪枝、向后剪枝等方法。同时，决策树也有一些有点，其一便是它可以生成我们可以理解的规则，其二是它的计算量不是很大。

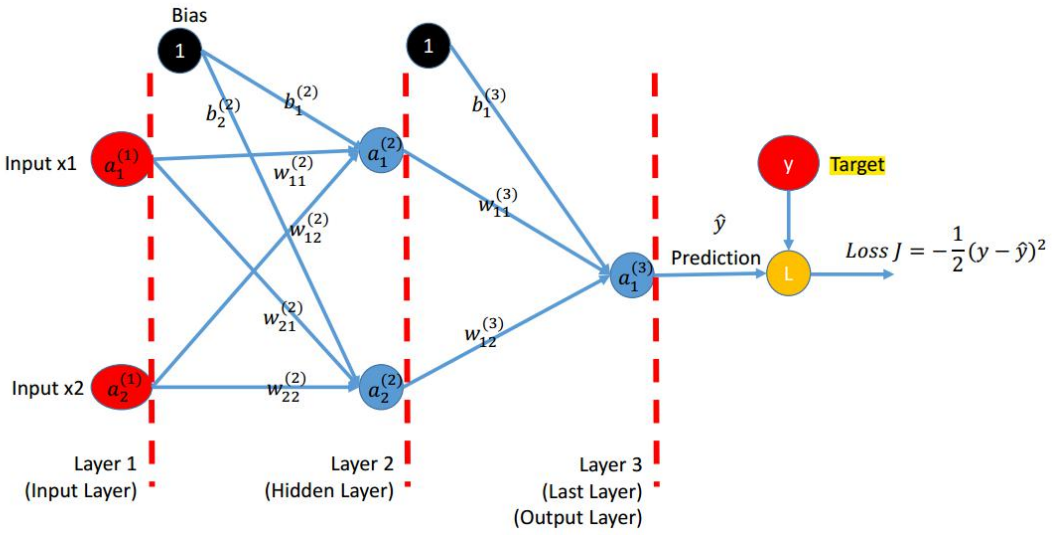
（3）神经网络

（3.1）神经网络结构

神经网络也就是人工神经网络（ANN），通俗来讲也有人称它为连接模型，因为它模仿了生物神经元之间的连接关系。我在学习这一章节的时候，为了让自己深入理解它的工作机

制，我通过查阅资料，自己做了一下各种公式的推导，并且运行了代码，实现了手写数字的识别。

首先，构造一个（3，3，2，1）结构的简单网络，如图，3 个网络层，3 个神经元，2 个输入，1 个输出：

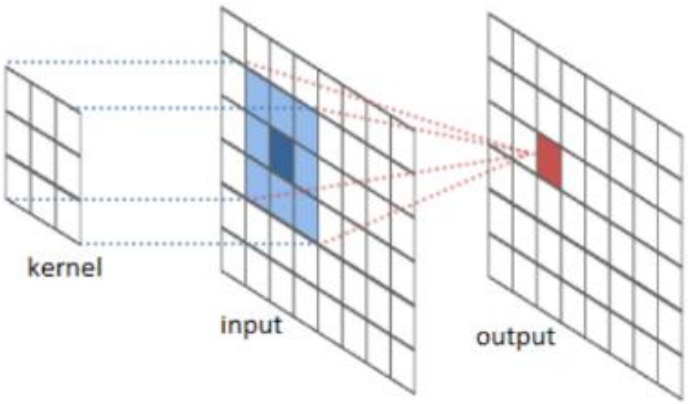


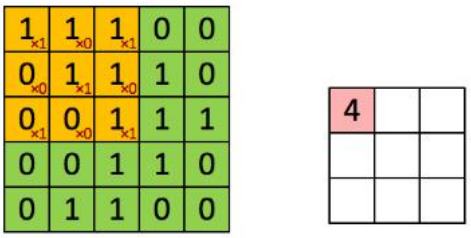
图中形象的展现了每一层网络与上一层之间的关系，若令 $z=\sum_{i=1}^{2} w_i x_i + b$，$a=\partial(z)=\frac{1}{1+e^{-z}}$，此处的$\partial$函数选择使用 sigmod 函数，它的选择还有许多，比如 tanh 函数、ReLU 函数、还有 Leaky 等。Loss J 实际上是反映了为预测值和真实值之间的差异，因此根据残差的定义，可以将 Loss J 函数定义为：$LossJ=-\frac{1}{2}(y-\hat{y})^2$。整个网络计算的过程就像图像中展现的这样，在每一层网络中都添加一些参数，如此可以模拟出非线性的关系，最后得到了初步的模型，模型实际上就是指各层网络的参数值组合，以及整个网络结构的选取。那么得到模型以后，如何进行优化呢，此处就从数学的角度进行了计算，根据 Loss J 的值，我们可以对每一层的每个参数求偏导，看每个参数对 Loss J 的影响有多大。在通过一个学习率函数，来调整参数的值，使得下一次计算出的 Loss J 可以变小，我们定义 $W=W-\gamma\frac{\partial J}{\partial W}$，$b=b-\gamma\frac{\partial J}{\partial b}$，这个过程即我们所说的反向传播的过程。