机器学习工程师纳米学位毕业项目

基于深度学习的猫狗图像识别（“猫狗大战”）

2018年4月26日

# 问题定义

## 项目概述

“猫狗大战”项目涉及的领域是计算机视觉，具体的说是“图像识别”问题。计算机视觉这个研究领域最早目的是为了研究如何让智能机器人像人类一样拥有高级视觉系统，能“看到和识别”外部世界中的各种物体。之所以选择这个项目是因为我本人对计算机视觉这个研究领域非常感兴趣，希望以后能将相关知识应用于智慧农业，特别是计算机自动识别常见的农业病虫害。

该项目是Kaggle上面的挑战项目之一，该项目自带了与项目有关的数据集，该数据集包含25000张图片组成的训练集和12500张图片组成的测试集。该项目将使用训练集图片训练一个基于深度学习的算法模型，然后在测试集图片上评估其准确率。

## 问题描述

这里将要解决的问题是训练一个算法模型来识别图片中是猫还是狗，是一个“二分类”问题。该问题对于人类而言并不是什么大问题，但对计算机而言却很困难，因为计算机“看到”的是由数字组成的点阵，它要分辨出这样的数字点阵是什么物体是很困难的。然而，像这样的图像识别问题已经有了很多效果不错的解决方案，深度学习就是其中之一。我在这里将结合深度学习中的卷积神经网络提出一个自己的解决方案。

这里准备采取的策略是“迁移学习+模型融合”，因为就卷积神经网络而言，迁移学习是一种很好的策略，特别是应用于图像识别领域，这是由卷积网络的分层架构决定的，网络中的前几层能识别图像中的一些简单的图案，比如边缘等等，这些往往是每个图像识别问题所共有的特征，没必要从头训练，可以复用并节约解决问题的时间，像ImageNet这样包含1000种分类的的超大规模图片数据集已经预训练了几种常见的卷积神经网络模型，可以直接使用。此外，对多个模型的输出结果进行融合，能够博采众长，兼听则明，有效提高预测的准确度，比只使用单个模型要好。最后得到的预测结果为一个概率值p，代表该图片是狗的概率。预期的结果是对于狗的照片，该值接近于1；而对于猫的照片，该值接近于0。

## 评价指标

本项目要得到的最终结果是一个二分类问题，所以这里将用准确率结合二元交叉熵损失函数作为算法性能好坏的评估指标，将根据训练集和测试集的损失函数表现来评估算法性能。如果验证集损失还在下降，那么需要增加模型复杂度或者多训练几代；如果验证集损失上升，则出现过拟合，需要正则化或Dropout防止过拟合；如果验证集的损失出现震荡，则需要减小学习率；如果验证集的损失趋于稳定，则可以减少训练代数。

* C：损失函数（Cost Function）；
* n：数据集数量；
* y：分类为狗（y=1）或猫（y=0）；
* p：模型预测分类为狗的概率。

# 2. 分析

## 2.1 数据的探索

训练集包括25000张图片，都为JPEG格式，是按照“猫/狗.编号”来命名的，比如“cat.0.jpg”。图片包含猫和狗的各种姿态的照片，排在前面的12500张图片全部是猫，剩下的全部是狗。通过对图片的大致浏览，发现某些图片还包含有人类，如图1所示。有的图片则同时出现了狗和猫，但被标记为猫，如图2所示。甚至还发现了异常值，如图3所示，一张人类的照片被标记为了猫，类似这样的异常值肯定会对分类准确性造成一定影响。



