

# 机器学习毕业项目开题报告

---

题目: 猫狗大战项目

作者: Kyle Chen

版本: 20180505v1

日期: 20180505

---

## 项目背景

- 项目涉及的相关研究领域
- 在猫狗大战项目研究中, 重点研究了深度学习在图像识别中的应用. 猫狗大战是一个典型的二分类应用场景, 主要用于将图片中的猫, 狗区分出来. 在此项目中, 输入是一张相片, 相片中, 可以是任何物体/动物, 当其中出现猫时, 期望预测结果为猫. 如若为狗时, 期望预测结果为狗. 当两者同时出现, 或两者都未出现时, 期望输出无法预测.
- 在现实生活中, 不乏很多多分类问题, 但是我们只有在对二分类问题非常了解的情况下, 才能对多分类问题有更深入的理解, 也对往后处理分类问题落地起到了至关重要的作用.

## 问题描述

- 解决办法所针对的具体问题
- 在此项目中, 我们需要解决针对图像的训练与分类问题, 这是一个有监督学习的二分类问题. 首先, 要先对训练集中的数据进行训练; 在多次训练与学习中, 提升对数据预测的准确度; 其次, 在训练完成之后, 对测试集进行预测与评分.

## 输入数据

- 问题中涉及的数据或输入是什么
- 在此项目中, 输入应为一张图片. 图片中可以是任何物体, 不仅限于猫狗. 但是当图片中出现猫时, 我们期待我们的分类器能将其归类为猫这个类型; 当出现狗时, 则期待分类器能将其归类为狗这一类; 当两者都出现, 或者两者都检查

不出来的时候, 直接输出无法预测则可.

- 在代码实现中, 可以使用keras.preprocessing中的image库加载RGB图像.
- 研究下kaggle给我们提供的样本:

```
→ dogs-vs-cats-redux-kernels-edition x ls -ahl train/ | grep -i cat |  
head -n 3  
-rw-r--r--      1 Kyle  staff    12K Sep 20   2013 cat.0.jpg  
-rw-r--r--      1 Kyle  staff    16K Sep 20   2013 cat.1.jpg  
-rw-r--r--      1 Kyle  staff    34K Sep 20   2013 cat.10.jpg  
  
→ dogs-vs-cats-redux-kernels-edition x ls -ahl train/ | grep -i dog |  
head -n 3  
-rw-r--r--      1 Kyle  staff    31K Sep 20   2013 dog.0.jpg  
-rw-r--r--      1 Kyle  staff    24K Sep 20   2013 dog.1.jpg  
-rw-r--r--      1 Kyle  staff    12K Sep 20   2013 dog.10.jpg
```

不难发现, 样本中的Y, 就是文件的prefix = [cat | dog], 标签和样本是绑定在一起的, 这方便了对样本打乱.

- 接着我们统计下训练集中猫,狗的类型分布:

```
→ dogs-vs-cats-redux-kernels-edition x find train/ -name "cat*" | wc -  
l  
12500  
→ dogs-vs-cats-redux-kernels-edition x find train/ -name "dog*" | wc -  
l  
12500
```

可以发现, 这里的猫狗是均匀分布的, 我们可以在训练集中取一部分来作为训练集, 另外一部分作为验证集.

- 在获得训练集, 验证集之前, 需要将train/目录下的文件打乱, 然后按照一定的比例将其划分到训练集, 验证集中.

## 解决办法

- 针对给定问题的解决方案
- 在一切开始前, 我们需要准备我们的数据集. 在猫狗大战这个项目中, 可以直接从kaggle上下载DataSet. 在安装kaggle api之后, 我们可以直接在终端执行:

```
→ Dogs_vs_Cats x kaggle competitions download -c dogs-vs-cats-redux-kernels-edition
```

- 接下来, 我们需要将DataSet中的数据分成训练集, 验证集, 测试集. 可以参考比例6:2:2来划分.
- 使用keras框架构建深度卷积神经网络, 这里我们使用Xception进行迁移学习训练, 在第一次调用时会自动到github上下载相关的训练好的特征权重模型, 供我们后面训练使用. 在构建模型时, 并不是直接将其加载进来就能直接使用, 我们需要将其嵌入我们需要训练的模型中去. 例如说, 猫狗大战, 是一个二分类问题, 所以我们需要将最终的预测结果(softmax layer的输出)修改为两类.
- 在训练前, 我们可以选择冻结/解冻训练模型中卷积层, 来选择我们需要训练哪些卷积层. 这里我们需要尝试多种方式, 可以将最开始的几层冻结, 或者将其全部冻结只训练输出层, 亦或者冻结其中某几层. 这个过程需要大量的测试来确保我们的模型能达到最优的结果.
- 在训练完成后, 我们选取最优的模型, 统计此模型在测试集中的成绩, 作为最终得分.

## 基准模型

- 用来与你的解决方案进行比较的一些简单的、过去的模型或者结果
- 在使用Xception训练前, 我们还需要将其与CNN做个比较, 在其不使用迁移学习的模型时, 是否还能有好的表现.
- 这里我们可以使用relu作为Hidden\_nodes的激活函数, softmax作为输出函数, 在中间添加Conv2D, MaxPooling2D. 如若需要, 可以加入BN防止过拟合. 当然, 在没有实际代码经过多次测试的基础下, 暂且不能确定最优的模型框架.

## 评估指标

- 衡量你解决方案的标准

- 可以使用模型在测试集中的得分(准确率)来对指标评估.
- 在kaggle页面中, 也为我们提供了验证logloss函数:

$$LogLoss = -\frac{1}{n} \sum_{i=1}^n [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

- 当然, 很重要的一点, 在最后, 需要进入kaggle猫狗大战挑战中的前10%, LogLoss分数大概在0.256以上.

## 设计大纲

- 你的解决方案如何实现, 如何获取结果
- 将猫狗图片分类, 从数据集中分割出训练集, 验证集, 测试集.
- 定义模型框架, 这里包含两个部分, 在一开始, 我们需要对CNN做一个测试, 看看是否可以直接用CNN解决这个二分类问题, 在训练完成, 获得在测试集中的准确率后, 对参数做持续调优, 看看是否还有优化空间; 其次, 使用Xception搭建新的模型, 将其融入我们的二分类模型中, 训练并获得测试集准确率, 并对其持续调优, 以获得更高的得分.
- 最后选取得分最高, 最好的模型, 作为最终模型.
- 在网上选取六张图片, 对其做分类, 评估其在实际生产环境中的准确率.
- 最后, 在上述基本模型上, 在加载图片时, 可以使用skimage对数据进行增强, 还可以加入模型融合, 类似于boosting, stacking的方法来获得更高的准确率.

---

备注: 本论文无摘抄部分, 纯属原创.