已知 $A^{(i)}=\sigma(W^{(i)}\cdot A^{(i-1)}+b^{(i)}\cdot N)$，$J=\frac{1}{N}\cdot\mathrm{sum}(Loss(A^{(LAST)},Y))$，根据数学计算，我们可以得到：$\frac{\partial J}{\partial \hat{y}}=y-\hat{y}$，$\frac{\partial J}{\partial B^{(i)}}=\frac{\partial J}{\partial A^{(i)}}A^{(i)}(1-A^{(i)})$，即$\frac{\partial J}{\partial b^{(i)}}=\frac{1}{N}sum(\frac{\partial J}{\partial B^{(i)}})$，对于权重有

$\frac{\partial J}{\partial W^{(i)}} = \frac{\partial J}{\partial B^{(i)}} A^{(i-1)^T}$，$\frac{\partial J}{\partial A^{(i-1)}} = W^{(i)^T} \frac{\partial J}{\partial B^{(i)}}$。根据以上公式，一层层倒退，更新权重和偏移量的值，达到优化模型的目的。

（3.2）卷积操作原理

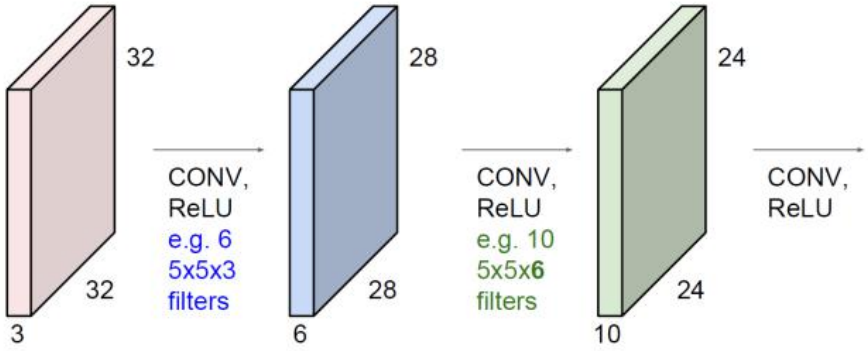
了解了神经网络最基本的工作机制，我又深入学习了卷积神经网络（CNN），卷积神经网络在模式分类领域应用广泛，该网络避免了对图像的复杂前期预处理，可以直接输入原始图像进行计算。一般，卷积神经网络包括两层，其一是特征提取层，在这一部分即使用卷积的算法让每个神经元的输入与前一层的局部接受域相连，并提取它的特征，其二是特征映射层，网络的每个计算层由多个特征映射组成，每个映射都是一个平面，平面上所有神经元的权值相等。首先，理解卷积操作的含义：





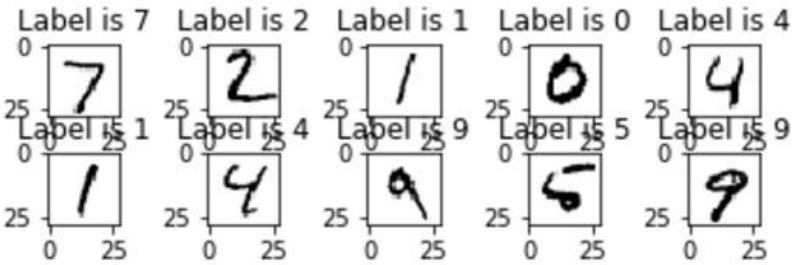
如果原图像是 5*5 像素，即 5*5 的像素矩阵，每个方格中的值表示其 RGB 的取值，本实例中初始化为黑白图像，以 0 或 1 表示黑白两种情况，卷积核即为黄色趋于 3*3 的矩阵，将卷积核在原图像矩阵上滑动（设置滑动步长为 1），将卷积核与覆盖区域作积（即对应元素相乘再相加），可得到第一个卷积的结果为 4，右侧的卷积结果的维度是由原图像大小和卷积核大小，以及滑动步长决定的，我们可以推导出，对于窄卷积（卷积核/滤波器长度<输入长度），输出长度为 $\frac{\text{输入长度}-\text{滤波器长度}}{\text{步长}}+1$，而对于宽卷积，输出是长度为

