**人工智能学习报告**

章晓慧 2015201972

人工智能可以说是近年来最火的话题之一，它是指利用计算机系统来模仿人类的感知、思维、推理等活动，近年来也得到了飞速的发展，为科技的发展，人类的进步作了很大的贡献，这个学期我通过选修“人工智能”这门课，也对人工智能有了进一步的了解。老师的教学分为三个部分，第一个部分主要对人工智能作了基本的介绍，还对arduino和aicar作了介绍，第二部分介绍了机器学习，第三个部分则介绍了更深层次的人工智能，即神经网络和深度学习。而本篇学习报告对我这个学期以来对“人工智能”的学习作了一个总结，也分为三个部分，也对应了老师所讲授的三个部分（并非按照顺序映射）。

1. **卷积神经网络学习心得**

老师在课程了进行了很多关于人工智能的算法的介绍，其中包括很多关于神经网络的介绍，其中我印象最深的卷积神经网络，因为在之前就经常在别的课程上听说这个名词，但是一直不曾真正了解这是一种什么样的算法，这次是第一次有了一个较为系统的学习，课后我也特地搜索资料对卷积神经网络进行了更多的了解，并将卷积神经网络与传统的神经网络进行了对比分析。

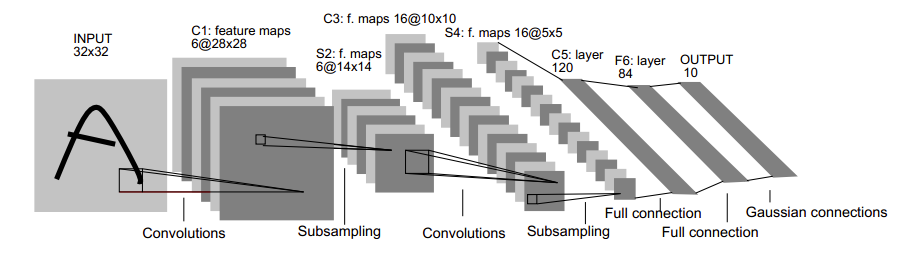
卷积神经网络的灵感来源于Hubel和Wiesel与1962年对于猫视觉皮层细胞的研究，他们发现大脑中的一些个体神经细胞只有在特定方向的边缘存在时才能做出反应，并提出了感受野的概念，也就是说每个细胞只能对一些特定的特征作出反应，比如有的是形状，有的是颜色。再想到传统的神经网络，在这种全连接的模式下，得到每个新图的像素点都需要依靠旧图的所有的像素点，连接数目非常庞大，当图片较大时，会需要很多的权值，同时也没有考虑到图片在相邻区域的相关性，而卷积神经网络采用了之前提出的感受野的概念，以及权值共享，解决传统神经网络的一部分问题。

传统神经网络中每个神经元都要与图片上每个像素相连接，这样的话就会造成权重的数量巨大，网络难以训练。卷积神经网络中每个神经元的权重个数都是卷积核的大小，这样就极大的减少了权重的数量。例如，假设有一张1000\*1000的图片，有10^6隐层神经元，那么在传统的神经网络中，每个神经元都将对应图中所有的像素，就会1000\*1000\*10^6=10^12个权重参数，但是如果采用卷积神经网络，每个神经元节点只与上一层的10\*10的接受域相连接，那么就只有10\*10\*10^6=10^8个权重参数，相较于传统的神经网络大大减少了。

那么如何实现这种部分连接呢，就使用了卷积核，这样的卷积核既达到了一个神经元对应部分区域的目的，也达到了提取图像的一种特定的特征的目的，在需要多种特征的时候就使用多个卷积核。同时为了方便处理，有时候需要保证卷积运算后得到的特征图的大小与之前保证一致，所以使用了padding，来为上一层特征图作填充。因为一张大小为n\*n的特征图，在使用大小为f\*f的卷积核以步长s进行卷积后得到的特征图大小为[(n-f)/s+1]\*[(n-f)/s+1]，所以为了使得特征图大小不变，我们需要为原图加一个padding值，设这个值为p，那么得到的新图大小则为[(n+2p-f)/s+1]\*[(n+2p-f)/s+1]，那么p就需要满足条件[(n+2p-f)/s+1] = n， 那么当步长s为1（较常用）时就可以推出p = (f-1)/2。

同时在卷积神经网络中，往往会使用池化的方法，通俗点说就是把特征图像区域的一部分求个均值或者最大值，用来代表这部分区域，池化利用了图像的局部相关性原理，在减少数据量的同时也保留了拥有的信息，一般会有最大池化、平均池化的方法，同时池化还可以分为无重叠池化和有重叠池化，其中有重叠池化可以防止过拟合问题。

我对于老师上课提到的Lenet-5模型作了更进一步的了解与学习，该模型常被用于作手写数字识别系统等，该模型包括输入层一共包括八层。即为输入层、卷积层、池化层、卷积层、池化层、卷积层（全连接层）、全连接层和最后的输出层。老师的ppt上对于Lenet-5的整体框架进行了介绍，我课后也找到了更加直观的能够反应结构的图：



