Dog Vs Cat: A Deep Convolutional Neural Network Based Dog/Cat Classifier

*

项目概览

使用深度学习方法识别一张图片是猫还是狗。

- 输入：一张彩色图片
- 输出：是🐕还是🐩
- 数据集: 来自Kaggle Dogs vs. Cats Redux: Kernels Edition 的已标记的🐱/🐶图片

问题说明

以彩色图片作为输入，训练适用于区分猫狗图像的分类模型。已有大量研究证明深度卷积神经网络(Convolutional Neural Network)是解决这类问题的非常有效的方法[1, 2, 3]。因此，在本项目中也将采用类似的模型搭建卷积神经网络并进行训练和测试。

模型搭建将采用两种方式：

1. homebrew model: 从头搭建一个卷积网络
2. stand on the shoulders of giants: 充分利用现存的已经经过实战检验的预训练过的模型，在此基础之上添加相应的结构获得希望的输出

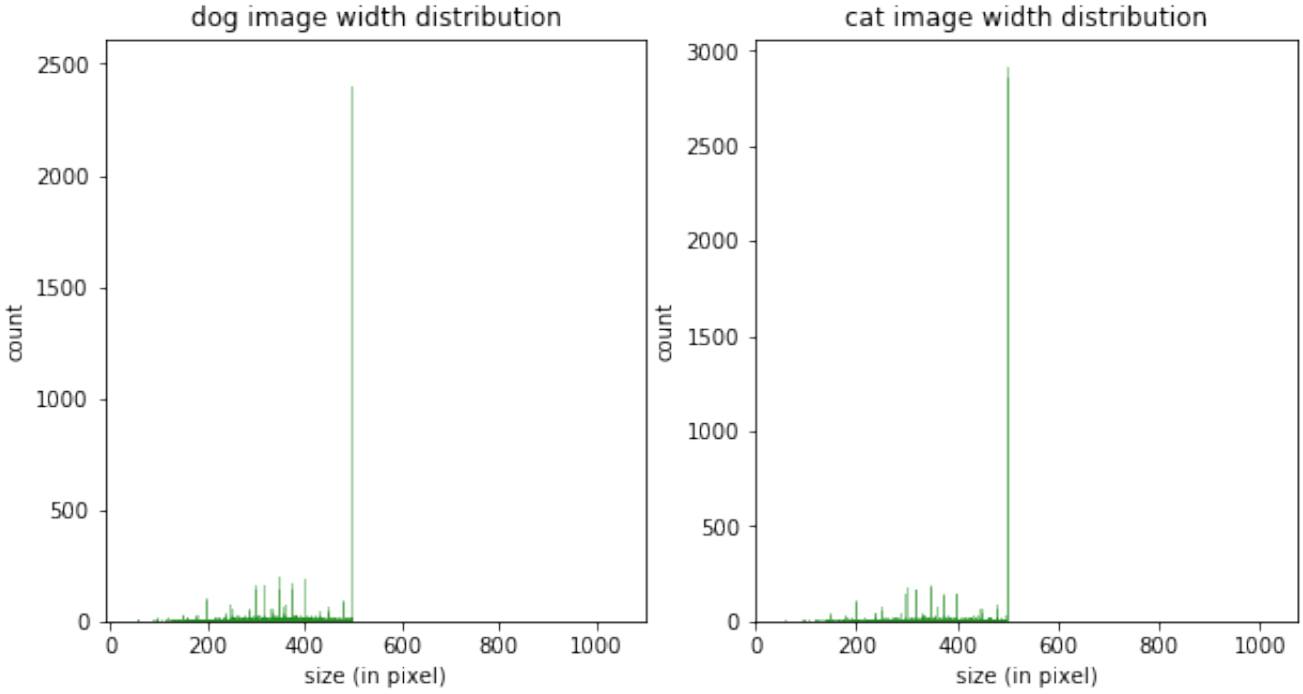
模型评价指标

