人工智能课程报告

2015202009 孟妍廷

人工智能是一门用于模拟、延伸和扩展人的智能的理论、方法、技术及应用系统的一门新的技术科学。近几年来，随着人工智能的技术越来越成熟，人们愈加广泛地在图像识别、自然语言处理和机器人等方面应用人工智能。可以说，对于人工智能的研究是当今科技进步的一个重要组成部分。本学期，我们学习了《人工智能》这门专业选修课，接下来我将分基础知识和设计实践两个方面对本学期的收获进行总结：

一、基础知识

本学期我们主要教材是《Hands-On-Machine Learning》这本书，学习下来觉得机器学习的思维方式与我们之前学习的传统计算机课程是有很大不同的，需要大量的实践和阅读文档才能够掌握。通过老师的讲解和阅读教科书，我主要了解了以下几个知识点：

（1）tensorflow：

tensorflow是一个采用数据流图，用于数值计算的开源软件库。其特点是具有高度的灵活性和真正的可移植性，使得它能在多种平台上展开计算，适用于机器学习、深度神经网络以及其他计算领域的研究。

在老师的指导下，我了解到tensorflow可以与jupyter结合，使用命令docker run -it -v ~/Desktop/python（自己命名）:/notebooks/python tensorflow/tensorflow /bin/bash将本地的代码上传到交互式笔记本上运行。可以说，利用tensorflow和python，大大地降低了学习人工智能的代价，使得入门变得简单。

（2）SVM（支持向量机）：

SVM是一种判别方法，可以用来机器学习的模式识别、分类和回归分析。它的主要逻辑是给定训练样本，建立一个超平面作为决策曲面，使得正例和反例的隔离边界最大化。有的样本在低维空间就是线性可分的，但是在更多情况下，样本在自变量和因变量的关系捕食现象的，此时可以通过高斯kernel函数将其映射到高维空间转化为被曲面隔开的线性可分的。

而SVM回归的核函数是一种特殊的损失-惩罚问题，特点是损失函数落在margin正确的一侧，只对margin以内的样本做线性惩罚。因此，模型由误差较大的点决定。

通过查阅资料，我了解到如果把SVM回归的思想推广到连续变量的情形，构造一个损失函数：对于小于e的损失不作惩罚，对于大于e的损失作线性惩罚，就得到了SVR。

（3）RNN（循环神经网络）

在传统的神经网络模型中，是从输入层到隐含层再到输出层。其中层与层之间是全连接的，每层之间的结点是无连接的。这仅适用于“上下文无关文法”，对于需要用到前面的单词的“上下文有关文法”是无能为力的。因此传统的神经网络是不能实现单词预测的。而RNN的进步之处就在于一个序列的当前输出与前面的输出有关，网络会对前面的信息进行记忆并应用于当前输出的计算中，各结点之间存在连接。在实践中，虽然不能实现对任意长度的序列进行处理，但是可以假设当前状态与之前的一定数量的状态有关。

同时RNN的一个非常有用的特征是每一个网络层的参数都是共享的，大大降低了网络中需要学习的参数。

基于以上特征，RNN非常适用于语言模型与文本生成、机器翻译、语音识别和图像描述生成。

（4）学习总结：

在学习人工智能的过程中，我觉得知识点大多比较抽象，理解起来比较困难。所以，我对经过实践的知识点的理解较为深刻，因为切身的实践使得抽象变为具象。以下的设计实践是我在本学期的人工智能学习中的实践项目，我希望今后经过训练我能够写出更实用更复杂的应用。

