机器学习工程师纳米学位毕业项目
基于深度学习的猫狗大战

杨青海

2018 年 9 月 28 日

目录

1. 问题的定义

1.1项目概述

项目将解决一个有监督图像的二分类问题，训练一个从给定图片中识别是猫或狗的模型，模型可以应用图像识别、图像搜索中。

在谷歌，图像分类、物体识别技术已经被广泛应用于谷歌无人驾驶车、YouTube、谷歌地图、谷歌图像搜索等产品中。谷歌通过图像处理技术可以归纳出图片中的主要内容并实现以图搜图的功能。这些技术在国内的百度、阿里巴巴、腾讯等科技公司也已经得到广泛的应用。

项目使用的模型是卷积神经网络(Convolutinoal Neural Network)。卷积神经网络在图像分类数据集上有非常突出的表现。卷积神经网络神经网络早在 1997 年便被用来解决字符识别问题，2012 年 ILSVRC 图像分类识别大赛中，AlexNet 获得冠军，掀起了卷积神经网络的研究热潮。[1]

项目选择的数据集来源于 Kaggle。数据集的类别是猫和狗。train 文件夹总共有 25000 张 JPG 格式的图片，12500 张猫，12500 张狗。此文件夹中的每个图像都将标签作为文件名的一部分。test 文件夹包含 12,500 个图像，根据数字 ID 命名。对于测试集中的每个图像，应该预测图像是狗的概率（1 =狗，0 =猫）。train 文件夹数据集用于训练模型，test 文件夹数据集用于测试并把测试的结果生成 csv 文件上传 kaggle。在 kaggle 上查看结果。

1.2问题陈述

项目选择的数据集是 Kaggle 下载的猫狗图片。图像来源有真实的猫狗图片，由于猫狗本身的相似度很大，再加上背景复杂、光影变化、位置等影响，都会加大识别难度。

1.3评价指标

评价指标使用对数损失函数(LogLoss)。LogLoss 是一个软分类准确率度量方法，使用概率来表示其所属的类别的置信度。LogLoss 具体的数学表达式为:

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

n: 测试集中的图像数量

y_i: 如果图像是狗 1，如果是猫 0

$\hat{y}_i$: 预测的图像是狗的概率

log():自然（基数 e）的对数

LogLoss 是对额外噪声(extra noise)的度量，这个噪声是由于预测值域实际值不同而产生的。因此最小化交叉熵，便是最大化分类器的准确率.

