**优达学城机器学习纳米学位毕业项目：猫狗大战**

徐早立

1. **项目定义**

给一张猫或狗的图片，能够识别出图片中的动物是猫还是狗。对于人类来说这近乎于本能，可对于人工智能来说，这已经算得上是一种挑战了。在很长一段时间里，人工智能对于人类来说就像是海市蜃楼，真正的人工智能仍在大洋彼岸，可望而不可及。直到深度学习这座桥梁的出现，人类通向人工智能才出现了一丝曙光。

如果把人工智能看做一个在“彼岸”的目标，客观地说，通往人工智能的方式有很多，可以“游泳”，也可以“坐船”，甚至可以“坐飞机”。可是到目前为止，这些方式都还具有相当的难度，短时间内看不到太大的希望。现在最有希望的方法是从桥上走过去，而所有的这些桥梁统称为机器学习，其中现在最好的一种桥叫做深度学习。所有建造机器学习“桥梁”的原材料就是数据。

所以，站在人工智能的角度，需要先有一个具体的人工智能的目标，再去围绕这一目标搭建一座合适的机器学习桥梁，最后再去寻找合适的造桥材料——数据。

本项目的目标是希望实现一个具有猫狗图片识别功能的人工智能。基于上述的分析，实现这一目标主要有两步：数据的搜集和机器学习算法的选定，而对本项目来说，重点在后者。因此本文将着重讲述如果选定和实现机器学习算法。

本项目也同时对应数据网站Kaggle的猫狗识别项目（https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition）。

评价一个人工智能的视觉识别能力需要一个量化的标准。在Kaggle上这一项目的评估标准采用的是loss值，定义为：

其中*y*表示实际图片中是猫还是狗。如果实际图片中是猫，则定义*y*=0；如果实际图片中是狗，则*y*=1。而*ŷ*表示所构建的人工智能判定一张图片中是狗的概率。*m*为测试图片总数。loss值越低，意味着视觉识别能力越强。

本项目的目标设定为loss值小于0.05629，这一目标对应于进入Kaggle公开排行榜前100名。

**2. 项目分析**

* 1. **数据积累**

前面提到，深度学习使得人工智能见多识广。换句话说，本项目的人工智能需要看过大量的猫狗图片，才能学到识别猫狗的本领。因此，本项目需要大量的猫狗图片，这些图片来自Kaggle（https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data）。数据分为训练集和测试集。训练集用来训练人工智能，而测试集则用来评判这一人工智能的识别能力。

数据通过下载的方式全部下载并保存到本地。

* 1. **探索性数据分析**

训练集共有25000张图片，猫和狗的照片各有12500张，如图1所示。每张图片的名字中含有该图片的分类与序号信息。比如，命名为“cat.0.jpg”意味着这张图片中是猫，图片编号是0。测试集包含12500张猫或狗的照片。测试集的图片只有序号，没有图片的分类信息。训练集的25000张照片会被用来训练具有猫狗辨识能力的人工智能，而测试集的图片会用来评判这一人工智能的猫狗辨识能力。

训练集中猫和狗的图片示例如图2所示（为方便展示，图片进行了缩放调整）。初步来看，训练集的数据特征有：

1. 猫狗照片数量均等，没有类别不平衡的问题；
2. 图片的尺寸不一致。如图2(a)中猫的照片尺寸为417×299，而图2(b)中狗的图片尺寸为251×359。
3. 有的图片中包含多只猫或狗，如图2(c)所示。
4. 数据集中含有一些疑似无关的图片，如图2(d)所示。