$\frac{\text{输入长度}+2\cdot\text{补零长度}-\text{过滤器长度}}{\text{步长}}+1$。因此整个图像的卷积过程可以表示为如图：



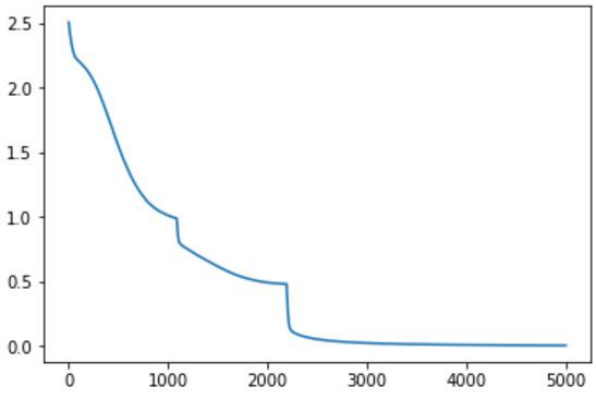
（3.3）代码运行结果

在用代码进行实践时，我使用了 MINIST 数据库的手写数字数据集。原始的数据集是一行长度为 784 的数据，每一行代表了一张 28*28 的图像，如图：



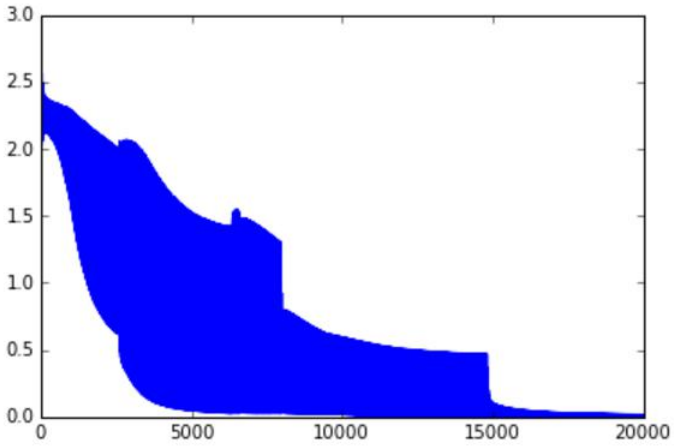
为了检测代码的计算速度，我首先使用了训练数据为 10 的数据集，得到了 Loss J 的变化曲线，可以看出，预测效果是变得越来越好了，J 的值最后逐渐趋近于 0，表示预测的准确性趋近于 100%，从这个曲线中可以看出，模型的训练效果是分阶段性的有所突破，这种阶梯型的下降实际上与网络的初始结构和自定义的学习率有很大的关联。

```
In  [12]: plt.plot(nn.J)
Out[12]: [<matplotlib.lines.Line2D at 0x34f28a0828>]
```



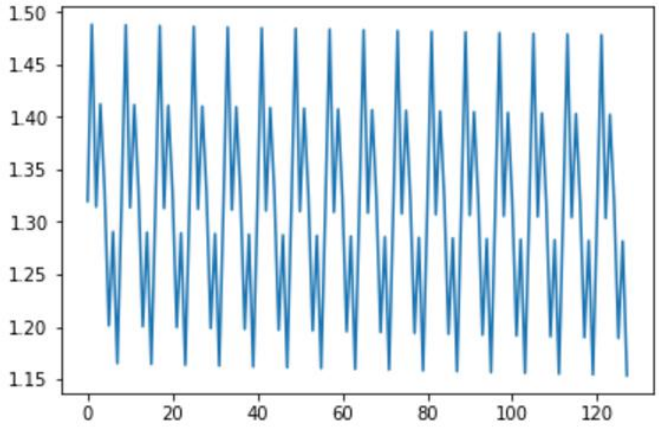
接下来我又进一步将训练集扩大到 100，得到了如下图像：

```
In  [36]: plt.plot(nn.J)
Out[36]: [<matplotlib.lines.Line2D at 0xe876650>]
```



