

机器学习纳米工程师毕业项目

猫狗大战

姓 名： 吴典校

日 期： 2018 年 02 月 27 日

目录

一、项目背景.....	3
二、问题陈述.....	3
三、数据或输入.....	3
3.1 数据描述及可视化.....	3
3.2 数据预处理.....	5
四、评估指标.....	6
五、解决方法描述.....	6
5.1 ResNet50	6
5.2 InceptionV3.....	7
5.3 InceptionResNetV2	9
5.4 Xception	9
六、基准模型.....	11
七、项目设计.....	11
7.1 数据预处理.....	11
7.2 模型训练.....	11
7.2.1 ResNet50	12
7.2.2 InceptionV3	12
7.2.3 Xception	13
7.2.4 Inception_Resnet_v2.....	13
7.3 模型改进.....	14
八、结果分析.....	15
九、主要参考文献.....	16

一、项目背景

猫狗大战是 Kaggle 上的一个关于在给定的图像中进行猫狗分类问题的分类问题，属于典型的图像识别问题。该项目所选择的数据集是 Kaggle 所提供的数据集，训练集是由 12500 张猫的图片 and 12500 张狗的图片组成，测试集是由 12500 张未标记（labeled）的图片组成。

深度学习的思维范式是人工神经网络，需要追溯到二十世纪四十至五十年代当时广为人知的控制论（cybernetics），所学的模型是最简单的线性加权，其应用领域极为受限，最为著名的是不能成立“异或”问题。

直到 80 年代，David Rumelhar 和 Geoffery E. Hinton 等人提出了反向传播算法，解决了两层神经网络所需要的复杂计算量问题，同时解决了单层神经网络无法处理“异或问题”，但受限于数据获取的瓶颈，只能在中小规模数据上训练，且过拟合极大困扰着神经网络算法。

随着软件算法和硬件性能的不断优化，直到 2006 年，Geoffery E. Hinton 等发表文章提出一种称为“深度置信网络”（deep belief network）的神经网络模型可通过逐层预训练的方式有效完成模型训练过程。

得益于软件算法、数据量剧增和硬件配置的发展，图像识别在静态平面图片识别上已十分成熟，但对于空间几何识别仍需要更多开发人员进行研究和开发，最终运用到实际运用中，对人类的发展产生深远的影响。

二、问题陈述

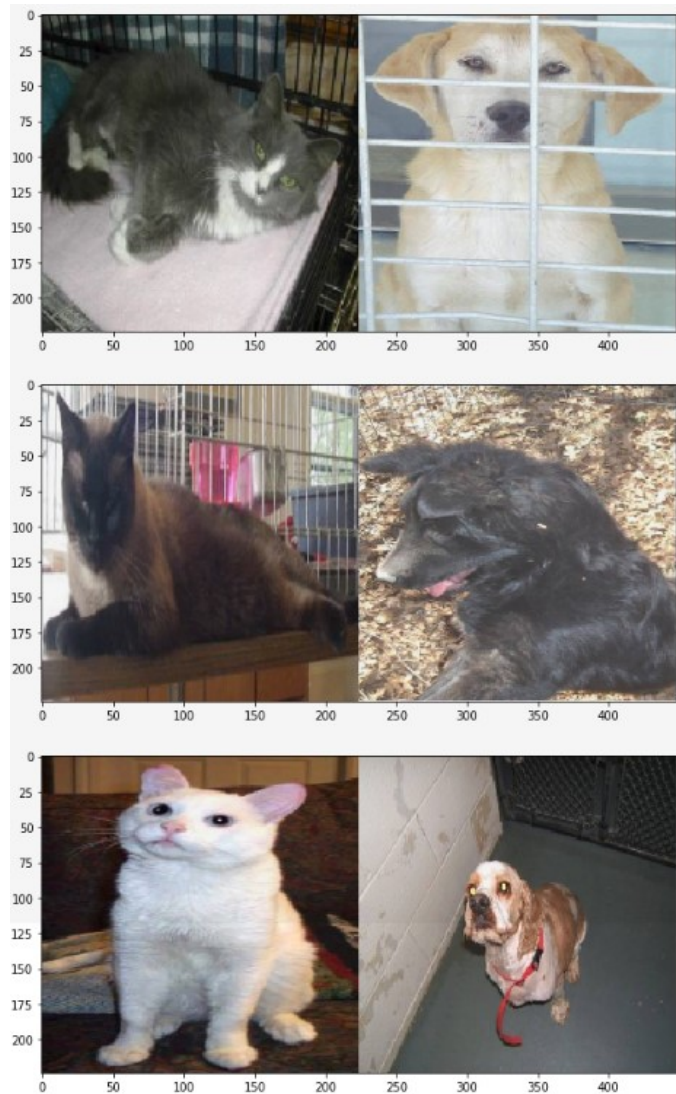
猫狗大战要解决的根本问题是图像分类，其中由于图片的分辨率差异大，尺寸大小不一，同时猫狗的种类众多，颜色各异，背景环境不同，因此该数据集的复杂程度亦增加了模型搭建的难度。