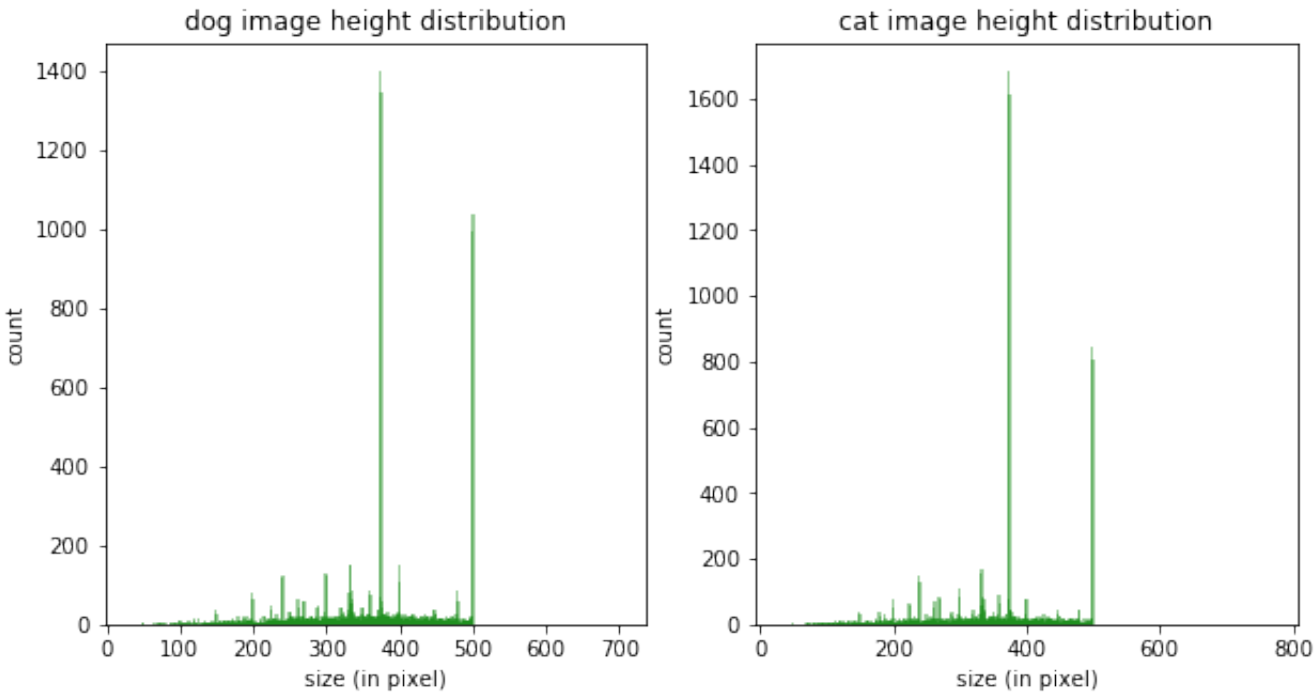
本项目中训练获得的模型将以区分猫狗的正确率(accuracy)做为评价标准。其中测试集的来源主要有两个部分: 一部分来自从原始数据集保留的部分没有用于训练的图片，按照业界通行的标准计算正确率。另一部分来自从互联网中获取的部分图片，用于对给予感性的理解。

数据研究

在建立模型前，首先对训练数据进行分析以获取数据的特性。

- 原始数据集中共有25000张已标记的图片，其中猫/狗图片各12500张，数目相当，因此不存在训练样本数量不均衡所导致的模型偏差。
- 原始数据集中图片的尺寸分布如图所示。可以看出图片的尺寸并不一致，无法直接作为模型的输入，需要进行尺寸的归一化处理。





- 人工查看了部分图片，发现训练集已经包括了多种不同背景，光照条件，动物姿态，颜色等的图片。作为一个二分类问题，我认为现有的数据集已足够模型使用，不需要进行进一步的data augmentation.
- 原始图片是通过文件名中的"dog"或者"cat"来标记🐶/🐱的，为方便之后模型的训练，这里采用了one-hot encoding的方法来将标记转换为2维的向量。