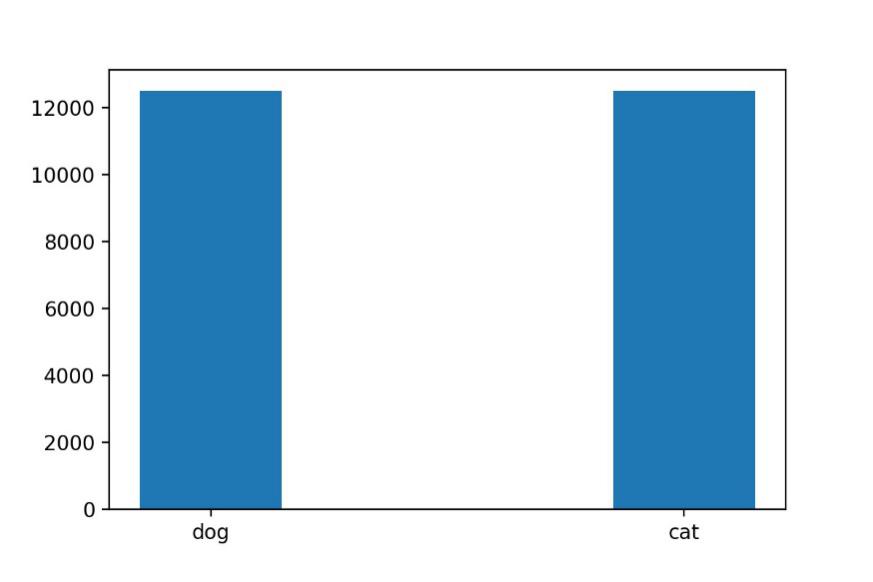


图1 训练集中猫/狗照片数量柱状图



**(a)**

**(b)**

**(c)**

**(d)**

图2 训练集图片示例。 (a) 猫的图片示例； (b) 狗的图片示例；(c) 两只猫的图片；(d) 疑似标记错误的图片示例

* 1. **数据预处理 （Data preprocessing）**

对于训练集的图片，所做的数据预处理包括：

1. 划分训练验证集。本项目采用“留出法”将训练集划分为互斥的训练/验证集，其训练/验证划分比例为0.96/0.04。验证集有1000张图片，如果两个不同的模型的识别准确率相差0.1%，那么在验证集上可以体现出来（0.1%×1000=1张图片）。
2. 调整图片大小。本项目的图片尺寸各一，需要统一图片的尺寸后，才能输入到卷积神经网络进行建模。本项目最终所选用的尺寸大小为128×128，这是综合建模结果和计算机运算能力的折中选择。若是尺寸选择过小，则图片清晰度不够，难以达到满意的识别精确度；若尺寸选择过大，则难以处理过大的数据量，运算效率降低。统一大小后的图片示例如图3所示。可以看到调整大小后的图片有两个特征：第一，猫和狗的形态比例相较于原始图片发生了变形；第二，图片的颜色也发生了变化。
3. 归一化。每张图片都是由像素点阵列存储在计算机中，而每一个阵列点是由0~255范围内的数字表示。为加快模型优化速度，采用常规的归一化处理，把所有的像素值用min-max归一化成0~1范围内的数字。
4. 独热编码。这一项目有猫狗两类， [0,1]表示猫，[1, 0]表示狗。
5. 保存预处理数据。方便直接导入处理过的数据，而无须每次调试模型时再预处理一遍数据。



图3 统一大小后的图片

对于测试集的图片，所做的数据预处理包括：

1. 调整图片大小，保持和训练集一致。
2. 归一化，保持和训练集一致。
3. 保存预处理数据。

在做数据预处理的过程中，遇到了计算机内存不足的问题。本项目所用的电脑内存为32GB，若是调整图片大小的时候选择64×64，则可以一次性处理完全部12500张训练图片并保存。若是调整图片大小的时候选择128×128或者224×224，则会报出内存不足的问题。经过分析，造成这一问题的原因在于保存预处理数据时使用的pickle库会使用大量的内存[4]。若是图片尺寸过大，则无法一次性保存所有预处理的数据。

解决这一问题可以选择使用HDF5 [5]或者TensorFlow的TFRecords [6]来代替pickle，减少内存的消耗。而本项目采取的是另一种方法，对数据集进行分批处理并保存。这一方法与图像分类项目中的做法一致 [7]。简略来讲，为解决数据量过大的问题，将数据集的图片分成几批，每次计算机只处理一批的数据。据试验，若一批中的图片为4096时，即使图片大小为128×128或者224×224都能完成预处理并保存在本地的磁盘中。

