机器学习纳米学位

猫狗大战 唐玉山

2018 年 08 月 25 日

1. 问题的定义

1.1. 项目概述

项目基于深度学习和图像识别技术，目标是通过神经网络算法训练处一个模型，使用训练的模型对图片进行识别，给出图片是狗的概率。

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。现在的深度学习技术进步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。

近年来出现了很多针对图像识别的算法，其中神经网络算法表现尤为耀眼，该算法的效果优于模式识别、支持向量机等传统

项目中的深度学习使用的是卷积神经网络(Convolutional Neural Network, CNN)是一种前馈神经网络，他的人工神经元可以响应一部分覆盖范围内的周围单元，对于大型图像处理有出色的表现。

项目中输入的数据就是猫或者狗的图片，输出的是图片是狗的概率。模型使用的训练数据集是 Kaggle 竞赛项目《Dogs vs. Cats Redux: Kernels Edition》的数据集，该数据集包含了 25000 张狗和猫的训练图片，12500 张测试图片。通过对训练图片进行训练最终得出在测试集上面测试表现较好的模型。

1.2. 问题陈述

项目解决的问题类型是有监督图像的二分类问题，具体是解决的是如何使用深度学习识别一张图片中的是猫还是狗，给出是狗的概率。

解决方案是：首先将数据集中的猫和狗的图像分开，利用图像处理技术进行预处理。训练集切分为训练集，验证集。对图像进行缩放，归一化处理。使用神经网络技术在训练数据集上训练 InceptionV3 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。在测试数据集上使用训练好的模型进行测试。验证最终模型，识别图片中的狗和猫，给出图片是狗的概率。

1.3. 评价指标

使用 kaggle 官方的评估标准：$LogLoss$，与准确率作为评估指标对比，LogLoss 要更有说服力。因为假设有两个模型，它们准确率一样，但是一个模型对一张猫的图片做出 99%是狗的判断，另外一个模型对这张猫的图片做出 60%的判断，这样第一个模型的 LogLoss 会很大，第二个模型的 LogLoss 要小很多，这样可以根据 LogLoss 来判断模型好坏。

$LogLoss$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：

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

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。

2. 分析

2.1. 数据的探索

Kaggle 数据集中包含训练数据集 Train 文件夹和测试数据集 Test 文件夹，

Train 中的图片可以根据文件名称区分出是猫的图片还是狗的图片。

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

图片数据的尺寸不一，需要进行缩放处理。例如：在 kaggle 数据集中的 train

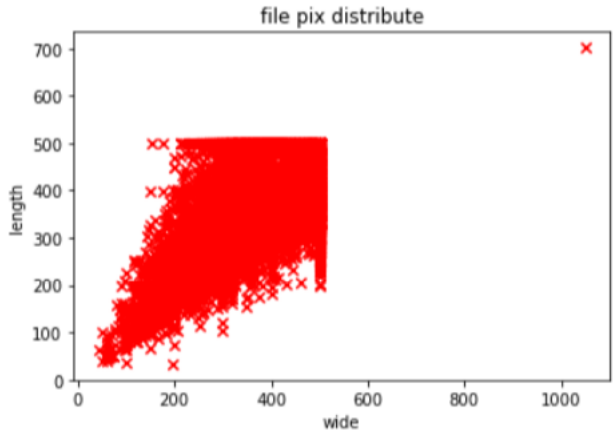
文件夹中的最小图片 cat.4821.jpg(60*39)，最大图片是 cat_835.jpg(1023*768)。

数据集中的猫狗图片除了大小和分辨率有差别外，由于是来自真实世界的照片，猫和狗的形态多种多样，拍摄照片的场景和各不相同。这些干扰因素，增加了识别难度。

2.2. 探索性可视化

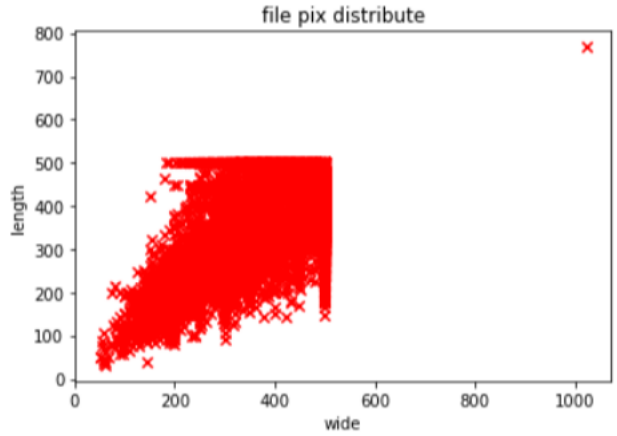
对文件的大小进行了探索。

训练数据集狗的文件大小散点图:



从图中可以看到大多数训练数据图片大小在 500x600 之间

测试集中猫的文件大小散点图:



从图中可以看到大多数测试数据图片大小在 500x600 字节之间

2.3. 数据预处理

2. 3. 1. 数据清理

从 kaggle 下载的数据中也有异常数据。

1 类别标签与图片信息不相符的

例如：虽然标识了是狗的图片，但是图片中没有狗。

2 照片的背景过于复杂，识别起来很困难

例如：虽然有狗但是有很多干扰元素。

异常数据清理方法：

先设置一个 ImageNet 评价指标的 Top 值，Top 值越低越严格。通过算法获取图片的 ImageNet 类型标识 Top 值数组，看数组中的值是否有在狗的 ImageNet 值数组中，如果没有就认为是异常数据，即不是狗的图片，就进行剔除。

这里选择 4 种算法:ResNet50, VGG19, Xception, InceptionV3 取 4 个算法处理后的并集进行数据清理。

2.3.2. 数据归一化

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

2.4. 算法和技术

图像处理

图像处理是指对图像进行分析、加工、处理，使其满足视觉、心里或其他要求的技术。图像处理是信号处理在图像领域上的一个应用。目前大多数的图像是以数字形式存储，因而图像处理很多情况下指图像处理。此外，基于光学理论的处理方法依然占有重要的地位。

图像处理是信号处理的子类，另外与计算机科学、人工智能等领域也有密切关系。

传统的一维信号处理的方法和概念任然可以直接应用在图像处理上，比如降噪、量化等。然而，图像属于二维信号，和一维信号相比，它有自己特殊的一面，处理的方式和角度也有所不同。

由于数据集中的图像大小分辨率等存在差异，所以我们使用图像处理技术对训练数据中的图像进行预处理，例如：进行归一化操作。让图像数据方便进行进一步分析处理。

深度学习（deep learning）

深度学习是机器学习的分支，是一种试图使用包含复杂结构或由多重非线性变换构成的多个处理层对数据进行高层抽象的算法。深度学习是机器学习中一种对数据进行表征学习的算法。观测值（如一幅图像）可以使用多种方式来表示，如每个像素强度值的向量，或者更抽象的表示成一系列边、特定形状的区域等。而使用某些特定的表示方法更容易从实例中学习任务（例如，人脸识别

或面部表情识别)。深度学习的好处是用非监督学习或半监督学习的特征学习和分层特征提取高效算法来替代手工获取特征。

表征学习的目标是寻求更好的表示方法并创建更好的模型来从大规模的未标记数据中学习这些表示方法。表示方法来自神经科学，并松散地创建在类似神经系统中的信息处理和对通信模式的理解上，如：神经编码，试图定义拉动神经元反应之间的关系及大脑中的神经元的电活动之间的关系。

至今已有的深度学习框架有：深度神经网络、卷积神经网络、深度置信网络、递归神经网络。已经被应用在计算机视觉、语音识别、自然语言处理、音频识别、生物信息学等领域，并取得了极好的效果。

卷积神经网络（Convolutional Neural Network, CNN）

卷积神经网络是一种前馈神经网络，它的人工神经元可以响应一部分覆盖范围内的周围单元，对于大型图像处理有出色表现。

卷积神经网络由一个或多个卷积层和顶端的全连接层（对应经典的神经网络）组成，同事也包括关联权重和池化层（pooling layer)。这一结构使得卷积神经网络能够利用输入数据的二维结构。与其他深度学习结构相比，卷积神经网络在图像和语音识别方面能够给出更好的结果。这一模型也可以使用反向传播算法进行训练。相比较其他深度、前馈神经网络，卷积神经网络需要考量的参数更少，使之成为一种颇具吸引力的深度学习结构。