模型建立

1. homebrew model: 使用keras建立具有三层Convolutional Layer的模型，输入为(200, 200, 3)的图片, 输出为2维向量。模型的具体结构如下图所示。。

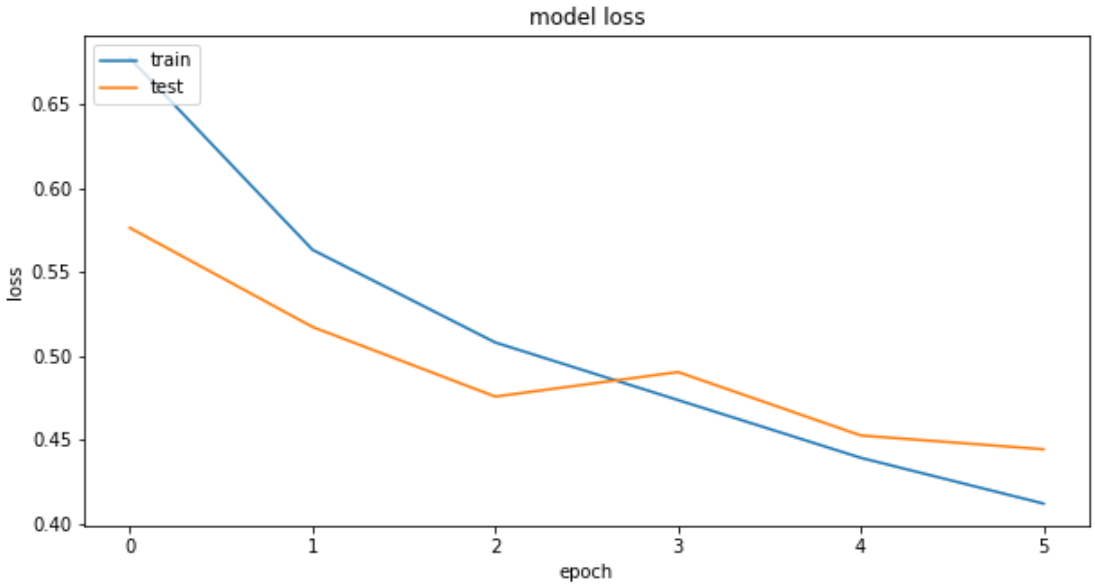
Layer (type)	Output Shape	Param #	Connected to
lambda_8 (Lambda)	(None, 224, 224, 3)	0	lambda_input_8[0][0]
convolution2d_22 (Convolution2D)	(None, 222, 222, 64)	1792	lambda_8[0][0]
activation_244 (Activation)	(None, 222, 222, 64)	0	convolution2d_22[0][0]
maxpooling2d_26 (MaxPooling2D)	(None, 111, 111, 64)	0	activation_244[0][0]
convolution2d_23 (Convolution2D)	(None, 109, 109, 32)	18464	maxpooling2d_26[0][0]
activation_245 (Activation)	(None, 109, 109, 32)	0	convolution2d_23[0][0]
maxpooling2d_27 (MaxPooling2D)	(None, 54, 54, 32)	0	activation_245[0][0]
convolution2d_24 (Convolution2D)	(None, 52, 52, 32)	9248	maxpooling2d_27[0][0]
activation_246 (Activation)	(None, 52, 52, 32)	0	convolution2d_24[0][0]
maxpooling2d_28 (MaxPooling2D)	(None, 26, 26, 32)	0	activation_246[0][0]
flatten_8 (Flatten)	(None, 21632)	0	maxpooling2d_28[0][0]
dense_15 (Dense)	(None, 16)	346128	flatten_8[0][0]
activation_247 (Activation)	(None, 16)	0	dense_15[0][0]
dropout_20 (Dropout)	(None, 16)	0	activation_247[0][0]
dense_16 (Dense)	(None, 2)	34	dropout_20[0][0]
activation_248 (Activation)	(None, 2)	0	dense_16[0][0]