可以看出输入层是一个32\*32的图片，然后再第一个卷积层中使用6个5\*5的卷积核，就可以得到6张28\*28的特征图（c1层）。

然后c1进行2\*2的非重叠池化，则得到了6张14\*14的特征图（s2层）。

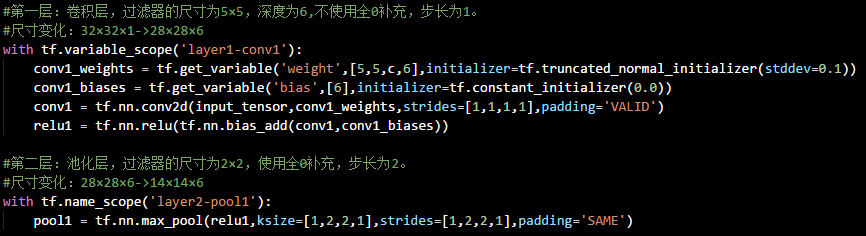
需要注意的是c3层，这也是一个卷积层，使用的是16个5\*5的卷积核，但是并不是每个卷积核都对6张特征图进行了卷积，然后求和，而是每个卷积核按照规律只选取6张图中的特定的几张进行卷积然后求和，这样处理的作用是，首先不完全的连接机制将连接的数量保持在合理的范围内（防止权重参数过多），第二则是破坏了网络的对称性，由于不同的特征图有不同的输入，所以迫使他们抽取不同的特征（希望是互补的）。

而s4层和之前的s2层一样，也是进行了2\*2的非重叠池化，得到了16张5\*5的特征图。

c5层则使用了120个5\*5的卷积核来进行全连接得到了120个神经元，即每个卷积核对16张特征图分别作卷积，则每个卷积核卷积完毕后能够得到16个数，将这16个数加起来就能得到一个神经元，一共有120个卷积核，所以最后得到120个神经元。这里的全连接是相对c3中的部分连接而言的，后者只选取特定的图片进行卷积然后相加，而后者选取全部的图片卷积后相加。

f6层则以84个神经元对c5层中的120个神经元进行全连接，最后的输出层则是由欧式径向基函数神经元组成，10个神经元（0-9），对f6进行全连接，然后使用soft max分类器，得到结果。

我在网上找到了Letnet-5模型的mnist数字识别的TensorFlow代码，并进行了学习，这里以第一层和第二层为例，可以看见代码与上面的描述是相符合的，最终使用csdn上提供的mnist图片包进行测试，准确度大概有99%：



对于Lenet-5的学习让我对于卷积神经网络有了一个更加具象的了解，也引发了我对于神经网络的一些思考，虽然卷积神经网络已经大大减少了权重参数的个数，但是这么多层的计算依然存在较大的复杂度，如何降低降低算法的复杂度（在不降低结果准确度的前提下）也成为了人们需要不懈努力的方向。

1. **制作小车过程学习心得**

我们组所制作的小车的功能是：小车首先自由直行，在途中遇到不同的“动物”（这里的动物指我们打印出来的动物彩图）并作出不同的反应，比如遇到老虎就后退，遇到海鸥就停止，并在小车行进过程中实时绘制小车的行驶路线，并对在行驶路线上遇到的动物实时作出标记。我们最终达到了三个功能：小车识别动物并作出不同的应急反应、实时观测小车行驶路线、绘制“动物园地图”。

我们的前期工作主要是按照老师的要求，对小车进行了组装，解决了一些硬件上的问题，实现了蓝牙控制小车和自动避障两个功能，但是这个时期的小车一日安存在着一些硬件上的问题，第一个是小车不能够倒退行驶，第二个小车过小，没有安放手机的位置，所以我们购买了新的零件，重新组装了一辆能够倒退行驶、安装了电源开关，并且能够安放手机的更大更稳的小车。

我们的后期工作主要分为三个部分：动物类别的识别、小车行进路线的实时绘制、电脑连接校车的蓝牙串口模块。

其中动物类别的识别我们主要使用了卷积神经网络（CNN）。这部分内容是由大家共同完成的，我们一共收集了8种动物，每个动物100张图片进行了标记，然后进行了灰度化处理之后使用最简单的CNN进行训练，最终模型能够对一些拍摄比较清晰，动物特征比较明显的照片进行较准确地识别，区分出八种不同的动物。具体实现过程是，我们安装在小车上的手机能够对道路进行拍摄，并通过软件“WiFi手机摄像头”软件传送给电脑，实时显示在电脑上一个网址为手机IP地址的网页上，我们利用Python代码循环读取这个网页上的图片，每读取一次图片便进行一次识别。

而小车行进路线的实时绘制则是使用了Python中的Turtle模块，按照我们设计的规则对路线进行绘制，如果电脑上识别到一张老虎的照片，便作一个标记“虎”，并控制Turtle像小车一样后退一段距离，如果识别到一张海鸥的照片便控制Turtle像小车一样停止一段时间，这样就能够使小车行驶路线与绘制出的路径一致。