这次项目中使用的卷积神经网络，建立一个识别猫和狗的神经网络模型

卷积层（Convolutional layer）

卷积神经网络中每层卷积层由若干卷积单元组成，每个卷积单元的参数都是通过反向传播算法最佳化得到的。卷积运算的目的是提取输入的不同特征，第一

层卷积层可能只能提取一些低级的特征如边缘、线条和角等层级，更多层的网络能从低级特征中迭代提取更复杂的特征

反向传播（Backpropagation BP）

反向传播是误差反向传播的简称，是一种与最优化方法（如：梯度下降算法）结合使用的，用来训练人工神经网络的常见方法。该方法对网络中所有权重计算损失函数的梯度。这个梯度会反馈给最优化方法，用来更新权值以最小化损失函数。

反向传播要求有对每个输入值想得到的已知输出，来计算损失函数梯度。它被认为是一种监督式学习方法，虽然它也用在一些无监督网络（如：自动编码器）中。

反向传播算法（BP 算法）主要由两个阶段组成：激励传播，权重更新。

三层网络算法（一个隐藏层）：

```
初始化网络权值（通常是小的随机值）
Do
    forEach 训练样本 ex
        prediction = neural-net-output(network, ex)     //正向传递
        Actual = teacher-output(ex)
        计算输出单元的误差(prediction-actual)
        计算 wh 对于所有隐藏层到输出层的权值 //反向传递
        计算 wi 对于所有输入层到隐藏层的权值 //继续反向传递
          更新网络权值 //输入层不会被误差估计改变
Until 所有样本正确分类或满足其他停止标准
Return 该网络
```

InceptionV3 模型

在 2015 年 12 月提出的 Inception V3 结构借鉴 inception 的结构设计了采用一种并行的降维结构。

InceptionV3 网络主要做了两方面改造：

1.引入了大卷机分解为小卷积的思想

将一个较大的二维卷积拆成两个较小的一维卷积，如 7x7 卷积拆成 1x7 卷积核 7x1 卷积，这样可以节约大量参数，加速运算并减轻过拟合，同时增加了一层非线性扩展模型表达能力。这种非对称卷积结构相较对称拆分方式可以处理更多、更丰富的空间特征，增加特征多样性。

2.优化了 Inception 模块结构

InceptionV3 对原始的 Inception 模块做了改进，将 5x5 的卷积改为 2 个 3x3 的卷积如下图所示：

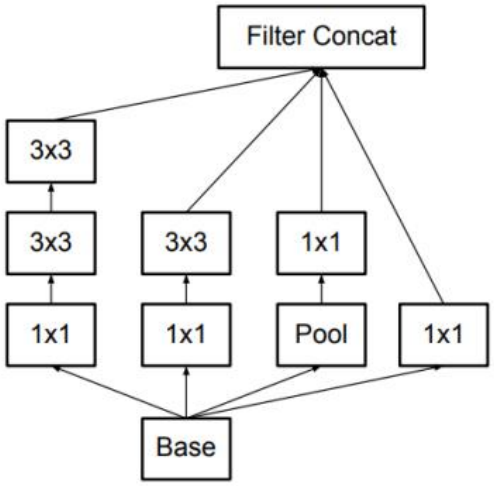


Figure 5. Inception modules where each 5×5 convolution is replaced by two 3×3 convolution, as suggested by principle 3 of Section 2.