2. stand on the shoulders of giants: 将带有预训练权重的ImageNet图像分类模型模型前端与自定的模型后端进行连接建立用于本项目的模型。这里主要使用了VGG16, VGG19, ResNet模型前端, 并冻结其权重来进行特征提取，之后加入自定的几个全连接层用于本项目中的图片分类。

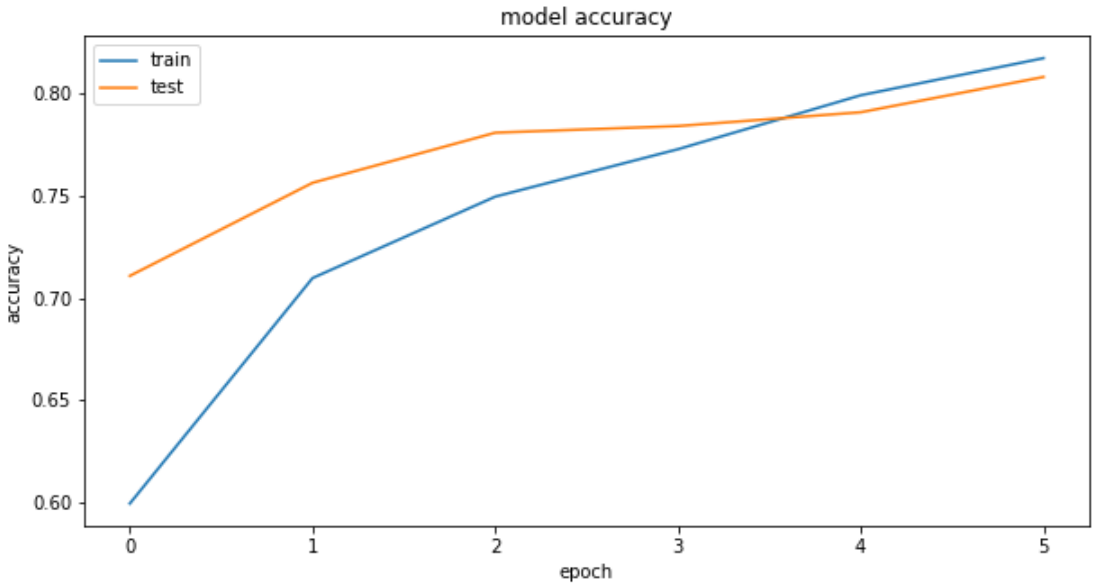
模型训练与评估

因所需的计算量较大, 本项目的模型训练使用了AWS p2.xlarge instance, 并参考了这篇文档进行了配置。关于数据预处理,可视化和模型搭建的代码可以参阅这个Jupyter Notebook。 原始训练数据按照8:2的比例划分为训练验证集和测试集，之后训练验证集同样按照8:2的比例划分为训练集和验证集。 经过尝试发现，对于hombrew model使用6个epoch训练，而对于采用带有预训练权重的model使用3个epoch训练就可以达到比较好的效果。模型训练中loss和accuracy随epoch变化的曲线罗列如下:

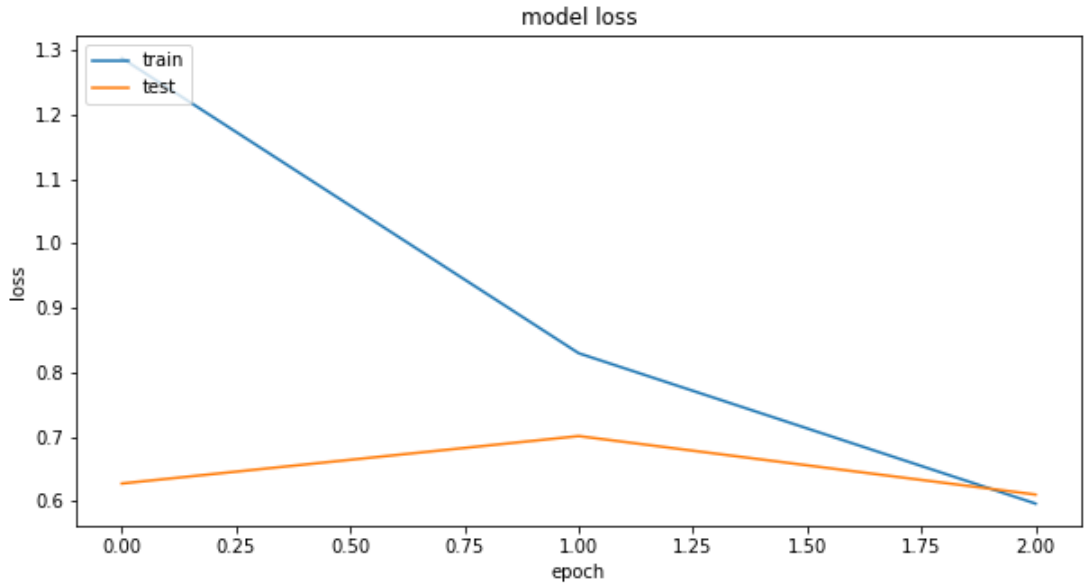
- **homebrew model**
    - loss



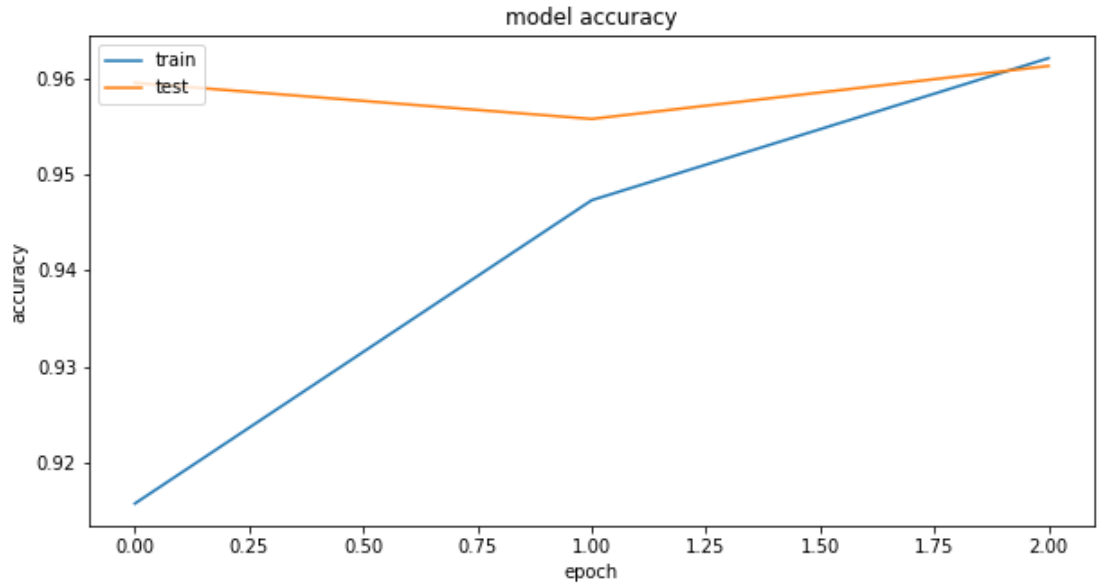
- accuracy (final accuracy = 0.80)