解决图像识别问题存在以下方法：KNN、SVM、BP 神经网络、CNN 和迁移学习，为了得到性能更优的模型，本项目仅讨论 CNN 与迁移学习。

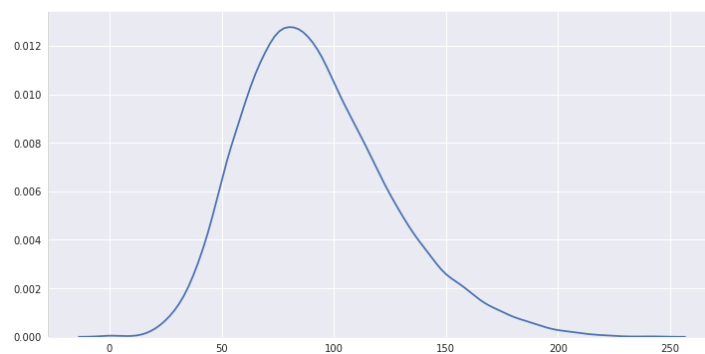
三、数据或输入

3.1 数据描述及可视化

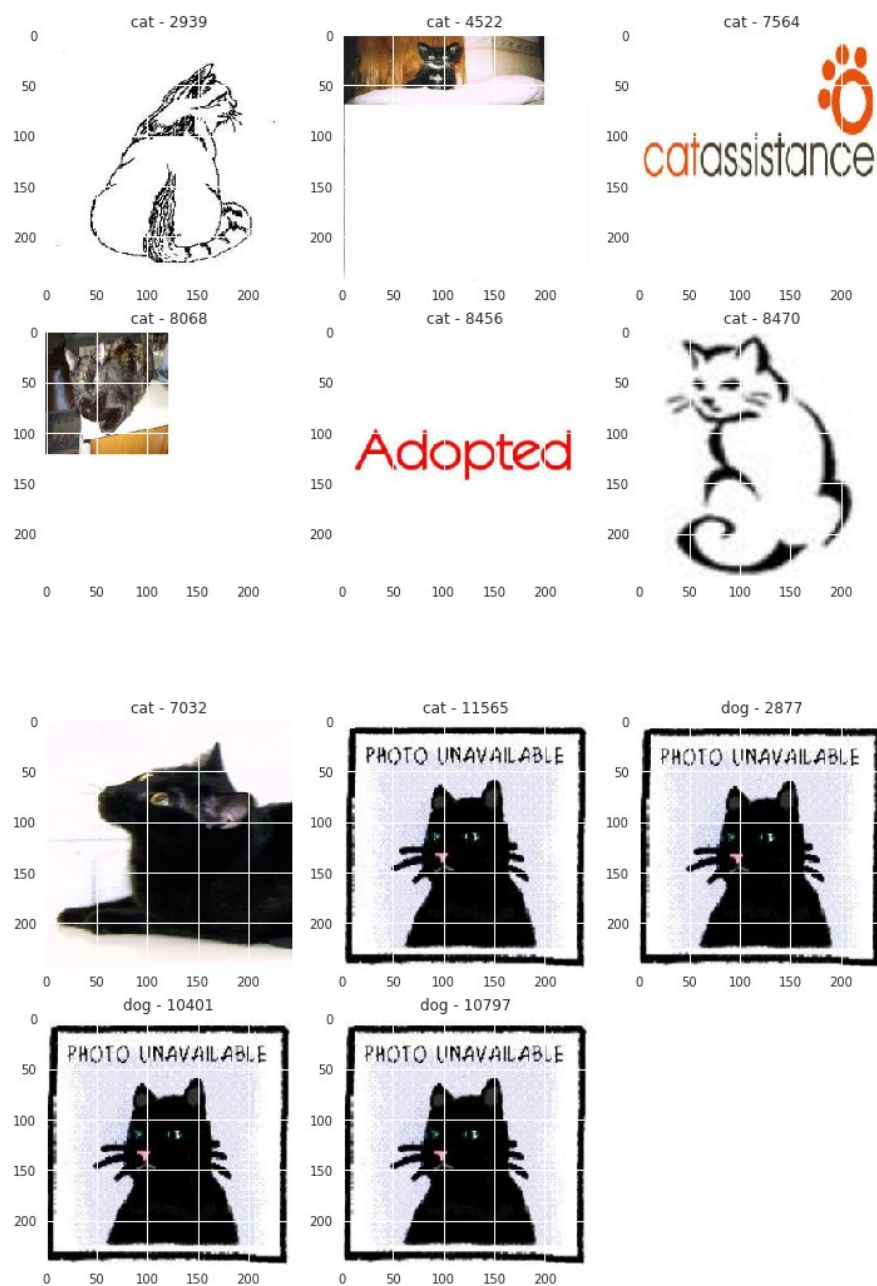
该项目所选择的数据集是 Kaggle 所提供的数据集，训练集是由已标记（labeled）的 12500 张猫的图片 and 12500 张狗的图片组成，测试集是由 12500 张未标记（Not labeled）的图片组成。



