**机器学习纳米学位**

**毕业项目——猫狗大战**

范耀华 优达学城

2018年4月25日

**I. 问题的定义**

**项目概述**

人类能轻易地识别照片中的物体，比如照片中是一只猫还是还是一只狗。

但计算机却很难识别，这涉及到计算机视觉领域。因为计算机底层只认得0和1，对于图片来说，它只能区分一个个像素的RGB值。它难以整体的“感觉”到图像到底是什么。

猫狗大战是[kaggle.com](https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition)在2013年举办的一场竞赛，即通过计算机识别一张图片是猫还是狗，它同时提供了25000张标注好的图片以供算法训练。随着近几年机器学习的发展，特别是计算硬件性能及大数据量级的提高，现在深度学习方法识别猫狗准确率非常高。

此项目基于深度学习和卷积神经网略尝试识别猫狗。

**问题陈述**

问题：使用深度学习方法识别一张图片是猫还是狗。  
输入：一张彩色图片  
输出：是猫还是狗

在这个项目中，该问题是一个二分类问题。最后输出图片是狗的概率[0,1]

本项目使用Keras, TensorFlow为后端。设计流程为下：

1.下载数据集 [data](https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data)

2.数据预处理  
调用Keras的数据预处理API [ImageDataGenerator](https://keras.io/zh/preprocessing/image/)进行图像预处理,需要先对数据集按猫狗分目录以便后续方法flow\_from\_directory调用

3.导出特征向量  
使用Keras的预训练模型提取特征，导出多个不同预训练模型的特征向量综合

4.载入特征向量  
载入上一步生成的特征向量，合成一条特征

5.构建模型  
调用Keras的API构建模型

6.训练模型  
分割训练集验证集，然后进行训练

7.预测测试集  
对测试集进行预测，导出csv，然后上传到kaggle相关页面查看得分

8.继续优化  
可更换预训练模型，或者对预训练模型进行微调（fine-tune），或者进行数据增强（data augmentation）等

期望结果是构建的模型在kaggle 上排名靠前。

**评价指标**

评估标准为 𝐿𝑜𝑔𝐿𝑜𝑠𝑠，使用 kaggle 官方的二分类 𝐿𝑜𝑔𝐿𝑜𝑠𝑠 公式：

其中

* n是测试集中的图像数量
* 是图像是狗的预测概率
* ：如果图像是狗，则为1;如果是猫，则为0
* log()是自然对数e

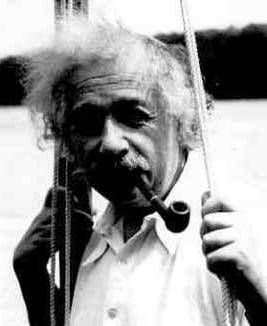
较小的log loss 更好

**II. 分析**

**数据的探索**

项目数据集可以从[kaggle](https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data)上下载。  
此训练集共有25000张jpg图片，猫狗各12500张，通过文件名区分，图片尺寸不定大小不定。测试集共有12500张jpg图片，没有区分是猫还是狗。

