

《人工智能导论》学习报告

学号：2016202137 姓名：龙彦兵

第一次关注人工智能，是在16年AlphaGo大败李世石的五番棋上，那时的确很好奇，人工智能到底是怎么做到的，有点无法想象。确实是缘分，那时还是小白的我两年后就正式接触了人工智能。在上这次课前对人工智能还是比较模糊的，按照老师的推荐，选择了《Hands-On Machine Learning with Scikit-Learn & Tensorflow》这本比较经典的书作为入门教材。不得不说，全英文的书读起来还是比较吃力，而且老师上课讲得飞快，适应起来还是比较难的。课程初期，全是概念和算法理论。于是自己在网上找了个十分简单的用TensorFlow来预测销售额的项目来练练手，用的是最简单的 $y = a_0 + a_1x$ 单变量线性回归模型。理论早就学过，但是用代码搭建神经网络还是第一次，从copy到理解，tensor、operation、shape以及计算准确率等概念一个一个的概念的吸收，进度虽然缓慢，但对增加学习信心十分有帮助，可以说我的coding操作就是从这正式开始的。不久后上课讲到了比较经典的MNIST数据集，早就听说过这是机器学习在视觉领域的hello world，很想去练练。每张图片上的是手写数字0-9，对应标签即为相应的数字。图片格式统一为 $28 * 28$ ，使得后续操作简化了不少。从像素级别来处理图片，讲其化为有784变量的向量来处理。按照书上步骤用matplotlib把图片可视化，初步尝试了matplotlib这个以后要常常打交道的库的使用。之后使用sklearn进行处理，书上讲了很多，但其实并不难。很多理论性的东西讲得比较简单，手过一遍代码后慢慢就可以理解。



以上两个项目可以说是我的入门指导项目了，这次课程最主要的当然是这个智能小车。一开始觉得既然是车，就凭感觉说服小组选了智能驾驶这个方向，后来实际操作后发现这真的是一个很有趣的项目。我们把项目分为若干部分，我主要负责对交通标志内容的识别。不得不说对数据集的处理也是挺麻烦的，虽然是直接从网上下载的，但对numpy、

pandas的不熟悉，而且是全英文的说明，也折腾了一番。其实整个流程基于TensorFlow很简单，难办的是一些细节上的处理，处理不同的事物经常要去选择和学习相应的库。图片处理使用了scikit-image，它对图片处理支持很好，除格式外，对图片的大小处理也十分容易上手。毕竟实验中的交通标志是从手机传来的每一帧图像中截出来的，大小不一，而且角度也不尽相同，使用scikit-image将其“矫正”，统一化为32*32格式。



而搭建Graph就比较简单了，如下

```
#Graph to hold the model.
graph = tf.Graph()

# Model in the graph.
with graph.as_default():
    # Placeholders for inputs and labels.
    images_ph = tf.placeholder(tf.float32, [None, 32, 32, 3])
    labels_ph = tf.placeholder(tf.int32, [None])

    # Flatten input from: [None, height, width, channels]
    # To: [None, height * width * channels] = [None, 3072]
    images_flat = tf.contrib.layers.Flatten(images_ph)

    # Fully connected layer.
    # Generates logits of size.
    logits = tf.contrib.layers.fully_connected(images_flat, 62, tf.nn.relu)

    # Convert logits to label indexes (int).
    predicted_labels = tf.argmax(logits, 1)

    # Define the loss function.
    # Cross-entropy is a good choice for classification.
    loss = tf.reduce_mean(tf.nn.sparse_softmax_cross_entropy_with_logits(logits=logits, label

# Training op.
train = tf.train.AdamOptimizer(learning_rate=0.001).minimize(loss)

# Initialization op to execute before training.
init = tf.global_variables_initializer()

print("images_flat: ", images_flat)
print("logits: ", logits)
print("loss: ", loss)
print("predicted_labels: ", predicted_labels)
```

最终决定使用全连接和Adam优化器，在可尝试的方法中效果最好了（K-折交叉验证方法效果也很好，不过有时候不稳定）。但是准确率仍有不尽人意，到了接近1.6的时候已经是极限，准确率不到7成。

```
Loss: 4.2588
Loss: 2.88972
Loss: 2.42234
Loss: 2.20074
Loss: 2.06985
Loss: 1.98126
Loss: 1.91674
Loss: 1.86652
Loss: 1.82595
Loss: 1.79212
Loss: 1.76336
Loss: 1.73851
Loss: 1.71678
Loss: 1.69759
Loss: 1.68047
Loss: 1.6651
Loss: 1.65121
Loss: 1.63858
Loss: 1.62705
```

这基本就是本次我主要负责的部分了。在Github上也有许多类似的项目，有的准确率甚至能达到98%，把项目clone下来阅读了代码，很多用了很多较高深的算法，而且调用了image等不常见的库，整个代码运行下来问题都很多。整个项目是基于Yann Le Cun的《Traffic Sign Recognition with Multi-Scale Convolutional Networks》这篇论文，内容涉及的多层网络及优化方法晦涩难懂，时间有限，难以消化，不得不降低目标。虽然本次实验离原来设定目标仍有距离，但是过程中发现了许多别人做的自动驾驶项目，十分吸引人，之后我会继续在这个方向学习。我想这正是这门课程最大的收获了吧，为我打开了新世界的大门。