同时对数据异常值进行监测，通过对图片三个维度数值的乘积进行分析，得到其分布图，即一个正态分布图：



显示值小于 10 以及大于 235 的图片，其图片如下。可以看出该训练集中仍存在部分异常值。考虑到查找出来的异常值数量为 16 张，仅占训练集的 0.064%，可以忽略不计。



3.2 数据预处理

首先使用 Opencv 库中的 `imread()` 函数读取训练集数据和测试集数据，由于基于现有模型进行计算，每种模型在读取数据时应满足固定的分辨率要求，如 Xception 和 InceptionV3 模型要求的输入分辨率为 299×299 ，ResNet、DenseNet 模型要求的输入分辨率为 244×244 。为了保证缩放后的图片不会出现严重变形或者是细节丢失的情况，为此使用 `resize` 函数将照片缩放到模型所要求的输入尺寸。。

同时，为了保证训练过程中调节参数，将原始训练数据拆分为照片总量为 20000 张的训练集，总量为 5000 张（猫狗图像各 2500 张）作为验证集。由于验证集仅是用来调节模型中的超参数，因此总量为 5000 张的验证集足以将模型参数调整至最佳。同时在分割训练集和验证集时使用 `Shuffle`

属性，将数据集打乱，防止分割时猫狗分布不均。

四、评估指标

对于分类问题，由于在 CNN 模型和迁移模型中都使用梯度下降的实现方法，即通过求评估函数的梯度以不断更新神经网络中的权值，但在求梯度下降最优解时，必须保证评估函数为凸函数，故使用 Loss Function，也称为 cross-entropy 检测算法运行情况，公式如下：

$$L(\hat{y}, y) = -(y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y}))$$

其中， y 为测试集中的 y 值， \hat{y} 为模型所预测的 y 值。

对运用于更新权重 w 、偏差 b 的模型中，使用的评估函数为：

$$J(w, b) = -\frac{1}{m} \cdot \sum_{i=1}^m (y^{[i]} \cdot \log(\hat{y}^{[i]}) + (1 - y^{[i]}) \cdot \log(1 - \hat{y}^{[i]}))$$

其中， $y^{[i]}$ 为测试集中第 i 条数据的 y 值， $\hat{y}^{[i]}$ 为第 i 条数据的预测值， m 为测试集数据总个数。

五、解决方法描述

CNN 卷积神经网络是一种层次模型，其基本结构为输入原始数据-卷积操作-汇合（pooling）操作-非线性激活函数（ReLU）映射、分类（Classification）等一系列操作的层层堆叠，其核心是通过使用共享权重的卷积层替代了一般的全连接层。

迁移学习则是将已经训练的卷积神经网络模型迁移到其他数据集，以节省训练时间。现阶段对于 ImageNet 数据已经存在诸多预训练模型，本项目中使用 ResNet50、InceptionV3、Xception 和 Inception Resnet v2 等四个模型进行训练。

迁移学习有两种策略，一是微调（fine-tuning），即训练已有神经网络中的所有层；二是冻结和训练（freeze and train），即将前面一定数量的层冻结后对其他层进行微调，一般情况下只训练最后一层。本项目中猫狗数据集与 ImageNet 接近，使用微调或者冻结均可，故在本项目中同时使用该两种方法并进行对比。

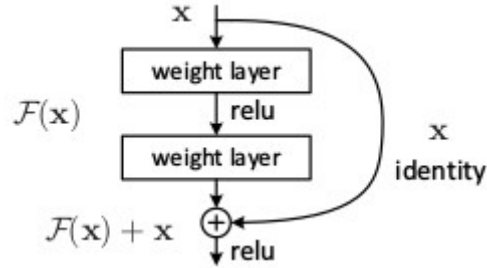
以下对 ResNet50、InceptionV3、Xception 和 Inception Resnet v2 五个模型进行简单的介绍：