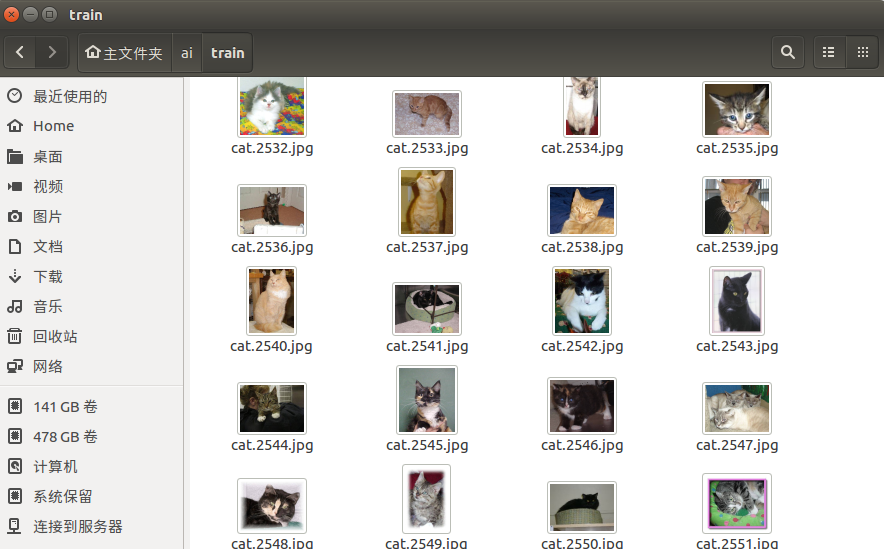
图片场景有：单独出现，多个出现，有人类入镜等。例如：  
  
 cat.247.jpg cat.81.jpg cat.1102.jpg

还有极少的异常值。例如：  
  
 dog.1043.jpg dog.1773.jpg

对于这部分异常值，后续可单独处理。

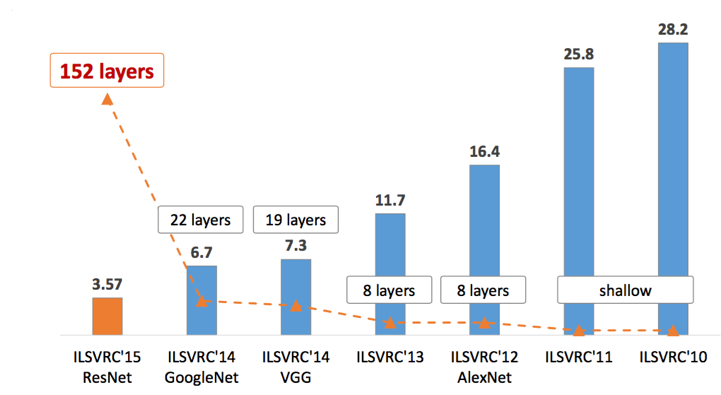
由于图片大小不一致，这里将会采用Keras的ImageDataGenerator函数进行统一的预处理，生成批次的带实时数据增益的张量图像数据。  
另外需要将数据集划分出训练集和验证集，比例暂定8:2。

**探索性可视化**



如图，训练数据集。

**算法和技术**



如图所示，这是一个计算机视觉相关领域比赛历年的得奖模型表。它们分别获得了ILSVRC比赛分类项目的2012年冠军（AlexNet[1], top-5错误率16.4%，使用额外数据可达到15.3%，8层神经网络）、2014年亚军（VGGNet[2]，top-5错误率7.3%，19层神经网络），2014年冠军（GoogleNet[3]，top-5错误率6.7%，22层神经网络）和2015年的冠军（ResNet[4]，top-5错误率3.57%，152层神经网络）。

在这个项目中，将使用前人优秀的模型进行预训练的网络融合获得更好的表现。

预训练的网络选取ResNet50, Xception, Inception V3

以下为在Keras中的API

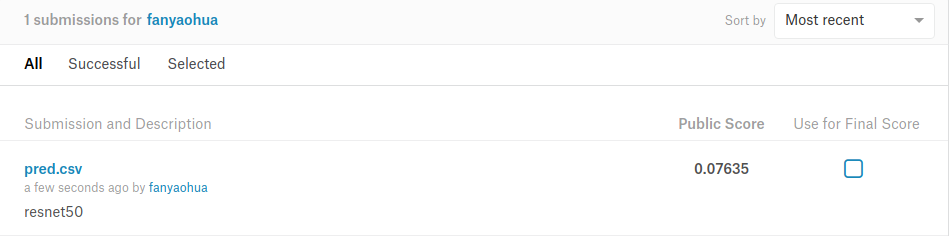
keras.applications.resnet50.ResNet50(include\_top=**True**, weights='imagenet', input\_tensor=**None**, input\_shape=**None**, pooling=**None**, classes=1000)

keras.applications.xception.Xception(include\_top=**True**, weights='imagenet', input\_tensor=**None**, input\_shape=**None**, pooling=**None**, classes=1000)

keras.applications.inception\_v3.InceptionV3(include\_top=**True**, weights='imagenet', input\_tensor=**None**, input\_shape=**None**, pooling=**None**, classes=1000)

**基准模型**

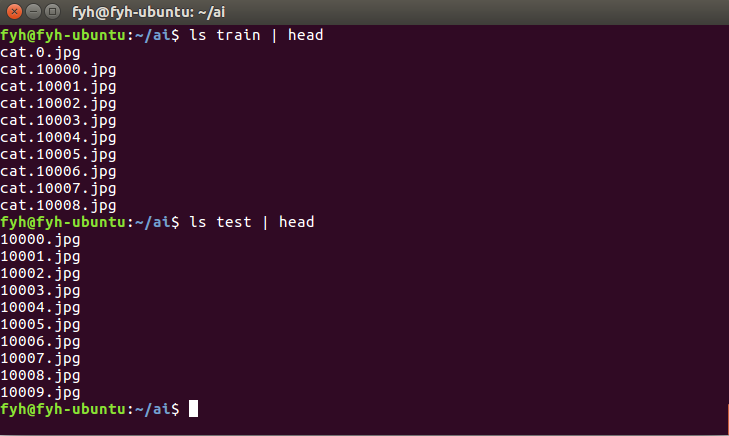
使用ResNet50的迁移学习模型作为基准模型。  
下图为使用基准模型在kaggle 上的跑出的分数截图

  
logloss分数为0.07635

基准阈值为 kaggle 排行榜前10%，也就是第131/1314名，也就是在 Public Leaderboard 上的 logloss 要低于 0.06127。

