机器学习纳米学位

猫狗大战开题报告

唐玉山
2018 年 8 月 17 日星期五

1. 项目背景

项目涉及的相关研究领域是图像识别，该项目是 Kaggle 的机器学习竞赛项目《Dogs vs. Cats Redux: Kernels Edition》，他是基于 2013 年的 《Dogs vs. Cats》项目，当时深度学习还没有得到充分应用。但是现在就连种黄瓜的农民也开始使用神经网络技术提高他们的收入。基于深度学习使得图像识别变得简便，但是如何使得图像的识别准确度得到进一步提高却是一个很大的挑战。。

2. 问题描述

解决办法所针对的具体问题是：如何使用深度学习方法识别一张图片是猫还是狗，给出是狗的概率。

3. 数据和输入

项目中需要用于模型训练的猫的图片数据和狗的图片数据，由于素材越多越有利于模型训练，所以应该尽可能多的收集素材。

3.1 Kaggle 数据集（All.zip）

下载地址：https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data
Train 文件夹包含了 25000 张狗和猫的图片，图片命名方式是：

- 猫的图片：cat.编号.jpg
- 狗的图片：dog.编号.jpg

Test 文件夹包含了 12500 张图片，图片命名方式是：

- 编号.jpg

搜集的图片数据的尺寸不一，例如：在 kaggle 数据集中的 train 文件夹中的最小图片 cat.4821.jpg(60*39)，最大图片是 cat_835.jpg(1023*768)，图片的大小是不同的，需要进行缩放处理。

3.2 扩展数据集 （images.tar.gz）

下载地址：http://www.robots.ox.ac.uk/~vgg/data/pets/
下载 Groundtruth_data: images.tar.gz 文件，包含了 7393 张猫和狗的图片，文件命名方式是：

- 品种_编号.jpg
- 产地品种编号.jpg

3.3 输入

项目中的输入的数据就是猫或者狗的图片，输出的是狗的概率。

4. 解决方法描述

针对给定问题的解决方案是：通过深度学习框架，结合图像处理技术，进行模型训练。验证最终模型，识别图片中的狗和猫，给出狗的概率值。

下面是使用的相关技术：

- 深度学习框架：keras, tensorflow
- 图像处理：openCV

5. 基准模型

本项目的最低要求是 kaggle Public Leaderboard 前 10%，也就是在 Public Leaderboard 上的 logloss 要低于 0.06127,位置是第 131 名。

参考链接：https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/leaderboard

6. 评估指标

使用 kaggle 官方的评估标准：$LogLoss$
计算公式：

$$L(Y,P(Y|X))=-\log P(Y|X)=-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}y_{ij}\log(p_{ij})$$

变量含义：
Y：输出变量
X：输入变量
L：损失函数
N：输入样本数
M：可能的类别数
Yij：是一个二值指标，标识类别 j 是否是输入实例 xj 的真实类别

Pij：模型或分离器预测输入实例 xj 属于类别 j 的概率
对于当前项目，只有两类可将公式化简：

$$-\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\, p_i+(1-y_i)\log\,(1-p_i)\right)$$

变量含义：
Yi：输入实例 xi 的真实类别，
Pi：为预测输入实例 xi 属于类别 1 的概率

对所有样本的对数损失表示对每个样本的对数损失的平均值，完美的分离器的对数损失为 0

7. 项目设计

解决方案的实现的步骤如下：

7.1.数据预处理

将训练数据集切分为：训练集、验证集。在模型训练时设置 fit 的参数 validation_split=0.2，将 20%的训练集数据作为验证集。

7. 2.图像预处理

缩放图片：将图片缩放为 299*299 的图像
归一化处理：对数据集中的每张图像的像素值除以 127.5 再减去 1
公式：
x = p / 127.5 – 1
变量含义：
p：图像的像素
x：归一化后的图像像素

7.3. 训练模型

在训练数据集上训练 InceptionV3 模型，步骤如下：
1. 构建不带分类器的预训练模型
base_model = InceptionV3(weights='imagenet', include_top=False)
2. 添加全局平均池化层
X = base_model.output
X = GlobalAveragePooling2D()(x)
3. 添加一个全连接层
X = Dense(1024, activation='relu')(x)

4. 添加一个分类器，我们有 2 类

```
Predictions = Dense(2, activation='softmax')(x)
```

5. 构建我们需要训练的完整模型

```
Model = Model(inputs=base_model.input, outputs=predictions)
```

6. 首先只训练顶部的几层（随机初始化的层），锁住所有 InceptionV3d 卷积层

```
For layer in base_model.layers:
    Layer.trainable = False
```

7. 编译模型（一定要在锁层以后操作）

```
Model.compile(optimizer='rmsprop', loss='categorical_crossentropy')
```

8. 在新的数据集上训练几代

```
Model.fit_generator(……)
```

9. 现在顶层应该训练好了，开始微调 InceptionV3 的卷积层。

锁住底下的几层，然后训练其余的顶层。

查看每一层的名字和层号，看看应该锁多少层

```
For I, layer in enumerate(base_model.layer):
    Print(I, layer.name)
```

10. 我们选择训练最上面的两个 Inception block

锁住前面 249 层，然后放开之后的层

```
For layer in model.layers[:249]:
    Layer.trainable = False
For layer in model.layers[249:]:
    Layer.trainable = True
```

11. 重新编译模型，使上面的修改生效

设置一个很低的学习率，使用 SGD 来微调

```
From keras.optimizers import SGD
Model.compile(optimizer=SGD(LR=0.001, momentum=0.9), loss='categorical_crossentropy')
```

12. 继续训练模型

训练最后两个 Inception block 和两个全连接层

```
Model.fit_generator(……)
```

7.5. 验证模型

在验证数据集上对训练模型进行验证，对模型进行调参。

选择合适的 Epoch 训练次数：如果随着 Epoch 的次数增加，logloss 在一定时间内（比如 5 到 10 次）变化很小，就可以停止 Epoch。开始时可以把 Epoch 次数设置的大一些，观察在哪个地方准确度变化很小，就把 Epoch 设置成几

为了直观的判断调参的后模型的表现，将每次的 logloss 进行保存，进行可视化展示。

7.4. 测试模型

在测试数据集上使用训练好的模型进行测试。

8. 参考文献

[1] Dogs vs. Cats Redux: Kernels Edition,2017. https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition.

[2] Dogs vs. Cats,2013. https://www.kaggle.com/c/dogs-vs-cats.

[3] Kaz Sato,How a Japanese cucumber farmer is using deep learning and TensorFlow,https://cloud.google.com/blog/products/gcp/how-a-japanese-cucumber-farmer-is-using-deep-learning-and-tensorflow.

[4] Log Loss,http://wiki.fast.ai/index.php/Log_Loss

[5] Christian Szegedy,Vincent Vanhoucke,Sergey Ioffe,Jon Shlens,Zbigniew Wojna, Rethinking the Inception Architecture for Computer Vision,https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Szegedy_Rethinking_the_Inception_CVPR_2016_paper.pdf