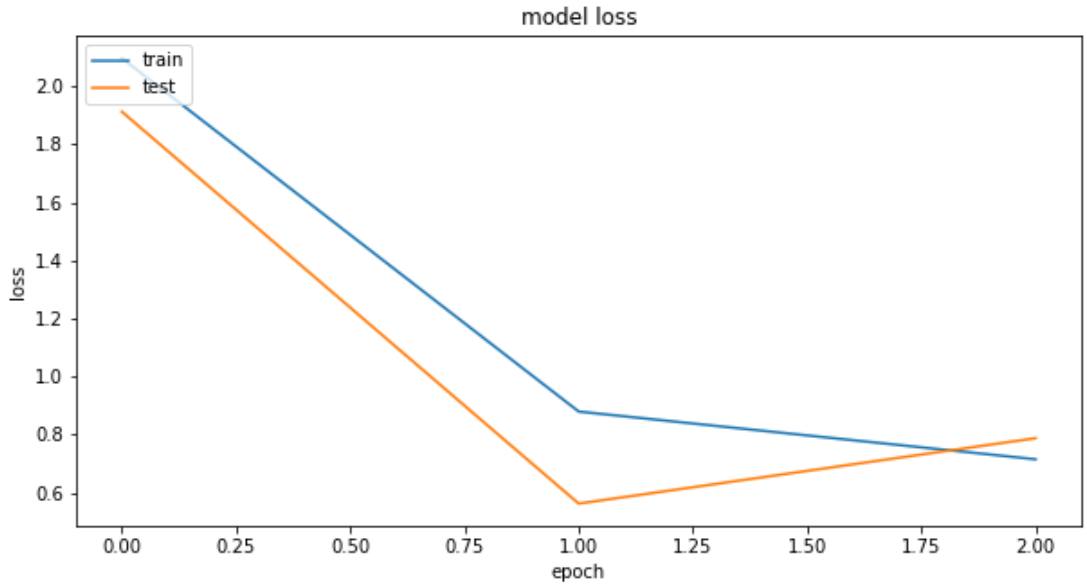
- **VGG16 model**
  - loss



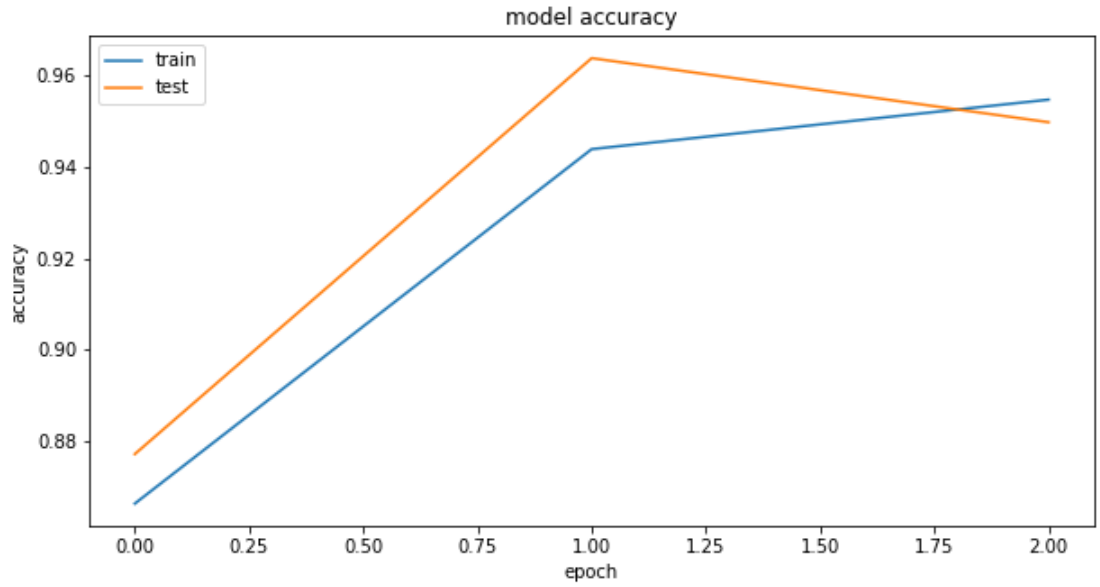
- accuracy (final accuracy = 0.97)