5.1 ResNet50

ResNet50 模型是 2015 年 ILSVRC & COCO competitions 的 ImageNet Detection, ImageNet localization, COCO detection 以及 COCO segmentation 项目中获得第一名的模型。该模型是由微软的

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun 所提出。解决了更深的模型反而比更浅的模型效果更差的问题。

其主要改变是原来深度学习神经网络的基础上，对于两层以上的 block 块，增加 shortcut connection，即是 $y=x$ ，形成一个 block，可以防止在参数传递过程中，若出现权重为零，会把输入值置零的情况，其 Block 块如下图所示。



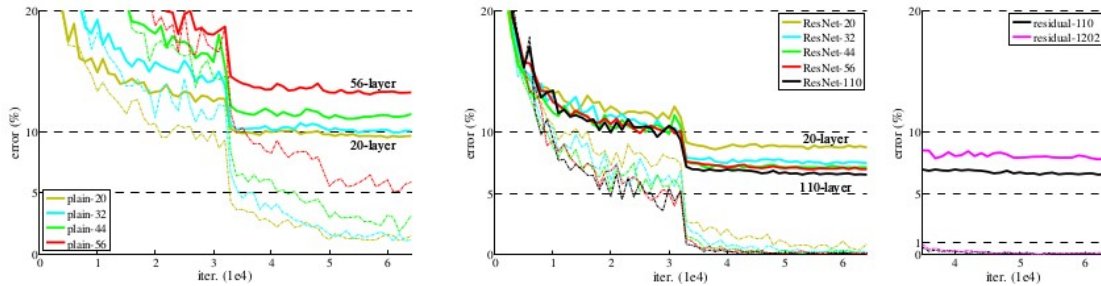
以上模块有两层 Relu 非线性函数层，首层表达式为：

$$\mathcal{F} = W_2 \sigma(W_1 \mathbf{x})$$

第二层加入 shortcut，获得输出：

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}.$$

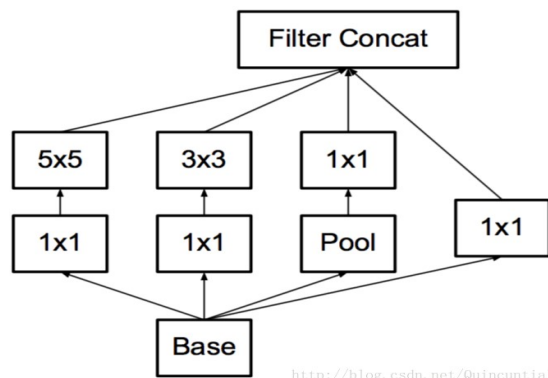
在对 CIFAR-10 数据集测试中，其测试图如下：



其中左边是传统神经网络模型，可以看出 56-layers 的模型是比 20-layers 的模型训练效果更差。中间 ResNet 模型训练结果，可以看出层数原告，其训练效果更好，而且普遍比传统神经网络的训练效果好。右图是层数为 110 和 1202 的 ResNet 模型，其中 110-layers-ResNet 模型是比 1202 的 ResNet 模型训练效果要好。

5.2 InceptionV3

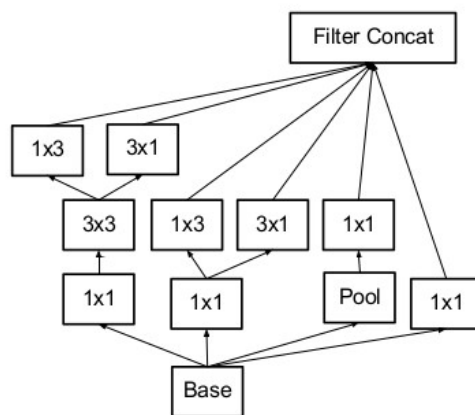
InceptionV3 是 2015 年由 Google 团队提出的改进模型，在原有 Inception 模型上进行优化，进一步减少计算成本。原有的 Inception 网络模型结构的基础模块如下所示：



Inception 网络模型中的基础模块是将 1×1 , 3×3 , 5×5 的卷积层和 Pooling layer 堆叠在一起, 增加了网络的宽度以及对尺度的适应性。