2. 分析

2.1数据可视化

数据集分为两个子集，训练数据集、测试数据集。最大的图片为 768x1023，最小的图片为 38x50。有两个离群数据，不过所有的数据都会转成统一大小。图片中

还会存在错误数据，训练数据集中总共检测出 67 个既不是猫也不是狗的异常数据，异常数据会被剔除。

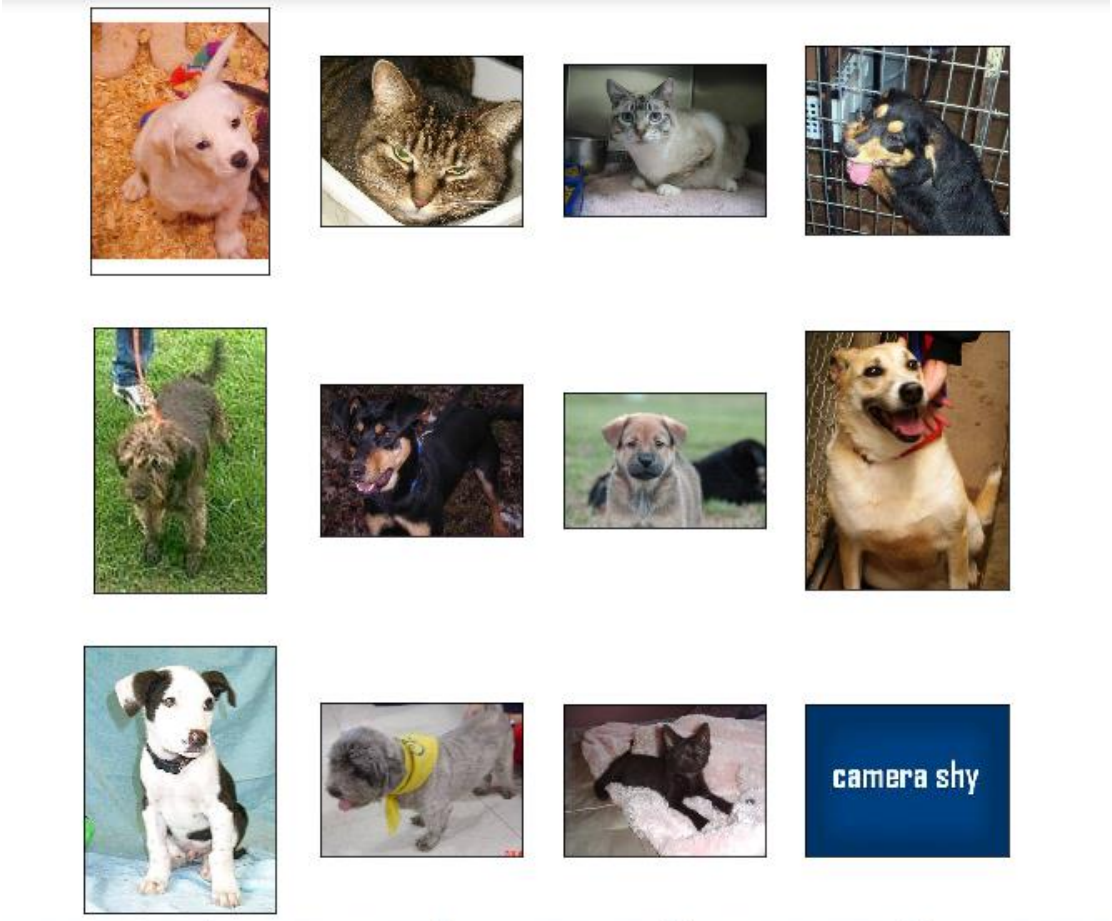


图 1

图片的分辨率分布图:

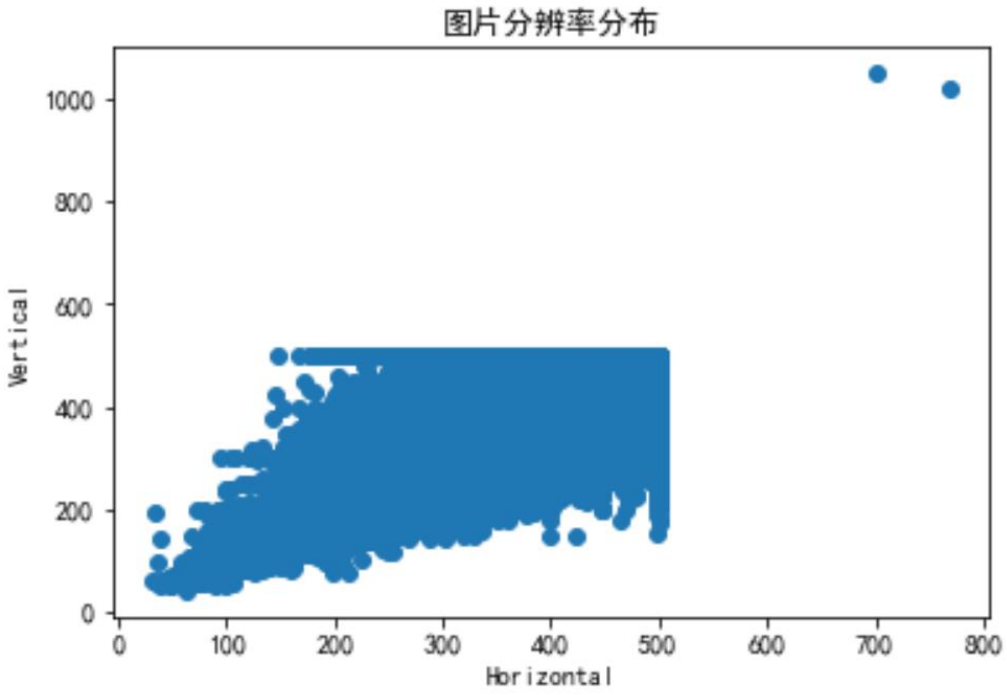


图 2

2.2算法和技术

2.2.1 神经网络

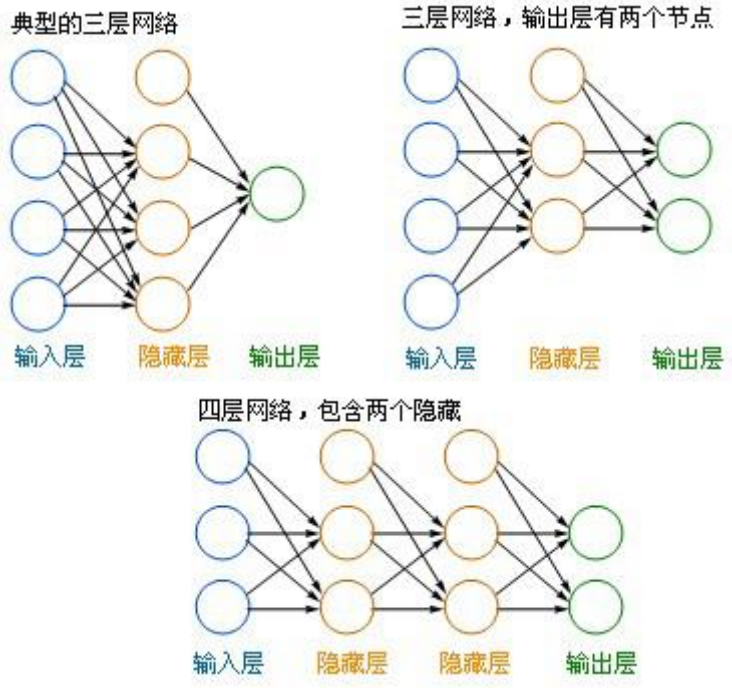


图 3：前向神经网络的结构，包含输入层、多层中隐藏层和输出层

人工神经网络类似于大脑中的神经网络。大脑中的神经网络由一个个的神经元组成，神经元从多个树突中获取输入信号，然后通过轴突输出；人工神经网络同样由一个个基本单元组成，单元也被称为神经元。每一个单元从其它单元获取输入的值，通过某种计算得到输出，作为下一个神经元的输入。每一个神经元是一个简单的非线性函数。输入首先加权求和，在数学上这是一个输入与权重的点积操作。然后通过 Sigmoid 得到输出。加权求和时的权重是神经元的参数。神经网络中的每一个神经元具有相同的结构，不同的是输入的权重。[2]

为了逼近图像到数字的映射关系，神经网络通过后向传播算法更新权重。对于分类的问题，可以使用神经网络预测的结果和图像的真实类别计算出损失函数交叉熵(cross entropy)。交叉熵对于每一个权重是可导的，因此可以通过梯度下降法更新权重的值。

神经网络可以有数百万的权重，后向传播算法则提供了一个高效计算误差关于权重梯度的算法。神经网络根据神经元连接方式的不同分为两种，前向神经网络(feed-forward neural network)和递归神经网络(reccurent neural network)。项目中使用的卷积神经网络具体来说时一种前向神经网络。前向神经网络的神经元组成一种分层的结构，前一层的神经元只与后一层的神经元相连，每一层之中的神经元没有连接。神经网络的层数又叫做深度，多层的神经网络就是深度学习名字的来源。

前向神经网络可以分为输入层、多层隐藏层和输出层，如图 3 所示。第一层隐藏层是输入数据的非线性映射，进而后一隐藏层也是前一层的非线性映射。这种映射不同于支持向量机，它是由数据决定的。模型通过权重更新学习得到了一种易于分类的映射。

2.2.2 卷积神经网络

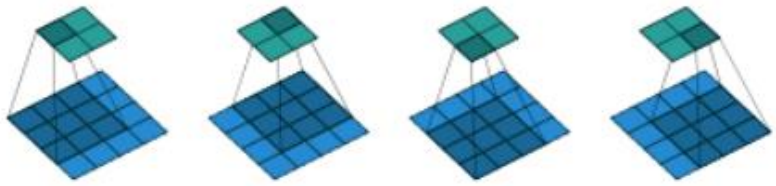


图 4 卷积神经网络的核心组成部分——卷积。4x4 的输入图像，经过 3x3 的卷积操作得到了 2x2 的特征图。

卷积神经网络（Convolutional Neural Network, CNN）是一种常见的深度学习网络架构，受生物自然视觉认知机制启发而来。1959 年，Hubel & Wiesel 发现了视觉系统的信息处理，可视皮层是分级的。[3]