二、设计实践

1.ai-car设计（详见展示报告）

本学期我们的大作业是利用分组设计一辆人工智能小车，主要实现思想和结果如下：

（1）基础功能：

首先我们按照老师的ppt，拼装了小车的基础框架，利用老师的代码实现了蓝牙控制小车左转、右转、后退和停止，以及遇到障碍物自动右转。

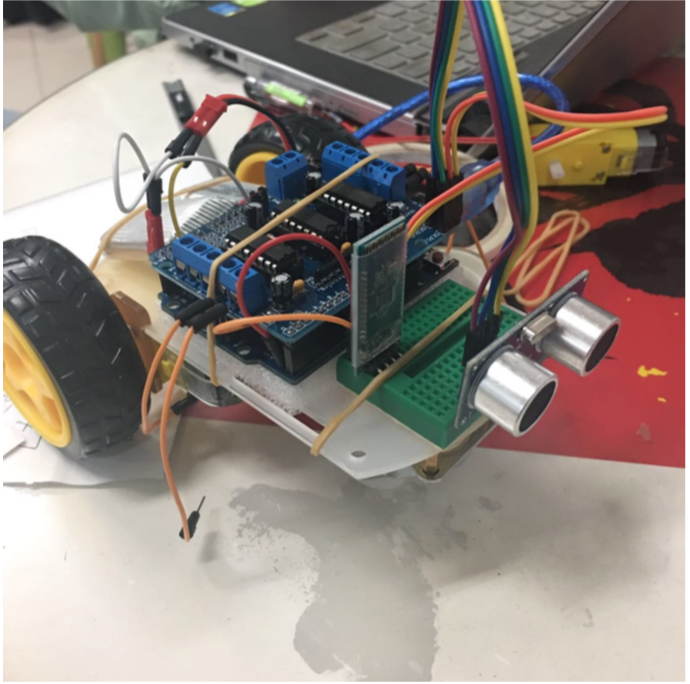


图1

（2）特色功能：

在实现基础的自动避障之后，我们希望小车在遇到障碍物的时候不只是自动右转，而是实现更高级别的避障功能——根据当前障碍物的特点决定是左转、右转还是后退。因此，首先我们在小车上固定手机并打开摄像头，利用DroidCam通过WiFi将手机拍摄的视频传到电脑上。

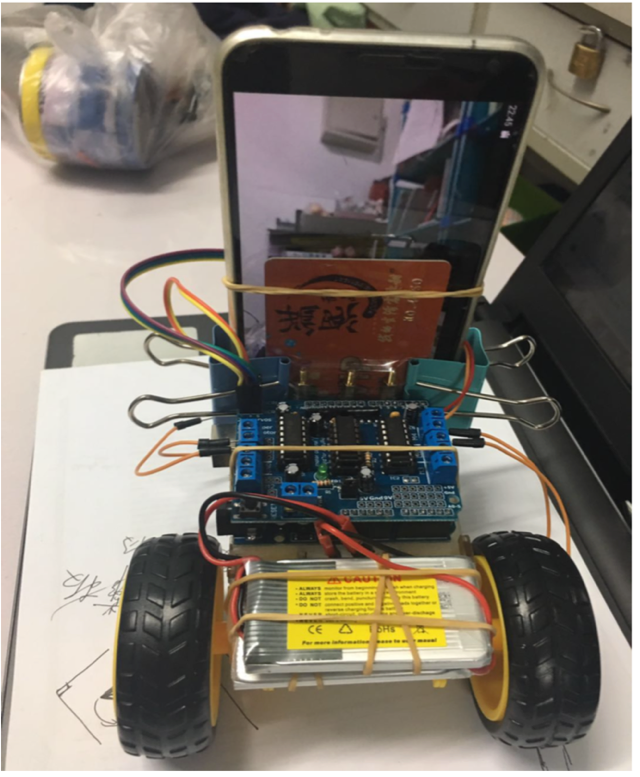


图2

我们在电脑端安装了python的opencv包，利用opencv每隔两秒截取一次视频得到不同类型的障碍物的图片。经过筛选之后得到1787张有效图片。

接下来对图片进行灰度处理，使用高斯平滑处理图片进行降噪，再使用Canny边检测器检测障碍物的边缘。对障碍物的边缘使用尺度不变特征变换（SIFT）提取特征点，得到特征向量。紧接着用词袋模型对向量进行聚类并且利用k-means算法提取所有图片的SIFT特征，并将每一维特征降维到一维。