然而, 5×5 卷积核所需的计算量太大, 故 Inception 网络模型将 5×5 卷积核由两个 3×3 卷积核替换, 可减少计算成本是全卷积的, 每个权重对应每个激活的乘法, 降低计算成本会导致参数量降低, 从而加快训练,

为了进一步简化 3×3 卷积层序列, 可以通过使用非对称个卷积, 即 $n \times 1$ 。例如使用 3×1 卷积后接一个 1×3 卷积, 其效果相当于 3×3 卷积核。其网络结构如下:



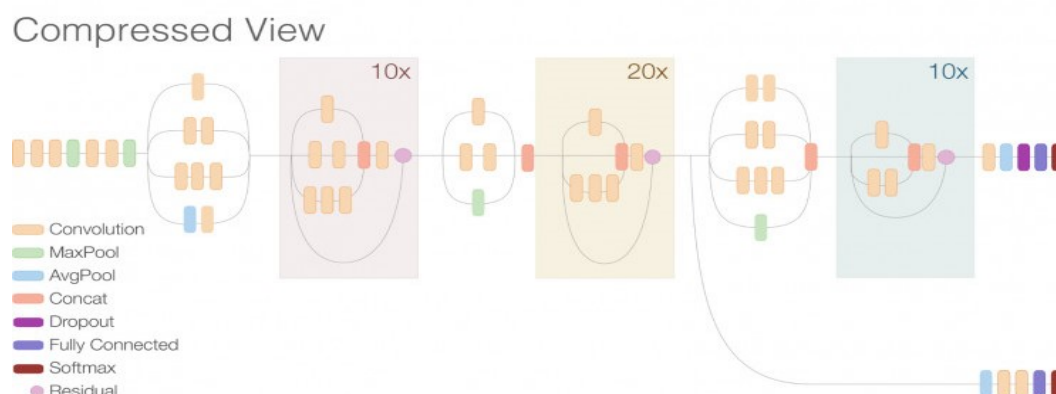
对于输入和 Filters 数量相等, 对于相同数量的输出 Filters, 该方案可以便宜 33%。理论上, 可以通过 $1 \times n$ 卷积后接一个 $n \times 1$ 卷积, 且随着 n 增长, 计算成本节省显著增加。但实际上该分解在靠前的层工作效果不佳, 但是到中等网络尺寸 $m \in (12, 20)$, 其效果非常好。对于在 17×17 网络中, 可以选择 $n=7$ 。

通过在 ILSVRC-2012 验证集上进行评估, 测试结果如下所示。可以看出该模型是比传统深度神经网络模型处理的效果会更好。

Network	Models Evaluated	Crops Evaluated	Top-1 Error	Top-5 Error
VGGNet [18]	2	-	23.7%	6.8%
GoogLeNet [20]	7	144	-	6.67%
PRReLU [6]	-	-	-	4.94%
BN-Inception [7]	6	144	20.1%	4.9%
Inception-v3	4	144	17.2 %	3.58% *

5.3 InceptionResNetV2

InceptionResNetV2 模型是 2016 年 Google 对 Inception V3 模型的另一种改进,即是在 Inception V3 模型上增加一个 shortcut, 其模型结构如下所示。其模型深度比 Inception V3 模型更深, 同时重复的残差区块被压缩简化, 即包含更少的并行塔 (Parallel towers)



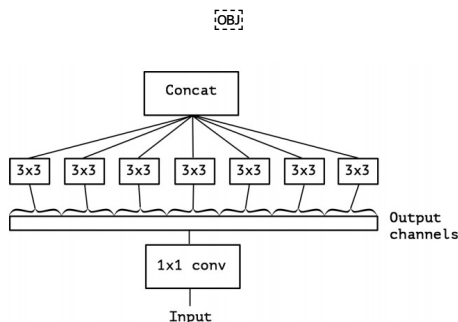
通过在 ILSVRC-2012 验证集上进行评估, 其训练结果亦优于 Inception V3 以及 ResNet151, 其测试结果如下所示: 。

Network	Crops	Top-1 Error	Top-5 Error
ResNet-151 [5]	10	21.4%	5.7%
Inception-v3 [15]	12	19.8%	4.6%
Inception-ResNet-v1	12	19.8%	4.6%
Inception-v4	12	18.7%	4.2%
Inception-ResNet-v2	12	18.7%	4.1%