普通的前向神经网络，后一层每一个神经元的输入是前一层所有神经元的输出。而卷积神经网络后一层每一个神经元的输入只与前一层局部的输出有关。这一局部区域也被称为感受野，通常大小为 3x3px 或 5x5px。因此卷积神经网络的前几层只响应于局部的输入，后续的隐藏层的输出逐步变得具有全局性。这等价于约束普通神经网络的一些权重必须为零。

卷积神经网络约束同一层的神经元具有相同的权重。这意味着每一层提取的是相同的特征，这一组相同的特征组成特征图(feature map)。这样不论特征出现的位置，这一层神经元都能够检测到，这称为平移不变性。一般一层神经元具有多组权重，即同时检测多组特征。

局部连接与权重共享大大减少了模型的参数个数，这意味着使用神经网络进行预测时的内存占用也会减少。内存占用的减少使得计算机能够训练层多、

每一层神经元数目更多的神经网络。

在数学上，这种局部连接同时权重共享的点积操作叫做卷积(convolution)，这也是卷积神经网络名字的来源。在图像处理领域，这组权重称为图像滤波器或者卷积核(kernel)。在卷积神经网络中，这种隐藏层称为卷积层。卷积操作如图謵所示，下侧蓝色图像代表卷积层的输入，上侧绿色图像代表卷积层的输出。图中的阴影代表一个卷积操作输入输出的对应关系，图中感受野的大小为 3x3。相邻两个感受野之间的距离称为跳跃距离(hop size)，一般为 1。图中 4x4 的输入图像，经过 3x3 的卷积操作得到了 2x2 的特征图。

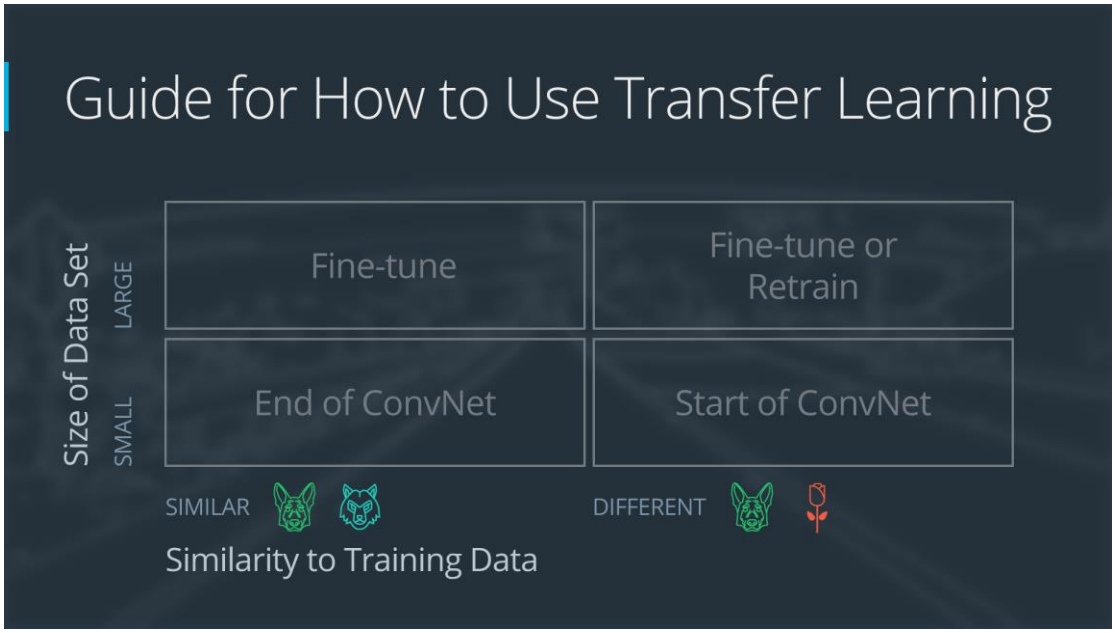
2.2.3 迁移学习

迁移学习是指对提前训练过的神经网络进行调整，以用于新的不同数据集。

取决于以下两个条件：

新数据集的大小，以及新数据集与原始数据集的相似程度

使用迁移学习的方法将各不相同。有以下四大主要情形：



情形 1：小数据集，相似数据

在这种情况下，因为数据与预训练模型的训练数据相似度很高，因此我们不需要重新训练模型。我们只需要将输出层改制成符合问题情境下的结构就好。我们使用预处理模型作为模式提取器。比如说我们使用在 ImageNet 上训练的模型来辨认一组新照片中的小猫小狗。在这里，需要被辨认的图片与 ImageNet 库中的图片类似，但是我们的输出结果中只需要两项——猫或者狗。在这个例子中，我们需要做的就是把 Dense Layer 和最终 Softmax Layer 的输出从 1000 个类别改为 2 个类别。