得到训练集后，最后对当前的图片使用SVM进行分类，决定转弯方向，利用蓝牙将命令返回给小车。

训练结果：

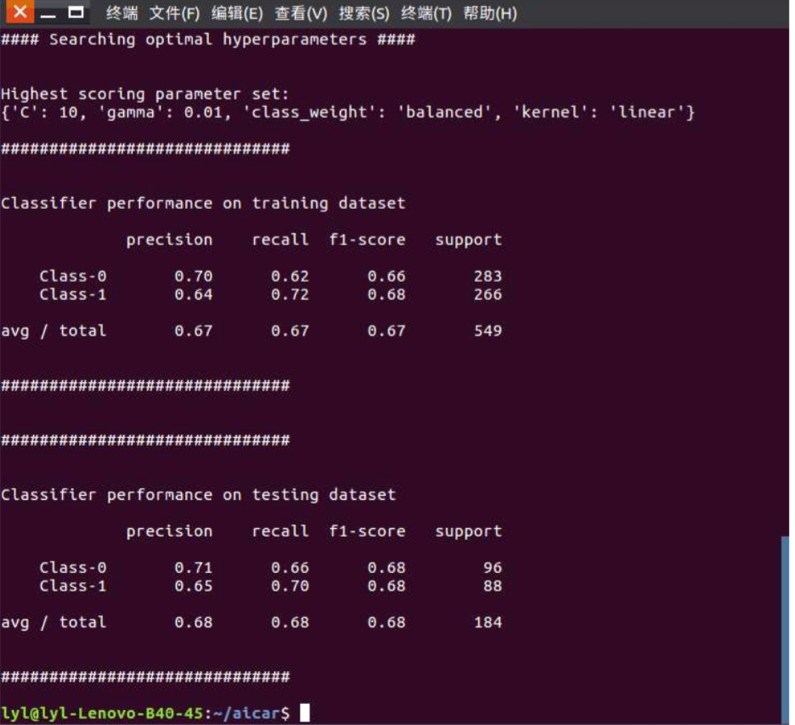


图3

SVM结果集：

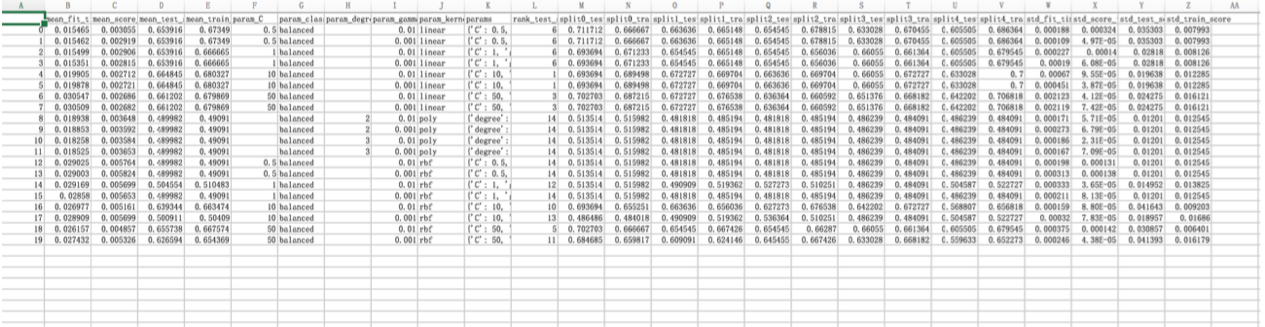


图4

2.写诗机器人实践

为了更好的理解和应用tensorflow和循环神经网络(rnn)，我在GitHub上找了一些利用rnn的ai项目，这些项目中最成熟最典型的是训练ai写诗的应用。我找到的项目中大多实现了自由诗押韵、藏头诗，以若干字为主题作诗等等功能。由于作为初学者水平有限，经过对这些项目的学习和模仿，我在课余时间写了一个简单的能做藏头诗的“写诗机器人”。