电脑连接小车的蓝牙串口模块本来是一个比较简单的操作，但是由于硬件原因，我们依然进行了较长时间的调试，多次调整了COM口才得以成功连接。识别到一个动物便给小车蓝牙串口发送一个指令，引导小车的行驶，在这个过程中我们还发现一个问题，每次连上蓝牙后，都会自动给小车蓝牙发送一段数据“<<32632”，导致小车会有一段控制混乱的阶段，所以我将数字控制改为了字母控制，例如用“b”表示backward，成功解决了这个问题。

因为我们需要一边进行实时绘制小车行驶路线，一边给小车蓝牙串口发送信息来控制小车行驶，所以我们对代码进行合并时，开了一个线程来对小车行驶路线进行绘制，这样就能达到同步的效果，在开设线程前，我们只能做到给小车蓝牙发送一条消息，绘制一段地图的效果，看起来很不流畅也不够实时，修改后二者并行执行效果变好了很多，既流畅又实时。

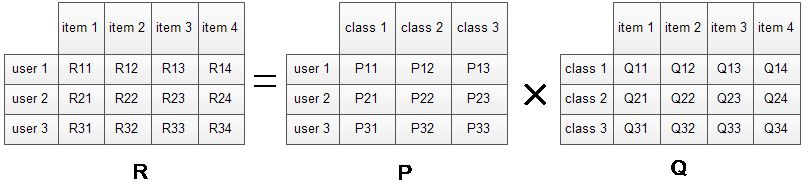
通过这一个学期以来的制作，我有了许多收获，从螺帽拧不下去、马达方向装反了这样的硬件问题到蓝牙连接失败、图像识别准确度低这样的软件问题，我既提升了自己的动手能力也学到了许多新的知识，我们组员之间也一直都团结合作，一起解决问题，共同提升小车性能，这也提升了大家的协作能力。

1. **课外拓展——机器学习在推荐系统中的应用**

在网络高速发展的今天，社交网络、视频网站、音乐播放软件等系统中的推荐模块也在顺应时代要求地高速发展，我在对于矩阵分拣推荐算法进行学习时发现，原来人工智能技术近年来也在推荐系统中有很多的应用。

在最早的推荐系统中，一般是使用一些简单的数学模型，如基于内容的推荐算法、协同过滤推荐算法，前者比较比物品之间的相似性，假设用户1喜欢物品A，而物品A与物品B相似，则把B推荐给用户1，后者往往是比较用户的相似性，如果用户1和用户2喜欢的物品列表相似，则把当前只有用户2喜欢的物品A推荐给用户1。但是这样的两种方法都是非人工智能的，推荐效果是基于长期的经验获取，没有办对突发事件、时间、新用户、新物品进行综合考虑。

而我学习的矩阵分解推荐算法则是基于人工智能子领域的机器学习。它的基本思路是假设用户对于某一物品的评分是受到若干隐因子的影响，将用户和物品映射到一个共同的隐因子空间。人们在购物时往往也是受到很多因素的影响，例如价格、品牌、款式等等，在本模型中这些隐因子是很难解释出其究竟是哪个具体的因素的，但是我们却又能够在模型中发挥这些隐因子的作用，一定程度上也体现了模型的智能性。矩阵分解就是将用户-物品评分矩阵分解成为两个矩阵，如下图所示，P矩阵中的user-class，user显然为用户，而class则表示之前提到的隐因子。



我学习了最基本的几种矩阵分解算法，在Basic MF（基本矩阵分解算法）中，我们把用户-项目评分矩阵R分解为两个矩阵W和H乘积的形式：



称计算出来的用户-项目评分矩阵R的近似值为计算矩阵，称WH为用户-项目交互矩阵，在这里用户-项目交互矩阵WH即为计算矩阵。为了能够实现R和WH的近似程度达到最大，需要计算R与计算矩阵WH之间的差异，记做D(R||WH)，用欧几里得距离来表示R与计算矩阵WH的不同程度，我们就得到了机器学习中的损失函数，也就是我们的优化目标，即：



然后将使用梯度下降法来得到W和H矩阵中的每一个元素（其实就是参数），而对于这些参数的调优其实就是机器学习思想的体现。由于参数数目很多，所以容易产生过拟合问题，所以还有优化的Regularized MF模型为损失函数加入了规范化因子，以防止过拟合问题。

当然矩阵分解推荐算法依然有很多没有解决的问题，也仅仅是停留在机器学习的阶段，与真正的人工智能仍有一段距离，它忽略了人机交互的系统性与时间性，也很难挖掘出用户对于未知领域的新兴趣。目前人们所探索的方向就是不再把推荐系统看作是一次一次推荐结果运行的片段，而是将整个人机交互过程视作一个整体，从而在整个时间维度上来对推荐系统进行动态优化，这将与人工智能更加贴近，人们也开始尝试将深度强化学习运用到推荐系统上来，不过在推荐系统上实现真正的人工智能依然任重道远。

1. **总结**

经过一个学期的学习，我对于人工智能有了比较系统的学习，也在做实验的过程中提升了自己的动手能力，使得人工智能对我而言不再是一个高端而陌生的领域，使得我自己也能够使用一些简单的人工智能的模型，不过这些了解依然比较浅显，如果要能更好地掌握更多的模型与技术，依然需要付出更多的努力。