情形 2：小型数据集、不同的数据

因为数据集很小，因此依然需要注意过拟合问题。要解决过拟合问题，原始神经网络的权重应该保持不变，就像第一种情况那样。但是原始训练集和新的数据集并不具有相同的更高级特征。在这种情况下，新的网络仅使用包含更低级特征的层级。

情形 3：大型数据集、相似数据

训练大型数据集时，过拟合问题不严重；因此，你可以重新训练所有权重。因为原始训练集和新的数据集具有相同的更高级特征，因此使用整个神经网络。

情形 4：大型数据集、不同的数据

虽然数据集与训练数据不同，但是利用预先训练过的网络中的权重进行初始化可能使训练速度更快。因此这种情形与大型相似数据集这一情形完全相同。如果使用预先训练过的网络作为起点不能生成成功的模型，另一种选择是随机地初始化卷积神经网络权重，并从头训练网络

2.2.4 Xception

Xception 是 Inception 架构的扩展，用 depthwise separable convolution 来替换原来 Inception v3 中的卷积操作。

Inception v3 的结构图如下 Figure1。于是从 Inception v3 联想到了一个简化的 Inception 结构，就是 Figure 2。再将 Figure2 延伸，就有了 Figure3，Figure3 表示对于一个输入，先用一个统一的 1x1 卷积核卷积，然后连接 3 个 3x3 的卷积，这 3 个卷积操作只将前面 1x1 卷积结果中的一部分作为自己的输入（这里是将 1/3channel 作为每个 3x3 卷积的输入）。再从 Figure3 延伸就得到 Figure4，也就是 3x3 卷积的个数和 1x1 卷积的输出 channel 个数一样，每个 3x3 卷积都是和 1 个输入 channel 做卷积。[4]

Figure 1. A canonical Inception module (Inception V3).

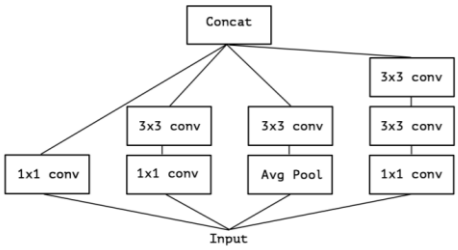


Figure 2. A simplified Inception module.

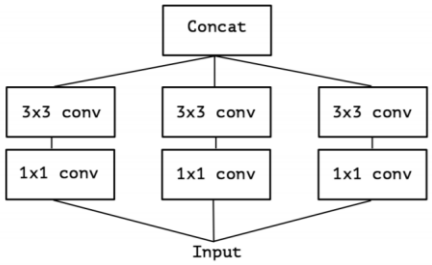


Figure 3. A strictly equivalent reformulation of the simplified Inception module.

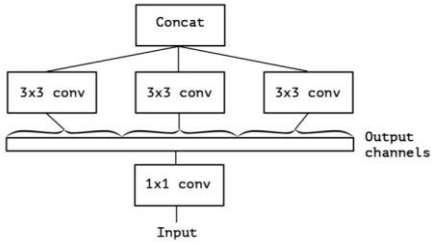
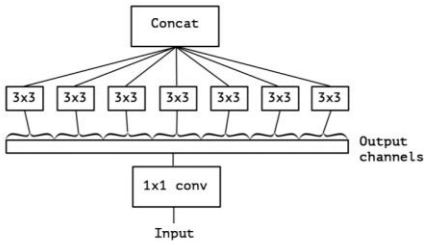


Figure 4. An "extreme" version of our Inception module, with one spatial convolution per output channel of the 1x1 convolution.