显然，在这张图中，J 的变化过程与上一张图有很大的差别，在训练集为 100 的情况下，J 的波动更为明显，将图放大以后，我们可以看到：

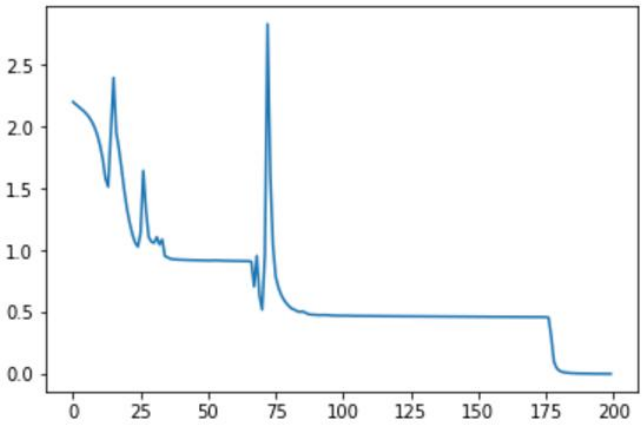
```
Out[21]: [<matplotlib.lines.Line2D at 0x853480be80>]
```



这也可以说明，根据梯度下降的算法，到了最后可能模型不会处于一个稳定的自由状态，而是会不停波动，此时我们就不必严苛的要求它要稳定在最低位置，而是给一个合理的足够小的范围即可认为训练已经结束。

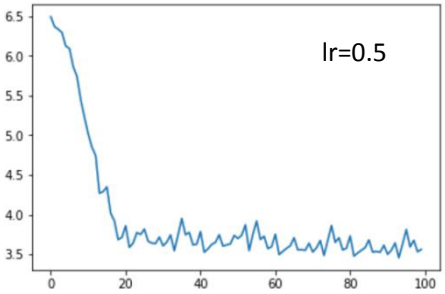
接着我又调整了网络的结构，将原来的一层卷积层，调整为了两层，并微调了里面一些参数，得到了这样的 J 的变化过程：

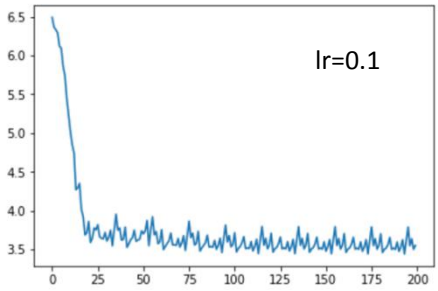
```
Out[11]: [<matplotlib.lines.Line2D at 0x8534ad3748>]
```



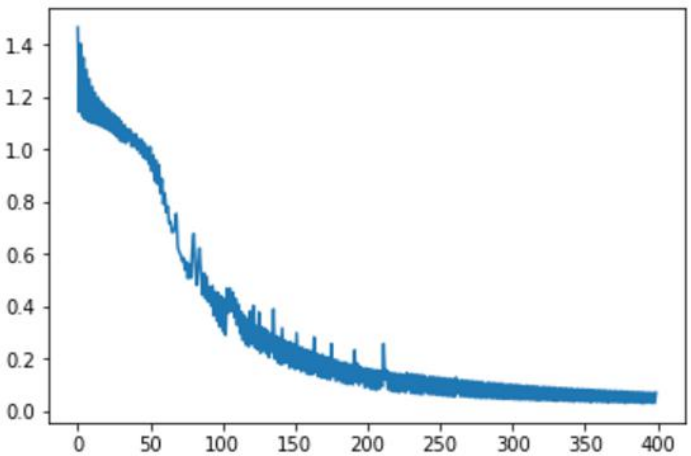
可以看出，卷积层变得更加复杂，模型却能更快地趋于最优，虽然其中经历了一些比较大的波动，但是最终也会逐渐趋于稳定，所以神经网络本身的结构，对于训练的效率也存在一定的影响，但目前只能依靠专家的经验，手动设置结构。