Inception 模型只在网络后部出现，前部还是普通的卷积层。并且 InceptionV3 除了在 Inception 模块中使用分支，还在分支中使用分支，是网络中的网络。Inception 结构主要是把稀疏结构近似成几个密集的子矩阵，从而在减少参数的

同时，更加有效的利用计算资源。将全连接甚至一般的卷积转化为稀疏连接的目的是增加网络深度和宽度，以提升网络性能。InceptionV3 有很多优点：

weight 数量小于 VGG 和 Resnet，大小为 92MB，

2.5. 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。

3. 方法

3.1. 数据预处理

1.需要对训练数据进行分割：

Train 中的数据没有对猫狗进行区分，需要将猫和狗的数据分别放在不同的文件夹中。

2.图像预处理：

因为本项目中使用的是 InceptionV3 进行模型训练，需要对图片进行处理格式化为 299 x 299 的大小

3.2. 执行过程

训练 InceptionV3 模型

训练步骤如下：

1.构建不带分类器的预训练模型

2.添加全局平均池化层

3.全连接层，可选，如果精度够用则可以不加

4. 添加一个分类器，使用 1 个神经元，sigmoid 激活函数

5. 构建我们需要训练的完整模型

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

7.编译模型（一定要在锁层以后操作）

8. 在新的数据集上训练几代

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

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

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=0.001，使用 SGD 来微调

训练过程中由于图片预处理的时候需要大量内存和 CPU 使用，在内存不足的时候会报 MemoryError:的错误，这个时候两个方法：

1.减少数据量，会降低训练效果

2.分批量加载数据

3.增加硬件配置，如加内存，这种方法会增加训练成本，但是训练的效果比较好。

4.使用云服务

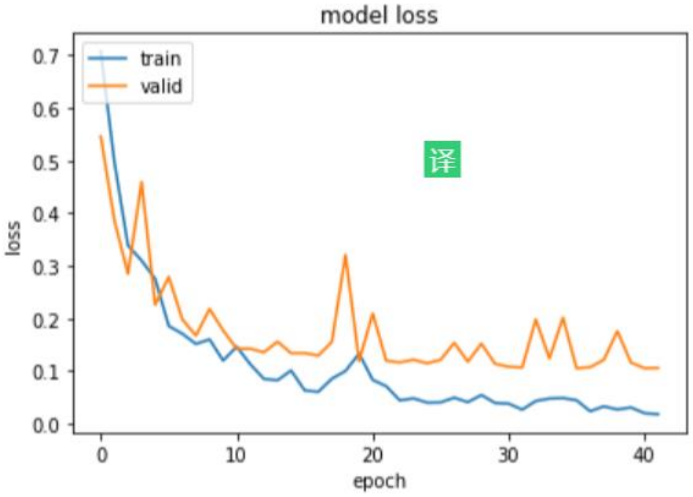
本项目中解决方法是第 4 种，在 aws 上面部署训练模型。因为本地训练的图片数量达到了 20000 万张，所以选择了 GPU 计算实例，类型=p2.8xlarge，vcpu=32，内存=488G，这样可以提高训练的速度。初步估算内存使用峰值为

80G 左右，使用内存不低于 100G，因为训练过程会消耗大量内存。

3.3. 完善

训练过程中 fit 函数的 epochs 参数是训练次数，这里并不是次数越多效果越好，一定次数后的效果就有所稳定，这样设置一个合理的 epochs 值很有必要，可以节约时间。采用 EarlyStopping 方法当 val_loss 若干轮不下降后自动停止训练。

初始时设置了 epochs=300, batch_size=64，EarlyStopping 的 patience=5 趋势图如下:



可以看到在 epochs=40 的时候就趋于稳定了。所以选择 epochs=40

可以看到下面是首次训练的 logloss 值，最小为 0.1201

通过设置一个很低的学习率=0.001，使用 SGD 来微调

4. 结果

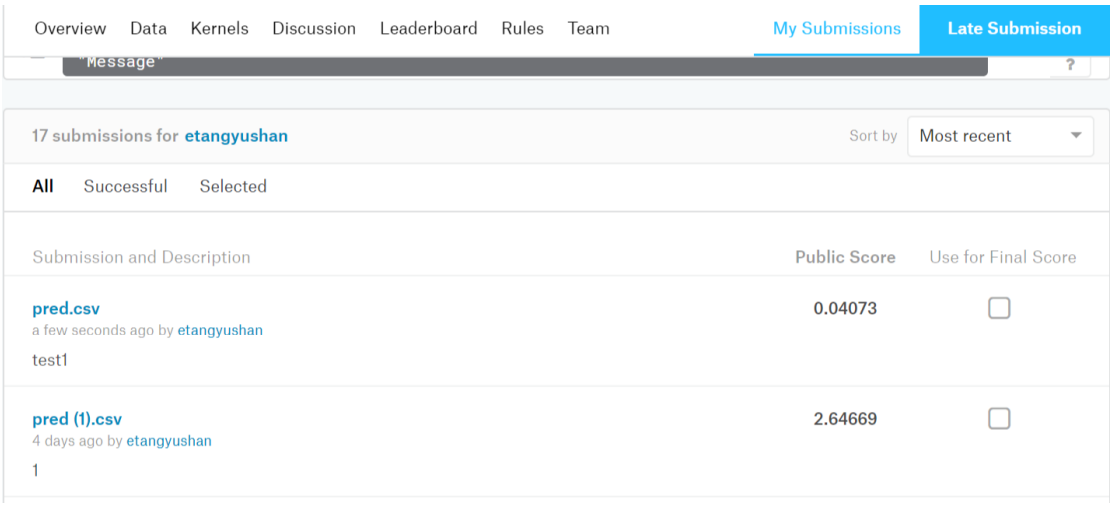
4.1. 模型的评价与验证

经过训练后的模型 Logloss 降低到 kaggle 中成绩排名的 10%以内，与期待的结果过相一致，参数经过调整后也很合理。

模型在训练数据较少的情况下表现不好，但是大数据量时，如果变动的数据量较小时对模型训练结果影响不明显。

经过在测试集上的验证，模型的效果很好，复合图片真实的情况。

提交 kaggle 得分：0.04073 满足要求



4.2. 合理性分析

对比基准模型，目前的模型表现很好。在测试集上面的表现也很稳定。

5. 项目结论

5.1. 对项目的思考

项目从查看项目要求文档开始，对给出的样例进行了仔细的阅读。对项目最终的成果有了大致的映象。

研究了 kaggle 的比赛数据，以确定基准模型。对数据集的样例的大小，图片分辨率等做了统计。最终依据模型训练的需要对图片进行预处理，归一化处

理。最后进行模型训练，验证模型，进行调参。确定最终模型后在测试集上进行验证。

搜集数据，对数据进行预处理，这部分工作对于机器的硬件要求较高。需要借助 gpu 运算，初期在自己机器上面进行预处理，结果在预处理后悔报错显示 MemoryError，最终解决方法是使用 AWS 进行图片的预处理和后续的模型训练，使用 AWS 极大的提高了训练的速度。并且使用 AWS 的竞价方式，解约了训练成本。

使用 AWS 的时候环境配置也是一个难题，在本地运行正常的在 AWS 环境会报错，总结了一下，大多数都是环境问题，常见的文件问题：

Python 版本问题，python2.7 与 python3.6 的兼容问题，本项目使用的是 python3.6 环境。

镜像中自带库版本与训练所需的库版本冲突导致，利用 anaconda 将正常的训练环境备份,镜像使用纯净版本的操作系统，手动配置运行环境，这样会减少莫名其妙的错误。

5.2. 需要作出的改进

训练过程中对机器要求比较高，

对于 InceptionV3 我们可以进一步增加网络深度，这个已经有相关算法出现如 InceptionV4，他增加了 InceptionV3 的深度。另外，Xception 是 Inception 架构的扩展，他用深度可分离的卷积代替了标准的 Inception 模块

6. 引用

[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_Incepti
on_CVPR_2016_paper.pdf
[6] Dogs vs. Cats Redux: Kernels Edition,2017. https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition.
[7] Dogs vs. Cats,2013. https://www.kaggle.com/c/dogs-vs-cats.
[8] 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.
[9] Log Loss,http://wiki.fast.ai/index.php/Log_Loss
[10] 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_Incepti
on_CVPR_2016_paper.pdf
[11] Krizhevsky, A.; Sutskever, I.; Hinton, G. E. Imagenet classification with deep convolutional neural networks
[12] OpenCV C interface: http://docs.opencv.org
[13] Lalis, Jeremias; Gerardo, Bobby; Byun, Yung-Cheol. An Adaptive Stopping Criterion for Backpropagation Learning in Feedforward Neural Network