在 Xception 中主要采用 depthwise separable convolution，传统卷积操作如图 5，depthwise convolution 操作如图 6。

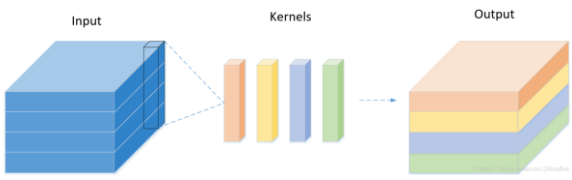


图 5

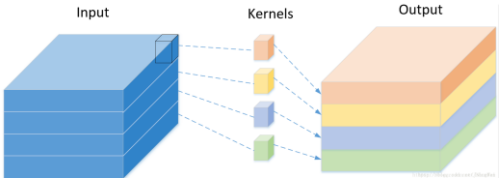


图 6

Xception 取得了比 ImageNet 上相比于 Inception V3 更多的准确率提升， 和 Inception V3 相比，Xception 的参数量有所下降，而训练时的迭代速度也没有明显变慢。另外，在 ImageNet 和 JFT 上的训练过程都显示，Xception 在最终准确率更高的同时，收敛过程也比 Inception V3 更快。

2.2.5 技术

项目使用 keras 深度学习框架。Keras 是一个高层神经网络 APIKerasPython 编写而成并基 Tensorflow、Theano 以及 CNTK 后端。Keras 更是被直接引入了

TensorFlow 的核心代码库，成为 TensorFlow 官方提供的高层封装之一。

Keras 为支持快速实验而生，能够把你的 idea 迅速转换为结果， 有一下有点：

简易和快速的原型设计（keras 具有高度模块化，极简，和可扩充特性）

支持 CNN 和 RNN，或二者的结合

无缝 CPU 和 GPU 切换

2.3 基准指标

kaggle 排行榜前 10%，也就是也就是在 Public Leaderboard 上的 LogLoss 要低于 0.06127

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

3. 具体方法

3.1 数据预处理

1、训练数据集中图片大小不一，需要把图片转换成统一大小。Xception 的默认输入尺寸为 299x299， 所以所有图片在读取时转成 299x299。

2、Xception 需要将数据限定在[-1, 1]，所以通过 preprocess_input 预处理数据

3、剔除训练数据中的异常图片

通过 Xception 预测训练数据集中图片的类型，比较每个图片前 20(Top20)最可能的结果是否为猫或狗。最终选择出 67 个异常图片。

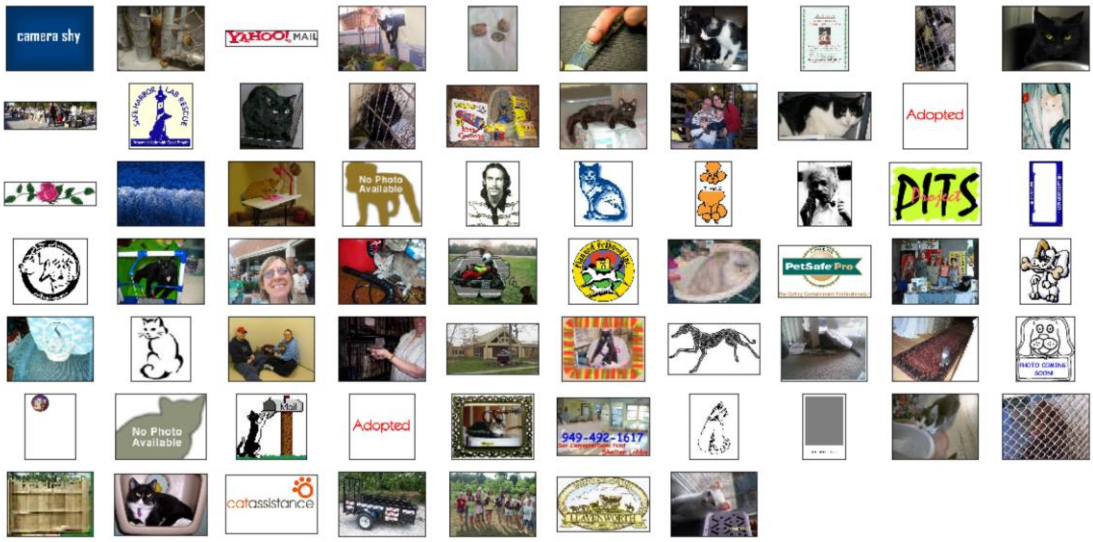


图 7 异常图片

3.2实现

1、在权重训练自 ImageNet 的 Xception 模型上添加一个全局平均池化层提取特征，添加一个全局平均池化层可以让保存特征的文件没有那么大，防止过拟合

2、构建一个包含 Dropout 层和 Dense 层的模型，Dropout 的 rate 为 0.5，Dense 的 units 为 1，激活函数为 sigmoid

