Machine Learning Engineer Nanodegree

Capstone Proposal

白杰
2018年6月9号

Proposal

Domain Background

本项目是美国的汽车保险和金融公司State Farm 2年前在kaggle上发起的一个比赛，主题是“Can computer vision spot distracted drivers?”。根据CDC汽车安全部门统计，五分之一的车祸是由司机的分心造成的，也就是说每年因分心导致交通事故死亡的人数有3000人。严重的交通事故给肇事双方带来的伤害是无法弥补的，如果能有一个方式来提高驾驶的安全性那么将是非常有意义的事情。包括目前谷歌、百度以及各个汽车厂和创业公司都在大力投入的自动驾驶技术都是希望把驾驶变成一件安全可靠的事情。这里State Farm希望得到的能够识别出驾驶员是否在安全驾驶的方式对目前人类驾驶来说是很有效的提高驾驶安全性的方式。能够完成这个项目不仅仅是对自己能力的提升，也是为了安全驾驶做一点努力。
该项目的图像是在一个可控的实验条件下得到的。司机并没有实际的在控制车辆，车辆是由一辆卡车拉着在路上行使的。
目前深度学习特别是卷积神经网络[1]在图像识别领域表现出了巨大的威力，在ImageNet的图像识别挑战赛中基本上所有的高分团队都是使用的CNN[2],取得了很好的效果。ILSVRC 2014中GoogLeNet[3] 将目标检测的平均准确率提高到0.439329,分类错误降低到0.06656。而司机正常驾驶与分心的时候比如打电话，发短信等行为还是有很大的却别的，具备实现的可能性。

Problem Statement

State Farm提供了通过在汽车仪表盘上的摄像机获取到司机驾驶过程中的2D图像作为数据集，并且提供了已经分类好的图片作为训练集和测试集来供模型训练使用。最终想实现的是能够利用模型准确的将司机的驾驶行为分类。这属于监督学习中的分类问题，模型通过学习训练集中的特征，来识别没有见过的图像的驾驶行为，可以通过使用CNN模型来实现该功能。

Datasets and Inputs

本项目中提供的数据集是司机驾驶过程中的照片，有的在发信息、吃东西，打电话或者是专心开

车等。已经给出了司机可能的10种状态，如下：

- c0: safe driving
- c1: texting - right
- c2: talking on the phone - right
- c3: texting - left
- c4: talking on the phone - left
- c5: operating the radio
- c6: drinking
- c7: reaching behind
- c8: hair and makeup
- c9: talking to passenger
  训练集和测试集是已经分类好的司机驾驶过程中的照片，其中训练集有22424张，测试集含有79726张照片。训练集中的照片都已经按照照片所属的类别放在了以状态命名的文件夹下。以便于我们训练。在训练过程中将训练集按照8：2的比例分成训练集和验证集，使用交叉验证来遍历所有的可能。在每一轮的训练中用训练集来训练模型，然后用验证集来评价其泛化能力。虽然模型在训练的时候没有使用验证集，但是模型每轮训练结束会更具验证集的泛化来调整，也就相当于验证集已经渗透进模型中了。所以使用模型从未见过的测试集来测试模型的性能。
  我们的目标是识别出每幅图片中司机的状态。
  一个司机的照片只会出现在训练数据集或测试数据集之中的一个。
  项目还提供了一个叫做driver_imgs_list.csv的表格，里面是数据集中的图片对应的分类类别和司机id。

Solution Statement

本项目准备采用卷积神经网络(CNN)来实现该功能。卷积运算是神经网络最基本的组成部分，这里的卷积与数学上的卷积不同，数学中卷积的定义是先将矩阵做镜像反转，之后在作元素乘积求和，而在这里的卷积运算只是是将filter与元图像矩阵中的数据乘积之后加权求和。卷积运算能够在CNN中能够在图像处理领域取得非常好的效果的主要原因是CNN具有参数共享和稀疏连接的特性。参数共享即同一个filter在整个图像上训练出来的参数只有一份，实现了图像的平移不变的特性。同时也减小了模型的参数数量，便于对高清图像的处理。稀疏连接指的是CNN输出层中的每个元素只依赖于输入层中和filter做卷积的那几个元素，输入层中的其他元素都不会影响该输出层中的元素。CNN通过这两种机制减少参数，以便我们用更小的训练集来训练它，从而预防过拟合。
经典的CNN网络介绍如下：