实现过程：

（1）训练集及格式：

4万首带标题的古诗的txt文档，每首诗占一行。正文和标题用“:”隔开，诗句以“，”和“。”结尾。

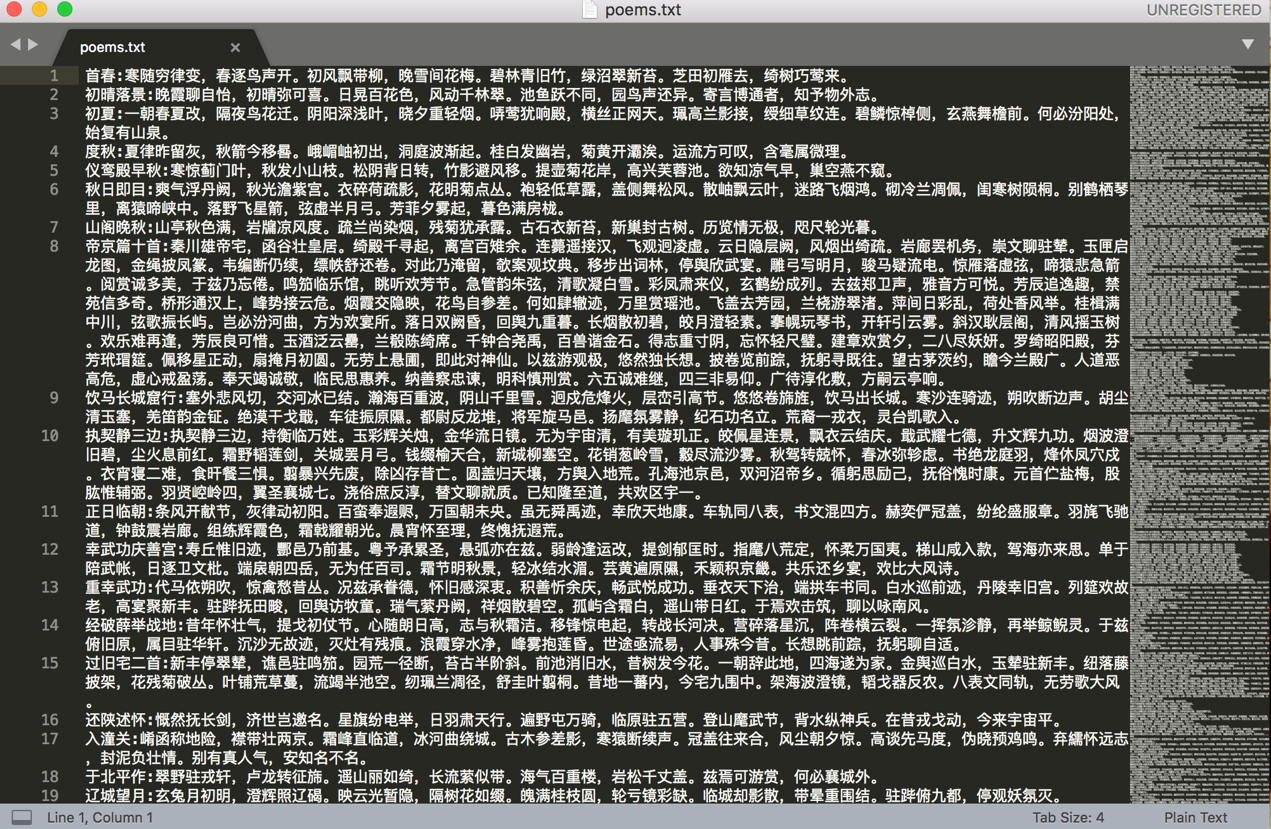


图5

（2）处理训练集为训练做准备：

首先将训练集的txt文本每一行读入并清洗：统一tab和空格，并且把训练集中含有（）、\_、《》、[]等影响训练结果的非法字符的行无视，而合法的行写入列表poem中。接下来对列表poem按照每首诗的字数排序，统计所有诗中每个字出现的频率。最后根据频率由高到低选出常用字，将每个常用字映射为一个数字ID存入列表。