之后我尝试着使用了数据量为 60000 的数据集，虽然最后模型的准确率不是很理想，但是我通过调整学习率，得出学习率对模型的影响。在这个例子中，学习率越大，Loss J 下降的速度更快，但是如果学习率过大，则会出现后期波动特别明显的情况，此时就无法得到最优的模型。






```
Out[30]: [<matplotlib.lines.Line2D at 0x8bdeb7ecc0>]
```



（4）其他模型

除了回归模型、决策树模型、神经网络，还有许多其他的常用的机器学习的模型，比如支持向量机模型（SVM），通过跟高斯核的结合，支持向量机可以表达出非常复杂的分类界线，从而达成很好的的分类效果。“核”事实上就是一种特殊的函数，最典型的特征就是可以将低维的空间映射到高维的空间；降维算法，是一种无监督学习算法，其主要作用是压缩数据与提升机器学习其他算法的效率，降维算法的主要代表是 PCA 算法（即主成分分析算法）；还有比较重要的是推荐算法，应用场景最普遍的就是购物网站对用户的产品推荐，推荐算法中最有名的算法就是协同过滤算法。

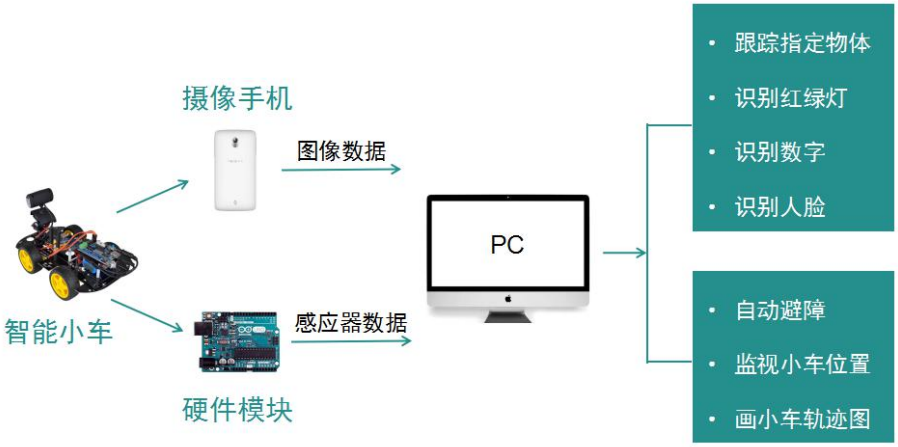
二、实践经验

在这门课的动手实践部分，我们以小组为单位，进行了人工智能小车的开发，前期主要按照老师的要求去完成小车结构的搭建，以及实现一些基本功能，后期我们逐渐加入了自己的想法，利用课堂理论知识所学，将机器学习的一些算法和模型融合进小车，丰富它的功能，期间我们遇到各种各样的问题，思考了许多的解决方案，我们小组每个星期都约定时间讨论，不得不说这样的实践经验极大地锻炼了我们的工程能力、动手能力、发现问题、解决问题的能力，更让人欣喜的是，在实现了这些原本以为非常高深莫测的功能以后，获得了深深的成就感，并且在亲自动手的过程，在脑海里留下了深刻的印象，这对于我们日后再进行这方面的开发或是研究，有着极大的作用。接下来我将阐述我们小组在实践部分做的一些工作，我们的想法，我们遇到的问题以及我们的解决方案，还有我作为小组成员在其中担任的角色，以及我自己的收获。

1、功能设计

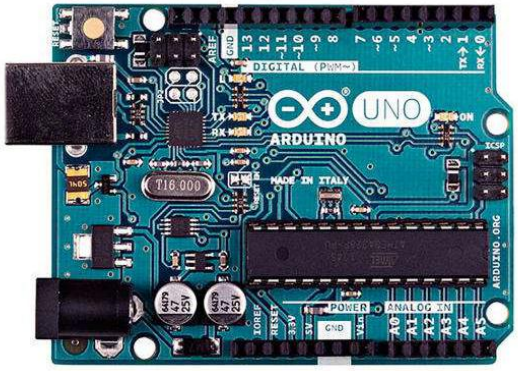
对于我们小组的智能小车，我的想法是实现一个智能机器人配送货物的流程。作为送货机器人，基础功能就是小车可以自行前往指定的目的地，并且在途中做到自动避开障碍物，以及识别红绿灯的功能，为了实时监控小车的位置，我们利用 GPS 模块获取其地理位置信息，以此为基础对其发出转向的指令，让它始终保持在前往目的地的正确方向上，为了简化问题，我们暂时还没有考虑实际的道路规划和实时路况信息，今后如果要完善我们的功能，可以加入地图中的实际道路数据和道路交通情况等信息，同时还需要做出最短路径规划的决策。在小车行进途中，我们一方面是需要沿着道路行进，这就需要实现一个寻迹的功能，具体方法就是通过图像识别，另一方面是要检测红绿灯，通过图像识别，判断当前是否是绿灯，