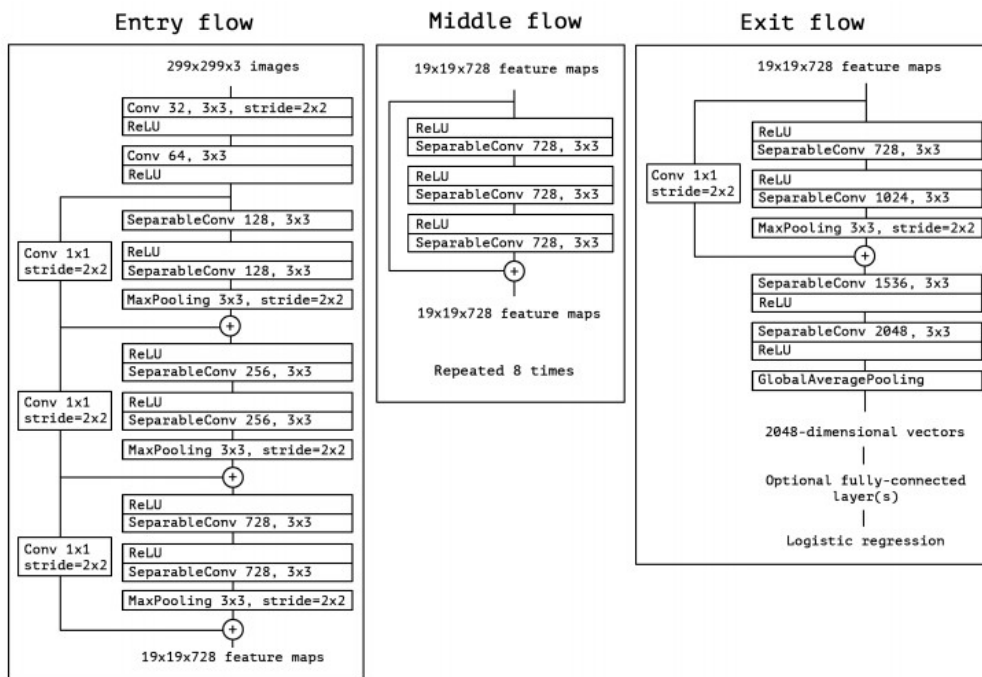
5.4 Xception

Xception 模型是谷歌对 Inception V3 模型的另一种改进。Inception V3 模型通过对 1×1 、 3×3 、 5×5 和 Pooling 结构, 以更少的参数和更少的计算成本以学习更丰富的特征。Inception 模块首先用 1×1 的卷积核将特征图每个通道映射到新的空间, 连接各个通道; 再通过 3×3 、 5×5 卷积核进行卷积, 连

接各层权重和各个通道。但此时各层权重的连接和各个通道的连接并没有完全分离，若将其权重连接性和通道连接性完全分离开，效果会不会更为出色。基于该考虑，Xception 尝试将所有 3×3 、 5×5 卷积核都作用在只有一个通道的特征图上。其结构如下：



该模型是通过多个 3×3 卷积核拼接，使其与 1×1 卷积核输出通道相等，此时每个 3×3 卷积核仅作用于包含一个通道的特征图上，即 Extream Inception 模块。其完整网络结构，如下图所示。



分别在 ImageNet 和 JFT 上进行训练，并分别在 ImageNet 和 FastEval14k 上进行测试，其结果如下所示。可以看出 Xception 的效果相比于 Inception V3 有所上升。

	Top-1 accuracy	Top-5 accuracy
VGG-16	0.715	0.901
ResNet-152	0.770	0.933
Inception V3	0.782	0.941
Xception	0.790	0.945

六、基准模型

Kaggle 上 Public Leaderboard 是对测试集数据进行评估，评估函数为 Log Loss 函数，本项目基准模型目标是，测试集的分类结果可以排在 Public Leaderboard 上的前 100 名，即 Log loss 小于 0.05629。

七、项目设计

7.1 数据预处理

首先使用 Opencv 库中的 `imread()` 函数读取训练集数据和测试集数据，由于基于现有模型进行计算，每种模型在读取数据时应满足固定的分辨率要求，如 Xception 和 InceptionV3 模型要求的输入分辨率为 299×299 ，ResNet、DenseNet 模型要求的输入分辨率为 244×244 。为了保证缩放后的图片不会出现严重变形或者是细节丢失的情况，为此使用 `resize` 函数将照片缩放到模型所要求的输入尺寸。

同时，为了保证训练过程中调节参数，将原始训练数据拆分为照片总量为 20000 张的训练集，总量为 5000 张（猫狗图像各 2500 张）作为验证集。由于验证集仅是用来调节模型中的超参数，因此总量为 5000 张的验证集足以将模型参数调整至最佳。同时在分割训练集和验证集时使用 `Shuffle` 属性，将数据集打乱，防止分割时猫狗分布不均。