图1 人类抱着猫咪的图片



图2 同时出现猫和狗的图片

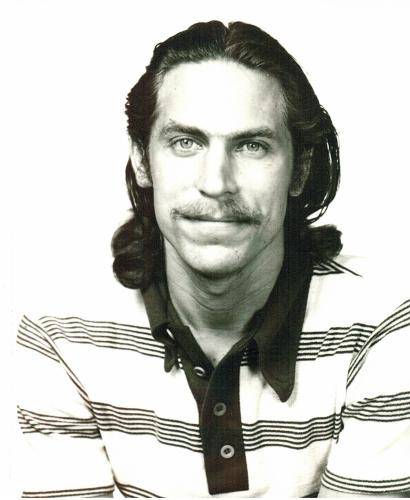


图3 一张人类的图片被标记为猫

另外，图片的尺寸不一致，在输入到神经网络之前，应该调整图片的大小。具体地，需要按照神经网络输入层的要求对图片进行resize操作。训练集数据需要进一步分为训练集和验证集并打乱顺序，这可以提高模型在测试集上的泛化能力。测试集则包含12500张图片，以数字编号，猫狗的出现顺序已随机打乱。

## 2.2 探索性可视化

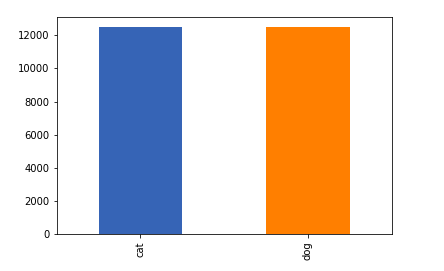


图4 猫和狗的图片各占比例

因为训练集的图片是已经标注过的，所以这里对训练集图片进行可视化，看看猫和狗的图片各占多大的比例，得到的柱状图如图4所示，可以看到训练集中猫和狗的图片各占一半，分别为12500张。但是根据上面的分析，训练集中实际上是存在一些异常值的。在稍后的数据预处理过程中会对异常图片进行删除，所以在预处理过后还需要对清洗过的数据再次进行可视化，观察清洗后的情况。

## 2.3 算法和技术

这里准备采用已在ImageNet上预训练过的4种卷积神经网络模型：VGGNet[1]，ResNet[2]，Inception v3[3]和Xception[4]来实现猫狗识别算法。ImageNet拥有一千多万张图片，对1000种物品进行分类，其中就有118种狗类和7种猫类，由它所导出的特征向量能高度概括图像中包含了哪些内容。首先，先分别去掉这四个模型的顶层（即全连接层），然后分别用25000张训练集图片在4个模型上进行预测，输出并保存得到的特征向量，得到4个特征向量文件；分别读取这些特征向量文件，将它们融合成一个特征向量；添加自己的全连接层，然后对模型进行编译构建；最后使用融合特征向量训练模型，并对测试集进行预测得到预测结果。

## 2.4 基准模型

按照毕业项目要求，我最后所得到的模型要能进入Kaggle排行榜前10%，即在Public Leaderboard上LogLoss值低于0.06127。

# 3. 方法

## 3.1 数据预处理

为了提高算法的性能，有必要对上面“数据的探索”部分提到的异常值进行检测和删除。采用的方法是使用ImageNet预训练过的上述4种模型来检测图片中存在的异常值。通过研究清单[5]可以知道ImageNet包含1000种分类，有猫和狗的品种，其中猫有7个品种，狗有118个品种。查阅Keras文档[6]可知准确率最高的模型是Xception，其Top-1和Top-5的准确率分别为0.790和0.945。这里的Top-N准确率是指在预测得到的概率值中，前N个结果包含正确值的占比。这里的关键在于通过不断试验选取一个合适的Top-N值，来尽可能多的检测出异常值并降低误报率。

首先，从Xception入手，选取Top-5值，并以1000张图片为样本，检测出了15张图片，对这些图片进行人工检查，发现很多正常图片被识别为异常值，效果不是很理想。然后把Top值提高到10，20和30进行试验，并适当提高样本量，开始检测出一些异常图片，例如图5这样的。通过不断地试验和观察，最后发现Top-60是一个比较理想的值。