是否可以继续前进，还是需要等待。我们原本的想法是让小车可以对交通信号灯的倒计时进行判别，但是因为实际情况很复杂，小车获取到的图像非常复杂，要想从中提取出我们想要的区域比较困难，目前这一部分还在考虑中。另一个功能是跟踪指定物体，我们同样是利用图像识别来完成，应用场景就是希望小车可以跟着人行动，例如帮助人搬运重物。当送货机器人到达目的地以后，需要找到具体的一户人家，因而就涉及到了门牌号的识别，在这一块我们是从网络上找到了门牌号的训练数据集，利用神经网络的模型对其进行了训练。除此之外，还需要完成的是人脸识别，我们目前可以实现的是从一张图像中找到人脸的区域，这部分也是通过图像识别来完成，在未来我们希望将人脸区域提取出来以后，可以真正实现人脸识别，需要进一步做的工作就是对训练数据的搜集。



2、硬件 & 软件

在进行智能小车的开发中，其中很重要的一部分是对硬件方面的学习，因为以前未曾接触过这方面的内容，因此进行实践的第一步就是了解 Arduino 开发板的工作机制，以及一些感应器和促动器的工作方式，经过学习，我们知道开发版支持的编程语言正是最基础的 C 语言，再加上许多功能模块都有最基本的调用示例代码，因此在硬件方面我们很快就上手了。

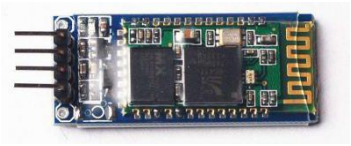


首先，需要用到的功能模块有超声波回声测距模块，用来识别距前方障碍物的距离，以便小车及时转弯，避开障碍物，不过它的工作模式有一定缺陷，即当小车以较大角度驶向障碍物时，由于波的反射原理，使得模块没有检测到侧前方的障碍物，这个问题我们目前还没有找到可行的解决方案，未来计划通过图像识别的方法来智能避障。

另一个模块是 GPS 模块，用于获取小车实时的地理位置（经纬度），由此我们就可以得到小车的行动轨迹，也便于我们对它进行远程的监视和控制。

蓝牙模块是用于通信的模块，它负责将小车搜集到的各种数据上传到我们的服务器（个人电脑/手机），也负责接收我们向它输送的指令，我们小组耗费了非常多的时间在蓝牙通讯这一部分。在前期我们遇到过蓝牙连不上、只能单向通讯、接收不到指令等等问题，通过不断使用测试代码对其进行测试，建立断点以定位出错的区域，最后解决了这一系列的问题。





在软件部分，我们编写代码、烧录程序是用的 Arduino 配套软件，对于数据的分析和处理我们用 python 实现，期间使用 python 完成通讯的部分我们遇到一些困难，图像数据传输的通讯通道很不稳定，有时会被迫中止，却找不到合理的原因，后来我们改蓝牙通讯为 wifi 通讯，目前的执行情况稍有提高。

三、总结

以上就是我对这门课的总结，在理论知识部分，我对人工智能的知识体系有了总体的把握，对机器学习领域重要的模型和算法都有了比较深的认识，尤其是其中的回归模型、决策树模型和神经网络。通过课后对代码的运行和调试，可以进一步发现模型的优缺点和模型的工作原理，通过真正将数据集利用起来进行训练和测试，我学会如何完整的实现一个机器学习项目，如何将这些理论知识运用到以后的实际工作中。同时，课后开发智能小车的小组项目，也是非常有意义的，我们可以将自己的想法付诸实践，并且与组员之间的思维碰撞极大地提高了我解决问题的能力，也加深了我对一些模型和算法的认识。机器学习的应用领域非常广，除了我们现在接触到的图像识别，还有许多别的应用方向，我还需要继续学习，巩固知识，开拓眼界，以理论结合实践的方法进行人工智能领域知识的探索。