**III. 方法**

**数据预处理**



这是kaggle提供的数据集中train 和test文件夹的文件概览。

我们将使用Keras提供的ImageDataGenerator函数进行图片预处理

keras.preprocessing.image.ImageDataGenerator(featurewise\_center=**False**,

samplewise\_center=**False**,

featurewise\_std\_normalization=**False**,

samplewise\_std\_normalization=**False**,

zca\_whitening=**False**,

zca\_epsilon=1e-6,

rotation\_range=0.,

width\_shift\_range=0.,

height\_shift\_range=0.,

shear\_range=0.,

zoom\_range=0.,

channel\_shift\_range=0.,

fill\_mode='nearest',

cval=0.,

horizontal\_flip=**False**,

vertical\_flip=**False**,

rescale=**None**,

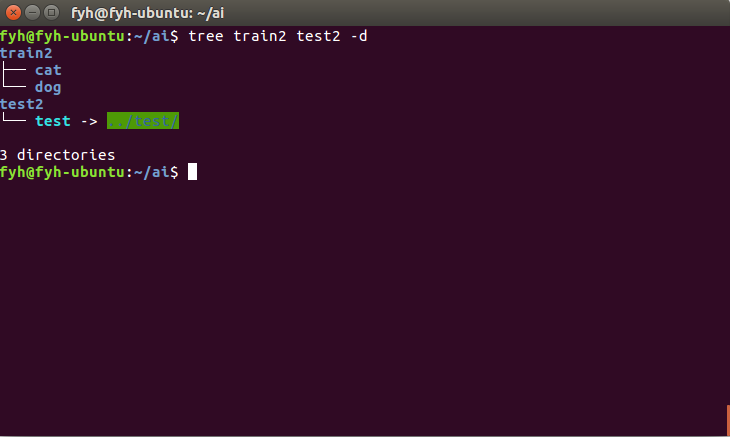
preprocessing\_function=**None**,

data\_format=K.image\_data\_format())

由于该函数需要将不同种类的图片分在不同的文件夹中，因此我们需要对数据集进行预处理。

1. **import** os
2. **import** shutil
4. train\_filenames = os.listdir('train')
5. train\_cat = filter(**lambda** x:x[:3] == 'cat', train\_filenames)
6. train\_dog = filter(**lambda** x:x[:3] == 'dog', train\_filenames)
8. **def** rmrf\_mkdir(dirname):
9. **if** os.path.exists(dirname):
10. shutil.rmtree(dirname)
11. os.mkdir(dirname)
13. rmrf\_mkdir('train2')
14. os.mkdir('train2/cat')
15. os.mkdir('train2/dog')
17. rmrf\_mkdir('test2')
18. os.symlink('../test/', 'test2/test')
20. **for** filename **in** train\_cat:
21. os.symlink('../../train/'+filename, 'train2/cat/'+filename)
23. **for** filename **in** train\_dog:
24. os.symlink('../../train/'+filename, 'train2/dog/'+filename)

处理之后文件夹结构如下



**执行过程**

在这一部分， 你需要描述你所建立的模型在给定数据上执行过程。模型的执行过程，以及过程中遇到的困难的描述应该清晰明了地记录和描述。需要考虑的问题：  
- 你所用到的算法和技术执行的方式是否清晰记录了？  
- 在运用上面所提及的技术及指标的执行过程中是否遇到了困难，是否需要作出改动来得到想要的结果？  
- 是否有需要记录解释的代码片段(例如复杂的函数）？

**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？

**IV. 结果**

（大概 2-3 页）

**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？

**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？

**V. 项目结论**

（大概 1-2 页）

**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？

**对项目的思考**

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？

**需要作出的改进**

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？

\*\* 在提交之前， 问一下自己... \*\*

* 你所写的项目报告结构对比于这个模板而言足够清晰了没有？
* 每一个部分（尤其**分析**和**方法**）是否清晰，简洁，明了？有没有存在歧义的术语和用语需要进一步说明的？
* 你的目标读者是不是能够明白你的分析，方法和结果？
* 报告里面是否有语法错误或拼写错误？
* 报告里提到的一些外部资料及来源是不是都正确引述或引用了？
* 代码可读性是否良好？必要的注释是否加上了？
* 代码是否可以顺利运行并重现跟报告相似的结果？