图5 非猫非狗的图片

找到合适的Top值后，下面就是结合4种模型分别对全体训练集进行检测，综合4种模型输出的结果得到118张图片，通过人工分析发现这些被判定为异常值的图片有以下几种情况：

1. 图片中有猫狗，但尺寸太小，不够清晰；
2. 图片中有猫狗，但图片内容比较复杂；
3. 图片非猫非狗；
4. 图片是猫和狗的卡通形象；

我认为情况1应该保留，因为这里不能假定所有输入模型的图片都是清晰的，总会有一些模糊的图片，模型应当对这样的图片具有一定的健壮性；情况2也应该保留，模型也应该对一些复杂的图片具有一定的健壮性；情况3毫无疑问应该删除；情况4则比较主观，但我训练该模型的目的是为了识别真实世界的猫狗，所以我选择删除。最终被删除的有38张图片。清洗后的训练集图片有24962张，其中猫有12483张，狗有12479张。

## 3.2 执行过程

为了方便之后使用Keras的ImageDataGenerator作为图片数据生成器，在训练模型之前首先要针对训练数据和测试数据重构目录结构。具体做法是创建trains和tests两个文件夹，结构如下：

* trains
* dogs：包含训练集中所有为狗的图片
* cats：包含训练集中所有为猫的图片
* tests
* test：包含测试集中所有的图片

为了避免浪费磁盘空间，新创建的文件夹中的所有图片都为原始图片的符号链接文件（symbol link）。

我采用的方法是迁移学习中的“导出特征向量”法，使用ImageNet预训练过的4种模型，去掉它们的全连接层（include\_top=False），只保留卷积层，使用ImageDataGenerator读取重构过的目录结构获得训练生成器和测试生成器，以batch size等于64为单位进行预测，然后将预测得到的特征向量分别导出为4个h5文件：

* bottleneck\_features\_resnet.h5
* bottleneck\_features\_xception.h5
* bottleneck\_features\_inception.h5
* bottleneck\_features\_vgg16.h5

这种方法基于预训练模型进行“特征工程”，将抽取到的特征向量保存下来，然后再根据特征向量创建自己的全连接层，最后编译构建自己的模型。

紧接着，读入刚才导出的全部特征向量，做合并和随机洗牌，就得到了训练集X\_train（24962\*6656）和y\_train（24962\*1），以及测试集X\_test（12500\*6656）。有了特征向量，构建模型的过程就比较简单了，只需要一层Dense层，激活函数为sigmoid，采用Adadelta作为优化算法，二元交叉熵为损失函数，以准确率为评估指标，然后在训练集上训练100代，其中抽取20%作为验证集。

执行过程中还是遇到了一些困难的。由于对Keras库不熟悉，在导出特征向量的过程中执行predict\_generator()函数提供了错误的steps参数，导致训练非常慢，Resnet50上用了5个多小时，导出的特征向量文件有19G，后来查了些资料并向其他同学请教问题，发现自己的steps设定有误，于是修改了这个bug，后来每个模型导出的时间只需要5分钟左右，且特征向量文件在375M左右。导出特征向量的关键函数如图6所示。