> 1. LeNet-5：主要是针对灰度图片来训练的，现在已经很少使用了，但是其提出的网络结构至今仍被大量使用：一个或多个卷积层后面跟一个池化层，然后又是若干个卷积层在接一个池化层，然后是全链接层,最后是输出 。
> 2. AlexNet网络,与LeNet-5相比，其结构相似，但是AlexNet网络更大，并且AlexNet采用了Relu激活函数代替了LeNet-5使用的sigmod激活函数。

3. VGG网络是一种专注于构建卷积层的简单网络，结构的优点是：结构规整，卷积层的数量是翻倍增长，而池化后的图像长、宽都减少一半。其缺点是需要训练的参数巨大。VGG16网络包含16个卷积层和全链接层，总共含有约1.38亿个参数，VGG19比VGG16的网络更大。虽然VGG19比VGG16的网络结构更大，但是表现却没有比VGG16更好。
4. Google的Inception网络；该网络带来了一些结构上的改变。不需要人为的决定使用哪个过滤器或者是否需要池化，而是由网络自行确定这些参数。我们可以给网络添加这些参数的所有可能值，然后把这些输出连起来，让网络自己学习它需要什么杨的参数，采用哪些filter的组合。该网络的优点是代替了人工选择网络结构。但是由于我们加入了太多的结构，计算成本变得巨大。这里引入了1x1卷积，得到一个bottleneck layer，在可以有效减少计算成本的同时还可以给网络增加非线性函数。
5. ResNets：由于存在梯度消失和梯度爆炸的问题，非常深的网络是很难训练的，ResNets采用跳跃连接构，可以构建超过100层的ResNets网络。
   对于计算机视觉的问题的处理，如果从头开始训练一个网络的权重，一方面需要巨大的计算成本（包括时间和金钱），另一方面也需要大量的训练样本。在本项目中我既没有能够训练上亿个参数的训练样本（训练集只有22424个样本），也没有训练的条件（使用的是aws免费的EC2 p2.xlarge 实例,该实例有4个CPU,60GB内存,1个NVIDIA K80 GPU,12GB GPU内存。机器系统为 Ubuntu 18.04.5 LTS)。所以这里使用迁移学习。采用keras里面已经训练好的模型。这些模型使用的数据集是ImageNet。在本项目中我将把他们当作一个很好的初始化用在我自己的神经网络中，站在前人们的肩膀上，好好的利用他们的成果。

Evaluation Metrics

这里采用kaggle上该项目的要求，提交到kaggle上的是每张图片被分类到每个类别的概率，使用logloss来评估模型的表现：
$$logloss =-\frac{1}{N}\sum_{i=1}^N\sum_{j=1}^My_{ij}\log(p_{ij})$$
其中N表示测试集中的照片数量这里是79726，M是照片的类别数量这里是10，log是自然对数。y_{ij} 当第i张图片属于第j类时为1，其余为0，p_{ij} 是模型预测第i张照片属于第j类的概率。logarithmic loss用来衡量分类模型的性能。一个完美的分类模型的log loss为0.
与常用的衡量标准精度(Accuracy)相比，Accuracy中计算预测的类别与实际的类别是否相同，只有是或不是两种可能。而log loss考虑的是预测的不确定性，是预测到每一个类别的概率，可以对模型有更细微的衡量。
在提交的数据没有要求某张图片的所有类别的概率和为1，因为每个概率会除以所有类别的概率和，做归一化处理。
为了避免出现log(0)的情况，对于模型预测的结果使用 $max(min(p,1-10^{-15}),10^{-15})$ 来代替直接输出。