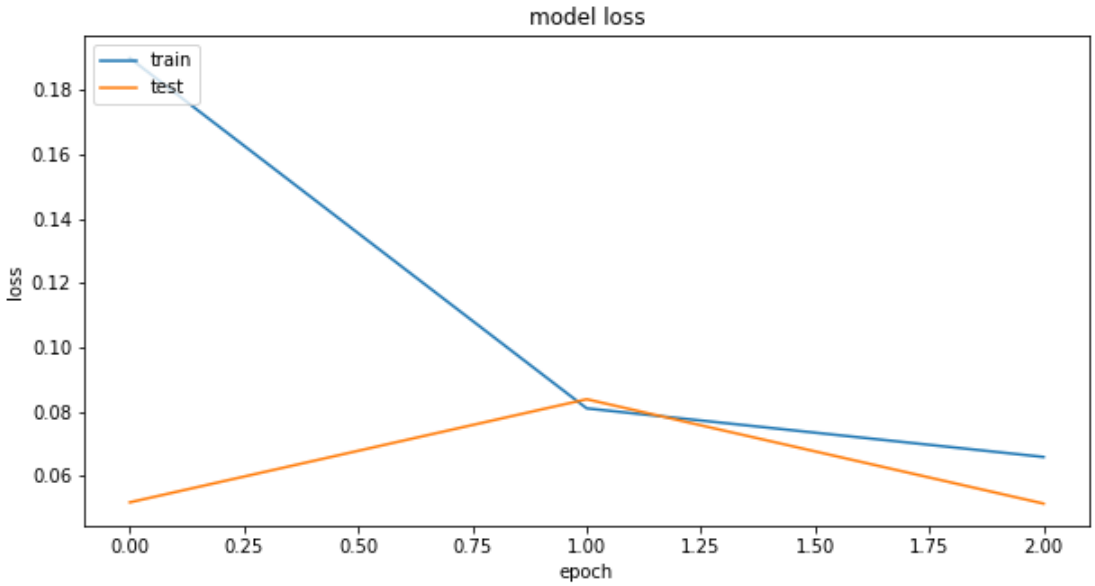
- **VGG19 model**
  - loss



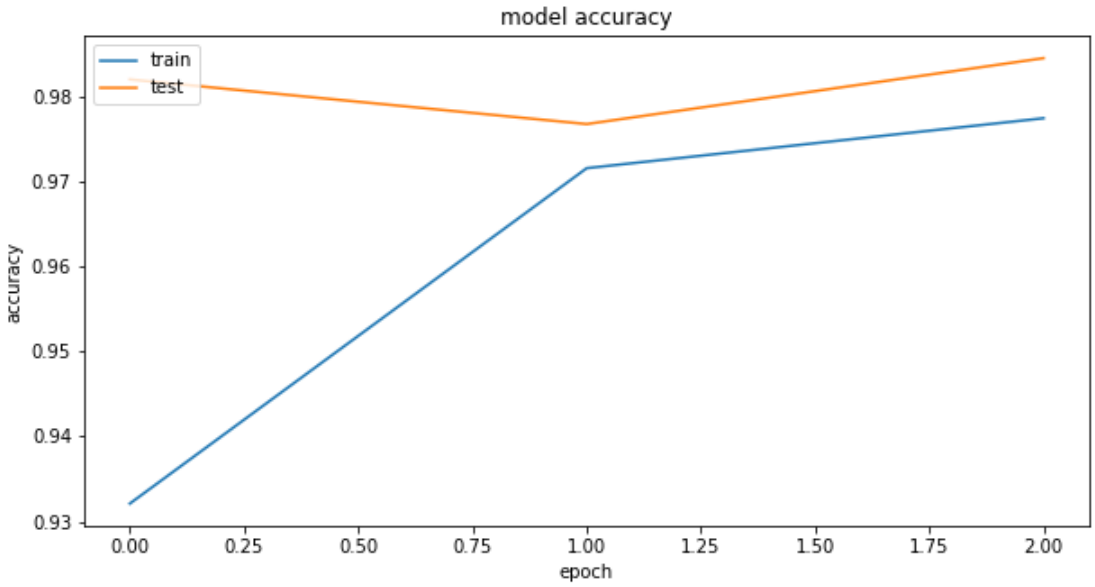
- accuracy (final accuracy = 0.95)



- **ResNet model**
  - loss



- accuracy (final accuracy = 0.99)