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 2048)              0
_________________________________________________________________
dropout_1 (Dropout)          (None, 2048)              0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 2049
=================================================================
Total params: 2,049
Trainable params: 2,049
Non-trainable params: 0
```

3、优化器为 Adadelta，它根据渐变更新的移动窗口调整学习速率。学习率选择默认的 1.0，损失函数为 binary_crossentropy，性能指标为 accuracy。

4、训练模型，使用预先提取的特征训练模型，数据按照 8:2 的比例划分训练数

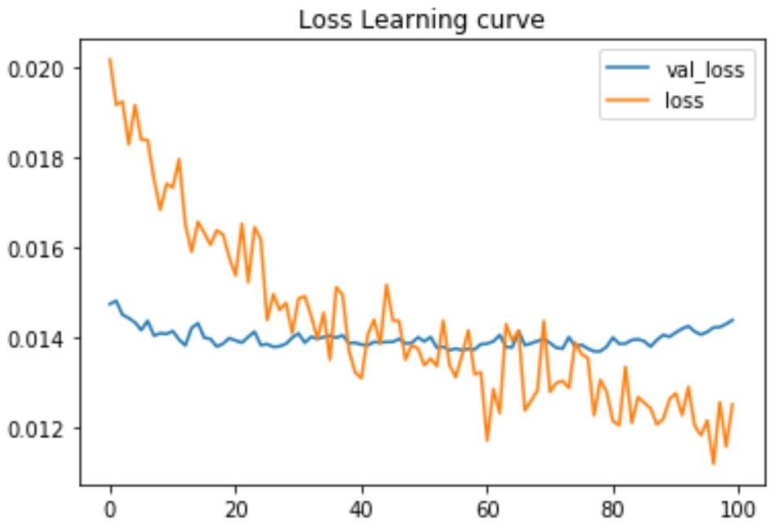
据和验证数据。总共训练 100 轮，每个梯度更新的样本数 128。

5、预处理测试集数，预测测试集生成 csv 文件，提交 Kaggle

选定 Xception 作为项目的训练模型。LogLoss 为 0.05072。

3.3 改进

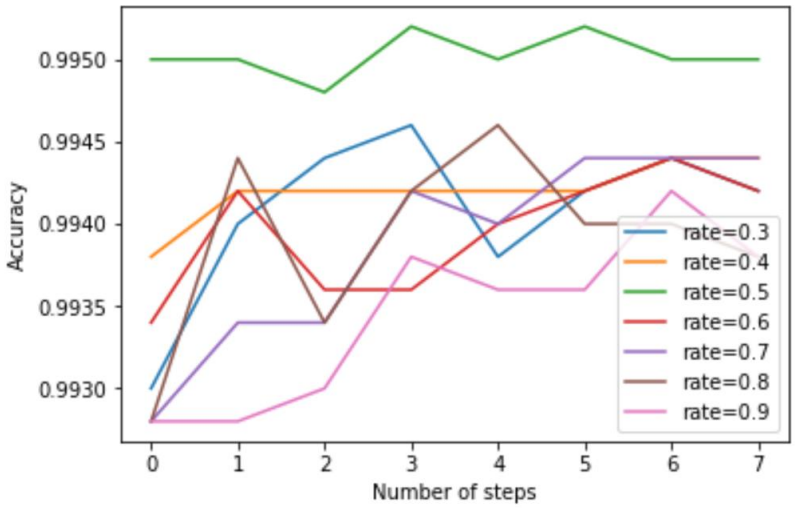
1、早期停止



从训练的学习曲线可以看出模型出现了过拟合，所以在 8epochs 处停止训练。

一方面增加训练步数会显著增加训练时间，另一方面训增加练步数会增加模型过拟合的可能性。

2、调节 Dropout 的 rate 参数，获取当 rate 为不同的值时验证集的准确率

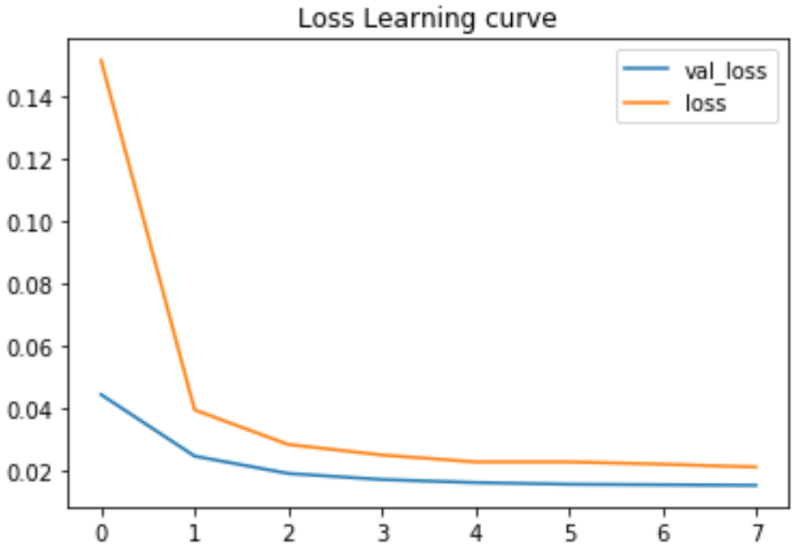


当 rate=0.5 时，验证集的准确率是最高的，所以选择 rate=0.5

3、模型预测的 label 限制在[0.005, 0.995]，模型的 LogLoss 到达 0.04107。因为对于预测正确的样本，0.995 和 1 相差无几，但是对于预测错误的样本，00.005 的差距非常大。

4. 结果

4.1 模型评价与验证

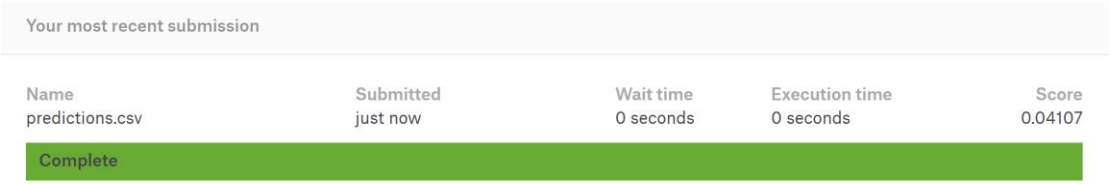