在生成模型时，使用 Lambda 层引用模型中的 `preprocess_input` 模块对载入模型的 Tensor 进行预处理，其中每种模型的 `preprocess_input` 的作用不一样，如 ResNet50 模型的 `preprocess_input` 是将 RGB 每个色彩通道减去 ImageNet 数据集相应通道的均值（103.939, 116.779, 123.68），并将每个色彩通道从 RGB 调整为 BGR；而 InceptionV3 和 Xception 模型 `preprocess_input` 是将数据从（0, 255）缩放到（-1,1）区间内。如 Resnet50 模型数据预处理代码如下所示。

```
input_x = Input((244,244,3))

input_x = Lambda(resnet50.preprocess_input)(input_y)

base_model_VGG = resnet50.Resnet50(weights = 'imagenet', include_top = False, input_tensor
= input_x)
```

7.2 模型训练

利用 Xception、InceptionV3、ResNet50、DenseNet121 模型进行建模，用 sigmoid 激活函数的层替代原有的最后一层。同时在训练过程中引入 Early Stopping 以避免过拟合，在 `val_loss` 连续 5

个 epochs 没有优化时停止训练。训练环境是在 Google Platform 上进行训练，使用的显卡是 NVIDIA Tesla P100。

对 Xception、InceptionV3、ResNet50、DenseNet121 模型超参数数值及其测试结果设置如下：

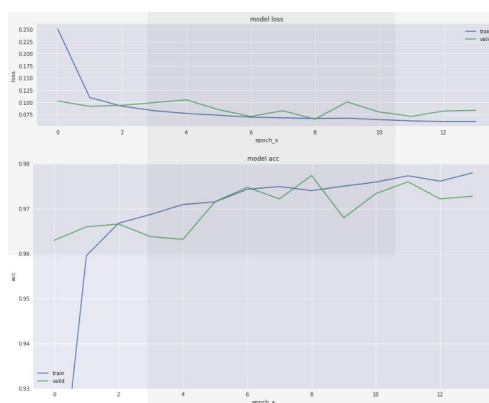
Model	Epochs	Batch_size	Optimizer	Dropout	锁层	Test Score
ResNet50	15	256	adam	0.4	True	0.08847
InceptionV3	20	64	Sgd(lr=0.001,momentum=,0.9decay = 1e-3)	0.5	False	0.04359
Xception	10	32	Sgd(lr=0.001,momentum=,0.9decay = 1e-3)	0.5	False	0.04151
Inception_Resnet_v2	20	32	Sgd(lr=0.001,decay = 1e-3)	0.5	False	0.03942

7.2.1 ResNet50

设定其超参数数值如下：

Epochs	Batch_size	Optimizer	Dropout	锁层
15	256	adam	0.4	True

其训练结果如下：



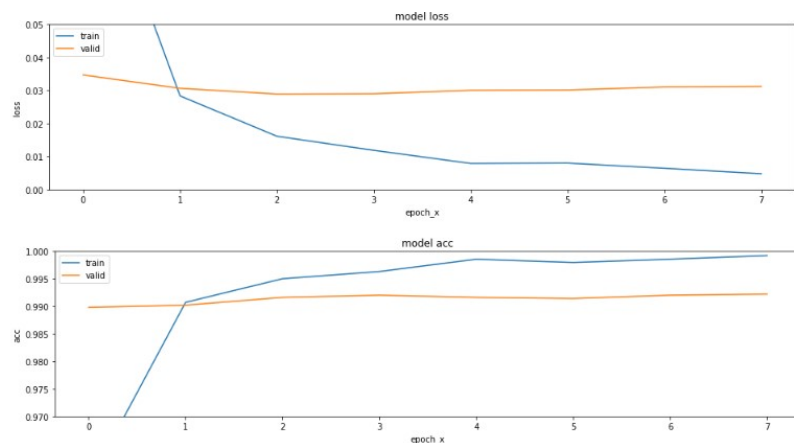
ResNet50 模型对测试集进行预测，并将结果上传至 Kaggle，获得得分为 0.08847。

7.2.2 InceptionV3

设定其超参数数值如下：

Epochs	Batch_size	Optimizer	Dropout	锁层
20	64	Sgd(lr=0.001,momentum=,0.9decay = 1e-3)	0.5	False