（3）训练模型：

每次将64首诗作为一批进行训练，找到这64首诗中最长的一首的长度，以此长度填充两个空的batch，记为x和y，并将64首诗中的字对应的ID还原进x，x中每一个子列表向左移一位得到y（eg: x:[1,2,3,4]->y:[2,3,4,4]）。

为了延长记忆时间减少误差，定义lstm cell，利用MultiRNNCell进行多层堆叠（cell = tf.contrib.rnn.MultiRNNCell([scell() for \_ in range(num\_layers)], state\_is\_tuple=True)）。

为了刻画传入graph中的词与词之间的关系，使用embedding矩阵来表示input中的每一个字（inputs = tf.nn.embedding\_lookup(embedding, input\_data)）。得到input之后利用dynamic\_rnn得到outputs和last\_state。

接下来将outputs传入softmax层，得到logits之后计算loss。

将训练的过程中的每一个end\_point存储下来，在中断后保证下一次可以从当前记忆点开始重新开始。

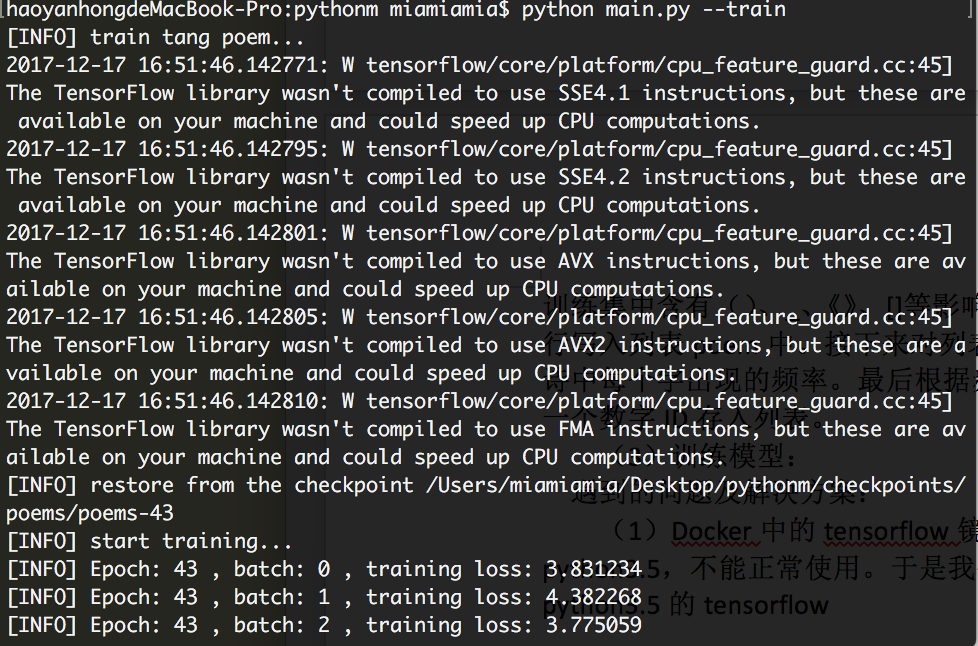


图6

成果（图片照于10月19日）：

以“孟”开头的藏头诗：

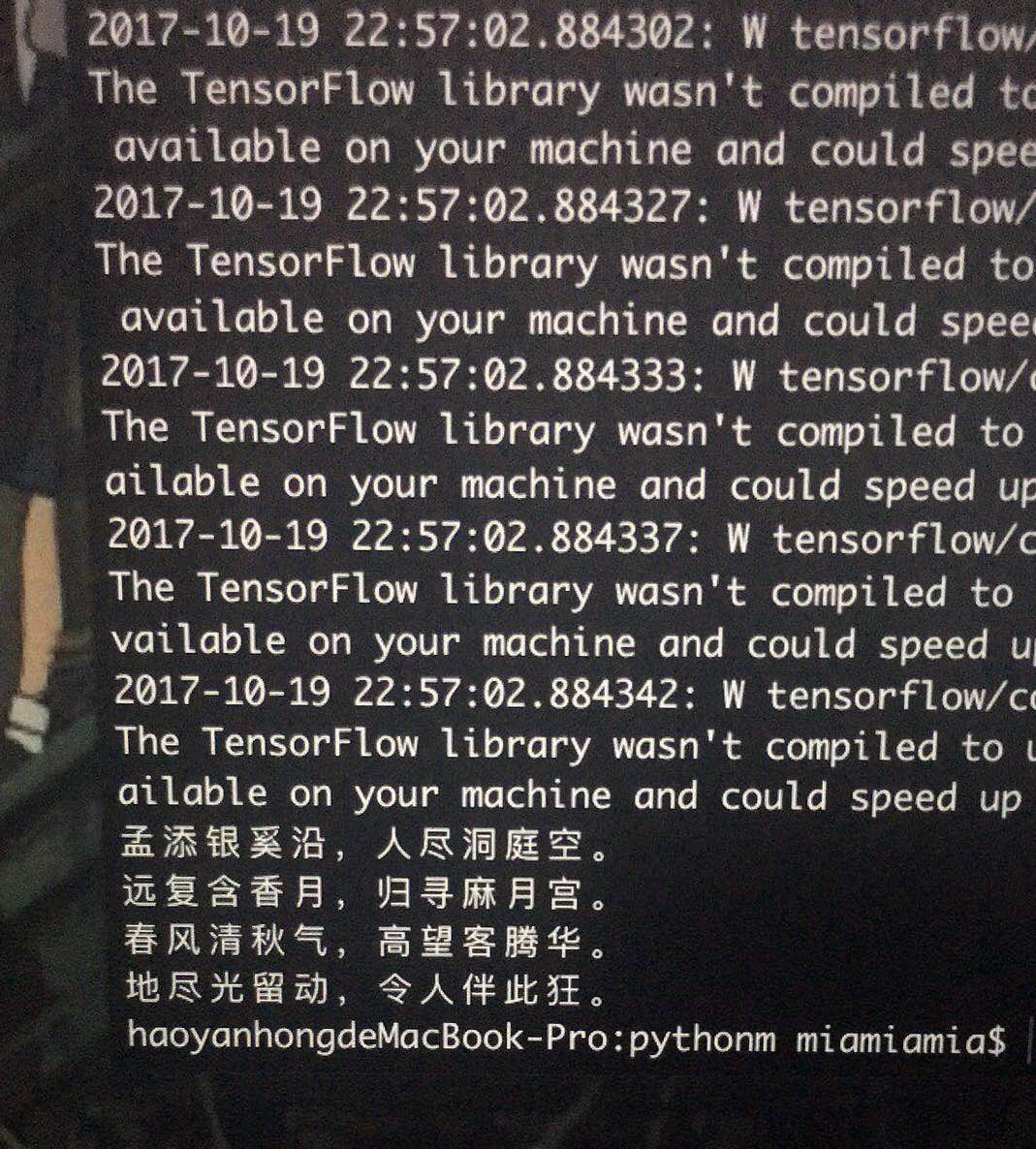


图7

以“杨”开头的藏头诗：



图8

遇到的问题：

（1）Docker中的tensorflow镜像只支持python2.7，而我安装的是python3.5，不能正常使用。于是我使用anaconda重新在本地安装了支持python3.5的tensorflow

（2）我参考的项目使用的tensorflow的api是1.4的，我的是1.0，MultiRNNCell中cell的循环堆叠格式不同，造成了很大的麻烦。后来通过查文档解决了问题

（3）设计模型的过程对于我来说难度较大，所以主要是通过读和模仿高手的代码进行理解来实现的

参考资料：

1. <https://github.com/norybaby/poet>

2. <https://github.com/jinfagang/tensorflow_poems>