经过训练，homebrew model的accuracy达到了80%以上，采用带有预训练权重的model的accuracy均在95%以上。

同时，为对模型的实际预测能力有一个直观的认识，我们从也通过搜索引擎从互联网上获取了一些图片用于测试，以下是部分测试结果。测试代码可以参阅这个Jupyter Notebook

cat1.jpg
ResNet predction: cat raw output: [1.00000000e+00   1.29284090e-18]
VGG16 predction: cat raw output: [1. 0.]
VGG19 predction: cat raw output: [1. 0.]
homebrew predction: cat raw output: [0.6765725 0.3234275]

cat2.jpg
ResNet predction: cat raw output: [1.00000000e+00   3.31773675e-09]
VGG16 predction: cat raw output: [1. 0.]
VGG19 predction: cat raw output: [1. 0.]
homebrew predction: cat raw output: [0.80154902 0.19845101]

cat3.jpg
ResNet predction: cat raw output: [9.99992132e-01   7.89054229e-06]
VGG16 predction: cat raw output: [1. 0.]
VGG19 predction: cat raw output: [1. 0.]
homebrew predction: dog raw output: [0.36286509 0.63713491]

cat4.jpg
ResNet predction: cat raw output: [1.00000000e+00   1.47028159e-08]
VGG16 predction: cat raw output: [1. 0.]
VGG19 predction: cat raw output: [1. 0.]
homebrew predction: dog raw output: [0.04296251 0.95703751]

cat5.jpg
ResNet predction: cat raw output: [1.00000000e+00   2.34724906e-09]
VGG16 predction: cat raw output: [1. 0.]
VGG19 predction: cat raw output: [1. 0.]
homebrew predction: dog raw output: [0.29939982 0.70060015]

dog1.jpg
ResNet predction: dog raw output: [1.53029614e-14   1.00000000e+00]
VGG16 predction: dog raw output: [0. 1.]
VGG19 predction: dog raw output: [0. 1.]
homebrew predction: dog raw output: [8.42586101e-04   9.99157429e-01]

dog2.jpg
ResNet predction: dog raw output: [2.76792996e-17   1.00000000e+00]
VGG16 predction: dog raw output: [0. 1.]
VGG19 predction: dog raw output: [0. 1.]
homebrew predction: dog raw output: [4.94772044e-04   9.99505162e-01]

dog3.jpg
ResNet predction: dog raw output: [2.16073886e-06   9.99997854e-01]
VGG16 predction: dog raw output: [0. 1.]
VGG19 predction: dog raw output: [0. 1.]
homebrew predction: dog raw output: [0.06887888 0.93112111]

dog4.jpg
ResNet predction: dog raw output: [0.01372575 0.9862743]
VGG16 predction: dog raw output: [0. 1.]
VGG19 predction: dog raw output: [0. 1.]
homebrew predction: dog raw output: [0.45201397 0.54798609]

dog5.jpg
ResNet predction: dog raw output: [2.38115408e-13   1.00000000e+00]
VGG16 predction: dog raw output: [0. 1.]
VGG19 predction: dog raw output: [0. 1.]
homebrew predction: dog raw output: [0.37651491 0.62348509]

dog6.jpg
ResNet predction: dog raw output: [0.01510979 0.98489016]
VGG16 predction: dog raw output: [0. 1.]
VGG19 predction: dog raw output: [0. 1.]
homebrew predction: dog raw output: [0.43500969 0.56499034]

可以看出采用带有预训练权重的model的表现十分出色，对所有的测试图片都给出了正确的预测结果。而homebrew model在cat4和cat5图片上给出了错误的结果。因为homebrew model的结构相对简单，这一结果也并没有出乎意料。

总结

在本项目中，我们使用了深度学习的方法来处理猫狗图片分类的问题，采用两种不同的路径搭建,训练并验证了CNN模型，最终都获得了较高的识别率(>80%)。相较而言，homebrew model还存在有较大的 的改进空间，需要在今后继续研究。

*** title image source: https://www.pinterest.com/pin/365636063472803484/**