训练集分类准确率的平均水平与交叉验证集的 loss 相当，说明模型没有明显的过拟合。

4.2 合理性分析

模型最终的 LogLoss 为 0.04107，超过了最初设定的 0.06127，所以模型是合理的。

提交 Kaggle 的结果:



5. 项目结论

5.1 结果可视化

图 8

在训练完模型之后，在网上随机找了几张猫狗的图片，模型预测的结果如图 6 所见，对单个猫或狗的图片预测比较最缺，但是对同一张图片中既有猫也

有高的预测就不确定了，所以现在模型的作用还是比较单一。

5.2 对项目的思考

整个项目的解决流程：

1、数据分析：分析图片的特点，大小，分布情况，是否有离群众，错误值

2、数据预处理：把图片转成统一大小，归一化处理，剔除异常数据

3、导出特征向量：加载预处理模型，提取特征向量

5、构建模型：构建一个包含 Dropout 层和 Dense 层的模型

6、训练模型: 加载特征向量，训练模型

7、预测测试集: 使用测试数据集预测模型

整个项目最有意以的是同过自己的努力完成了猫狗分类的问题。最困难的硬件运行环境的不支持。但还是在现有条件的努力下达到了预期目标。

5.3 需要作出的改进

1、通过数据增强(Data Augmentation.) 通过平移，翻转，加噪声等方从已有数据中创造出一批"新"的数据。

2、通过 Fine-Tuning，加载预训练的模型作为初始权重中，然后训练整个模型，

3、调节模型的参数，例如：optimizer

参考文献

[1] 郑泽宇，梁博文，顾思宇. Tensorflow 实战 Google 深度学习框架. 电子工业出版社，2018.
[2] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. arXiv preprint arXiv:1603.07285, 2015.
[3] J. Jin, A. Dundar, and E. Culurciello. Flattened convolutional neural networks for feedforward acceleration. arXiv preprint arXiv:1412.5474, 2014.
[4] Francois Chollet Google, Inc.Xception: Deep Learning with Depthwise Separable Convolutions arXiv preprint arXiv:1610.02357, 2016.