Project Design

1、导入数据集：项目提供的训练数据集中已经将每一类的图片放在了一个单独的文件夹下，并以该类别命名，使用scikit-learn库中的load_files函数，可以很方便的得到训练数据和对应的类别

标记。
2、数据预处理：本项目采用Keras来搭建深度学习网络模型，使用Tensorlow作为后端。

1. Keras的输入要求是一个4D的tensor即(nb_samples, rows, columns, channels)，因此将训练数据转化为这种格式的张量。
2. 原始数据的通道顺序为RGB，为了与预训练模型使用的数据相匹配，需要根据不同模型的需要使用opencv将RGB图像转换为BGR图像。
3. 根据预训练模型的输入进行归一化过程，比如对于ResNet-50 即对所有的像素都减去像素均值，因为预训练的模型是使用ImagNet训练的，所以这里的像素均值使用的是根据所有ImageNet图像算出来的均值，采用BGR的格式[123.68, 116.779, 103.939]。
   3、搭建模型：
4. 首先自己搭建一个简单的CNN网络，比如使用深度学习项目中给出的让自己实现的网络结构，如下

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 223, 223, 16)      208
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 111, 111, 16)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 110, 110, 32)      2080
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 55, 55, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 54, 54, 64)        8256
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 27, 27, 64)        0
_________________________________________________________________
global_average_pooling2d_1 ( (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 133)               8645
=================================================================
Total params: 19,189.0
Trainable params: 19,189.0
Non-trainable params: 0.0
_________________________________________________________________
```

   看看效果如何，这一步的作用主要是验证整个程序的流程是否正确。这里可以使用一些optimizer比如Adam，RMSprop，SGD等来看看哪个表现更好。loss采用项目给的logloss;metrics使用accuracy。也调节不同epochs的值看模型的表现。
5. 利用迁移学习：在keras中用“imagenet”预训练好的CNN模型(比如VGG16[4]，ResNet50[5]，Inception[3]和Xception[6])后面加上GAP[7]层，dropout[8]和FCL层等方式构建自己的网络，来给图像中司机的行为分类。训练模型的时候注意在训练数据中分隔出训练集和验证集，使用网格搜索来遍历所有可能的组合。
   4、预测结果：
6. 使用训练好的模型预测测试集中的图片，并将结果保存为所给的示例中的结构，即每一行记录的是被预测图片属于10中类别的概率。
7. 将结果上传到kaggle。

Reference

1. LeCun, Yann. "LeNet-5, convolutional neural networks". Retrieved 16 November 2013.
2. "ImageNet Large Scale Visual Recognition Competition 2014 (ILSVRC2014)". Retrieved 30 January 2016.
3. Szegedy, Christian; Liu, Wei; Jia, Yangqing; Sermanet, Pierre; Reed, Scott; Anguelov, Dragomir; Erhan, Dumitru; Vanhoucke, Vincent; Rabinovich, Andrew (2014). "Going Deeper with Convolutions". Computing Research Repository. arXiv:1409.4842
4. ABSTRACT arXiv:1409.1556v6 [cs.CV] 10 Apr 2015
5. Kaiming He, "Deep Residual Learning for Image Recognition". arXiv:1512.03385 [cs.CV], 2015
6. Franc¸ois Chollet. "Xception: Deep Learning with Depthwise Separable Convolutions". arXiv:1610.02357v3 [cs.CV] 4 Apr 2017
7. Min Lin, Qiang Chen, Shuicheng Yan. "Network In Network". arXiv:1312.4400 [cs.NE]
8. Srivastava, Nitish; C. Geoffrey Hinton; Alex Krizhevsky; Ilya Sutskever; Ruslan Salakhutdinov (2014). "Dropout: A Simple Way to Prevent Neural Networks from overfitting" (PDF). Journal of Machine Learning Research. 15 (1): 1929–1958.