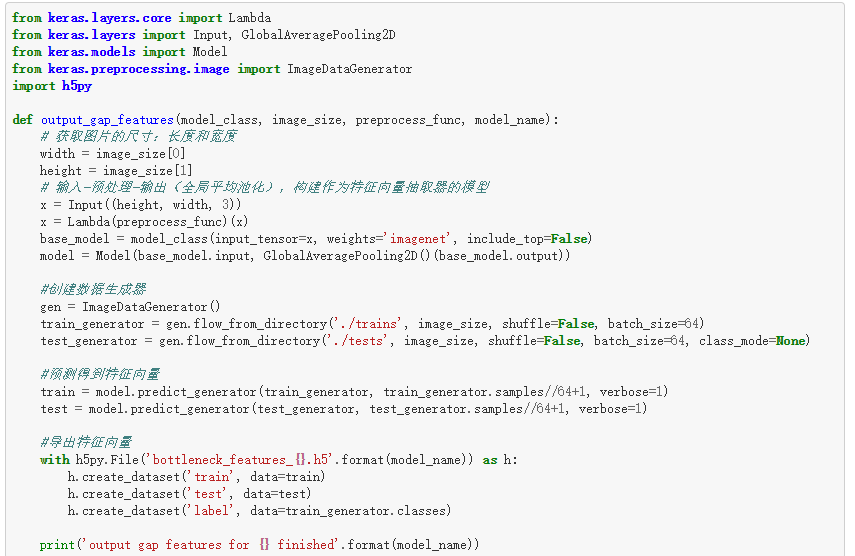


图6 导出特征向量的关键函数

## 3.3 完善

在这一部分，你需要描述你对原有的算法和技术完善的过程。例如调整模型的参数以达到更好的结果的过程应该有所记录。你需要记录最初和最终的模型，以及过程中有代表性意义的结果。你需要考虑的问题：

* 初始结果是否清晰记录了？
* 完善的过程是否清晰记录了，其中使用了什么技术？
* 完善过程中的结果以及最终结果是否清晰记录了？

# 4. 结果

## 4.1 模型的评价与验证

在这一部分，你需要对你得出的最终模型的各种技术质量进行详尽的评价。最终模型是怎么得出来的，为什么它会被选为最佳需要清晰地描述。你也需要对模型和结果可靠性作出验证分析，譬如对输入数据或环境的一些操控是否会对结果产生影响（敏感性分析sensitivity analysis）。一些需要考虑的问题：

* 最终的模型是否合理，跟期待的结果是否一致？最后的各种参数是否合理？
* 模型是否对于这个问题是否足够稳健可靠？训练数据或输入的一些微小的改变是否会极大影响结果？（鲁棒性）
* 这个模型得出的结果是否可信？

## 4.2 合理性分析

在这个部分，你需要利用一些统计分析，把你的最终模型得到的结果与你的前面设定的基准模型进行对比。你也分析你的最终模型和结果是否确确实实解决了你在这个项目里设定的问题。你需要考虑：

* 最终结果对比你的基准模型表现得更好还是有所逊色？
* 你是否详尽地分析和讨论了最终结果？
* 最终结果是不是确确实实解决了问题？

# 5. 结论

## 5.1 结果可视化

在这一部分，你需要用可视化的方式展示项目中需要强调的重要技术特性。至于什么形式，你可以自由把握，但需要表达出一个关于这个项目重要的结论和特点，并对此作出讨论。一些需要考虑的：

* 你是否对一个与问题，数据集，输入数据，或结果相关的，重要的技术特性进行了可视化？
* 可视化结果是否详尽的分析讨论了？
* 绘图的坐标轴，标题，基准面是不是清晰定义了？

## 5.2 思考

在这一部分，你需要从头到尾总结一下整个问题的解决方案，讨论其中你认为有趣或困难的地方。从整体来反思一下整个项目，确保自己对整个流程是明确掌握的。需要考虑：

* 你是否详尽总结了项目的整个流程？
* 项目里有哪些比较有意思的地方？
* 项目里有哪些比较困难的地方？
* 最终模型和结果是否符合你对这个问题的期望？它可以在通用的场景下解决这些类型的问题吗？

## 5.3 后续改进

在这一部分，你需要讨论你可以怎么样去完善你执行流程中的某一方面。例如考虑一下你的操作的方法是否可以进一步推广，泛化，有没有需要作出变更的地方。你并不需要确实作出这些改进，不过你应能够讨论这些改进可能对结果的影响，并与现有结果进行比较。一些需要考虑的问题：

* 是否可以有算法和技术层面的进一步的完善？
* 是否有一些你了解到，但是你还没能够实践的算法和技术？
* 如果将你最终模型作为新的基准，你认为还能有更好的解决方案吗？