* 1. **数据分析-建模与优化（Data analysis）**
     1. **迁移学习**

**2.4.1 模型架构**

本项目通过迭代优化的方式来建立卷积神经网络。最初的网络参照图像分类项目的网络结构，随后参照AlexNet [8]增加了网络了层数，最后再参考了VCC-16 [9]完成了卷积神经网络的搭建。参考[9] 的标记方式，本项目最终的模型架构如图4所示。

输入

（128×128 RGB图片）

conv3-64

maxpool

conv3-128

maxpool

conv3-256

maxpool

conv3-512

maxpool

FC-4096

FC-4096

softmax

conv3-512

maxpool

Output-2

图4 本项目卷积网络架构

conv3-64的含义是：3代表3×3的卷积滤波器，64代表这一卷积层的输出深度为64。stride值为1，采用的padding方式为 ’same’。其他卷积层的含义以此类推。

maxpool表示最大池化层，滤波器大小为2×2，stride为2, padding方式为 ’valid’。

FC-4096为全连接层，输出大小为4096。

Output-2为输出层，输出大小为2。与全连接层相比，输出层没有激活函数，直接输出。

softmax把输出归一化到和为1。

另外，卷积层、池化层和全连接层都使用Relu作为激活函数。

在每一个最大池化层和全连接层后面都加入dropout，来防止过拟合。

**2.4.2 模型调参**

总体来说模型调参是一个试错的过程，大致过程是：从一些经典模型或者常用的设置出发，在结合实际项目的数据情况进行试错调整。

keep\_prob: 任何一个给定单元的留存率。keep\_prob可以调整丢弃单元的数量，从而缓解模型过拟合的程度。keep\_prob最终选为0.6。

stddev: 卷积层、全连接层和输出层的权重值在随机初始化时的标准方差。权重的初始化是一个影响结果的重要因素，合理的权重初始化应该是既能使得模型能够摆脱全对称的状态（梯度消失使得权重值无法得到更新），又不能偏离全对称态太大（各个权重值相差太大导致梯度爆炸）。本项目中stddev选择的大小为0.05。

batch size: 本项目采用mini-batching和随机梯度下降来进行优化，batch size的选择主要是考虑内存大小，在计算允许的前提下尽量增大batch size。本项目使用的batch size值为128。

epoch: 模型的训练次数，本项目设定的最大epoch为100次。

**2.4.3 模型评估**

本项目在建模以及调参过程中，通过监控训练集和验证集的准确率和loss值来评估模型的好坏。具体的讲，在每一代优化结束后，计算、打印并储存训练集和验证集的准确率和loss值，以此来判断当前模型是欠拟合，还是过拟合。

本项目所建立的卷积神经网络在验证集上的loss值为~0.18。将测试集的数据导入模型，计算出结果并导入kaggle网站，得到的测试集上的loss值为0.22924。

* 1. **数据可视化 (Data visualization)**

**当前略**

**结果与讨论**

当前的loss值还并不能令人满意，没有达到之前设定的小于0.05629这一目标。希望能够得到一些反馈完善后再来完成这一部分。

**结论**

**当前略**

**参考文献**

[1] http://mp.weixin.qq.com/s/-wSYLu-XvOrsST8\_KEUa-Q, 浅谈人工智能：现状、任务、构架与统一——正本清源，2017，朱松纯，视觉求索

[2] https://en.wikipedia.org/wiki/AlphaGo

[3] https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html, 2016, Adit Deshpande, The 9 Deep Learning Papers You Need To Know About (Understanding CNNs Part 3)

[4] https://shocksolution.com/2010/01/10/storing-large-numpy-arrays-on-disk-python-pickle-vs-hdf5adsf/, 2010, Craig Finch, Storing large Numpy arrays on disk: Python Pickle vs. HDF5

[5] http://machinelearninguru.com/deep\_learning/data\_preparation/hdf5/hdf5.html#list

[6] http://machinelearninguru.com/deep\_learning/data\_preparation/tfrecord/tfrecord.html

[7] https://github.com/udacity/cn-deep-learning/blob/master/image-classification/dlnd\_image\_classification.ipynb

[8] https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf, AlexNet

[9] https://arxiv.org/pdf/1409.1556.pdf, VCC-16