机器学习纳米学位
开题报告

叶剑 Udacity

2018-03-04

一、项目背景

猫狗大战，之前是 2013 年 Kaggle 的一道竞赛题目，要求写一个算法区分图片里面是猫还是狗，是图像识别问题中典型的一种。图像识别问题（CAPTCHA 或 HIPS）可以用来区分人类和机器，以达到减少垃圾邮件，网站密码攻击等目的。对于图片里面是猫还是狗的问题，人很容易识别，而机器识别起来却比较困难。究其原因，人类从出生之日起，就有无数次机会看到猫或狗，从而区分它们；但是机器没有这种多次识别猫或狗的经验，所以无从识别。 但是如果机器能够像人类一样，能够找到海量的猫或狗的照片加上一个智能的算法训练，并一一识别区分它们，经过训练后的机器理论上应该也和人类一样，能够识别图片里面的猫和狗。 早在 2007 年就已经有论文[dogcat]指出机器识别图片中的猫或狗正确率超过 80%。 而 kaggle2013 年的竞赛 kaggle Public Leaderboard 上面，榜首机器识别图片中的猫或狗正确率更是达到了 98.533%。所以从理论和实践都说明机器学习来区分图片里面是猫还是狗的问题，应该并且能够得到解决。利用图像识别来区分人和机器已经变得不再安全了。

二、问题描述

该项目要求给出一张彩色图片，机器通过使用深度学习方法识别这张图片是猫还是狗。识别图片是猫还是狗的 score(log loss)要在 kaggle Public Leaderboard 前 10%，也就是 score 要低于 0.06，而且每次给出不同的猫或狗的图片，score 都达到上面的要求。

三、数据或输入

训练数据包括 25000 张猫和狗的图片，这个训练集的所图面的文件名都含有 cat 或 dog 的标签。训练数据集的下载地址为：[train.zip]. 测试数据包括 12500 张图片，里面的所有图片以数字 id 命名。测试集的下载地址为：[test.zip]. 训练数据用来训练算法模型，测试数据用来验证训练后的模型的预测准确率，也就是说，对测试集的每一张图片，要预测图片里面是一条狗的概率。

四、解决方法描述

首先，要对训练数据进行必要的预处理，然后建立卷积神经网络模型，用预处理后的训练数据对模型进行训练，然后用测试数据对模型进行预测并评估模型的准确性。

五、评估标准

对模型的评估方案使用 score(log loss)，score(log loss)越低，方案越好，目标是 score(log loss)低于 0.06

Score(log loss)的计算方式如下[摘自 kaggle]：

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

n 是测试集中的图片数，$\hat{y}_i$是第 i 张图片预测为一条狗的可能性，yi 是测试集的正确结果，yi=1 代表图片是一条狗，yi=0 代表图片是一只猫。Log（）是自然对数（底数为 e 的对数）。

提交文件只有 2 列，图片 id 和图片为狗的概率，格式如下：

```
id,label
1,0.5
2,0.5
3,0.5
...
```

六、基准模型

基准模型使用卷积神经网络模型，模型框架使用 tensorflow，keras。使用 ResNet-50 搭建 CNN。搭建算法框架代码如下（代码出自 [deep-learning-models]）：

```
# Determine proper input shape
input_shape = _obtain_input_shape(input_shape,
                                  default_size=224,
                                  min_size=197,
                                  data_format=K.image_data_format(),
                                  include_top=include_top)

if input_tensor is None:
    img_input = Input(shape=input_shape)
else:
    if not K.is_keras_tensor(input_tensor):
        img_input = Input(tensor=input_tensor, shape=input_shape)
    else:
        img_input = input_tensor
if K.image_data_format() == 'channels_last':
    bn_axis = 3
else:
    bn_axis = 1

x = ZeroPadding2D((3, 3))(img_input)
x = Conv2D(64, (7, 7), strides=(2, 2), name='conv1')(x)
x = BatchNormalization(axis=bn_axis, name='bn_conv1')(x)
x = Activation('relu')(x)
x = MaxPooling2D((3, 3), strides=(2, 2))(x)

x = conv_block(x, 3, [64, 64, 256], stage=2, block='a', strides=(1, 1))
x = identity_block(x, 3, [64, 64, 256], stage=2, block='b')
x = identity_block(x, 3, [64, 64, 256], stage=2, block='c')

x = conv_block(x, 3, [128, 128, 512], stage=3, block='a')
x = identity_block(x, 3, [128, 128, 512], stage=3, block='b')
x = identity_block(x, 3, [128, 128, 512], stage=3, block='c')
x = identity_block(x, 3, [128, 128, 512], stage=3, block='d')

x = conv_block(x, 3, [256, 256, 1024], stage=4, block='a')
x = identity_block(x, 3, [256, 256, 1024], stage=4, block='b')
x = identity_block(x, 3, [256, 256, 1024], stage=4, block='c')
x = identity_block(x, 3, [256, 256, 1024], stage=4, block='d')
x = identity_block(x, 3, [256, 256, 1024], stage=4, block='e')
x = identity_block(x, 3, [256, 256, 1024], stage=4, block='f')

x = conv_block(x, 3, [512, 512, 2048], stage=5, block='a')
x = identity_block(x, 3, [512, 512, 2048], stage=5, block='b')
x = identity_block(x, 3, [512, 512, 2048], stage=5, block='c')

x = AveragePooling2D((7, 7), name='avg_pool')(x)

if include_top:
    x = Flatten()(x)
    x = Dense(classes, activation='softmax', name='fc1000')(x)
else:
    if pooling == 'avg':
        x = GlobalAveragePooling2D()(x)
    elif pooling == 'max':
        x = GlobalMaxPooling2D()(x)
```

七、项目设计

下载好训练和测试数据后，计划通过以下步骤解决该问题：

1. 导入测试和训练数据
2. 针对训练数据进行预处理，包括异常图片检测，标准化，one-hot 编码，数据混洗等。
3. 构建神经网络模型
4. 用训练数据训练神经网络

用测试数据测试模型