其训练结果可视化如下：



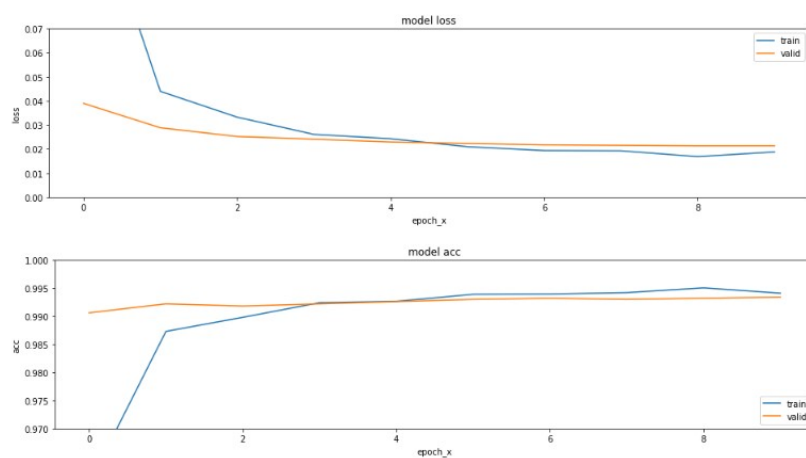
InceptionV3 模型对测试集进行预测，并将结果上传至 Kaggle，获得得分为 0.04359。

7.2.3 Xception

设定其超参数值如下：

Epochs	Batch_size	Optimizer	Dropout	锁层
10	32	Sgd(lr=0.001,momentum=,0.9decay = 1e-3)	0.5	False

其训练结果可视化如下：



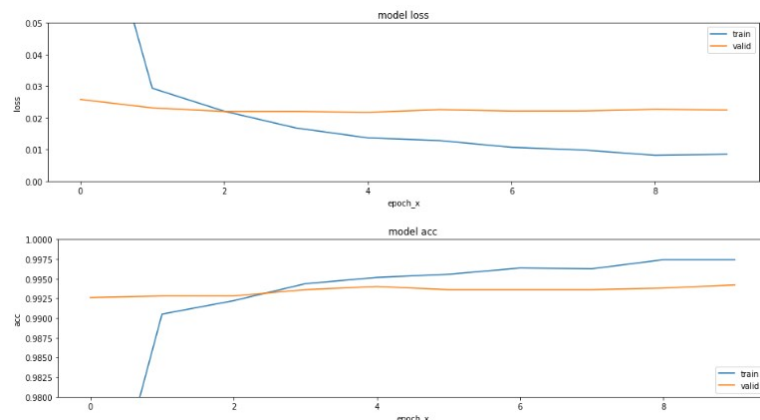
Xception 模型对测试集进行预测，并将结果上传至 Kaggle，获得得分为 0.04151。

7.2.4 Inception_Resnet_v2

设定其超参数值如下：

Epochs	Batch_size	Optimizer	Dropout	锁层
20	32	Sgd(lr=0.001,decay = 1e-3)	0.5	False

其训练结果可视化如下：



模型对测试集进行预测，并将结果上传至 Kaggle，获得得分为 0.03942。

鉴于 Inception_Resnet_v2 模型训练效果最佳，因此对 Inception_Resnet_v2 模型进行调参，调节参数为 Dropout，调节效果如下：

Dropout	accuracy	loss	val-acc	val-loss	Test Score
0.2	0.9978	0.0079	0.9936	0.0220	0.03952
0.5	0.9982	0.0065	0.9940	0.0223	0.03942
0.8	0.9974	0.0085	0.9942	0.0225	0.03955
1.0	0.9977	0.0096	0.9977	0.0250	0.04005

由以上结果可以，看出在 Dropout 设置为 0.5 时训练效果最好，故将 Dropout 设置为 0.5。

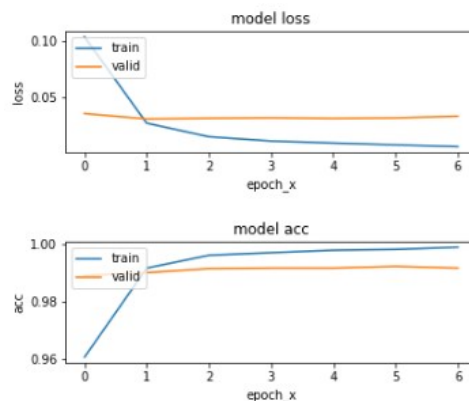
7.3 模型改进

由上可知，InceptionV3、Xception 和 Inception_Resnet_v2 三个模型得分最高，提取该三个测试集特征进行堆叠，获得融合后的特征集合。

设定其超参数值如下：

Epochs	Batch_size	Optimizer	Dropout	锁层
20	64	Sgd(lr=0.001, decay = 1e-3)	0.5	False

其训练结果可视化如下：

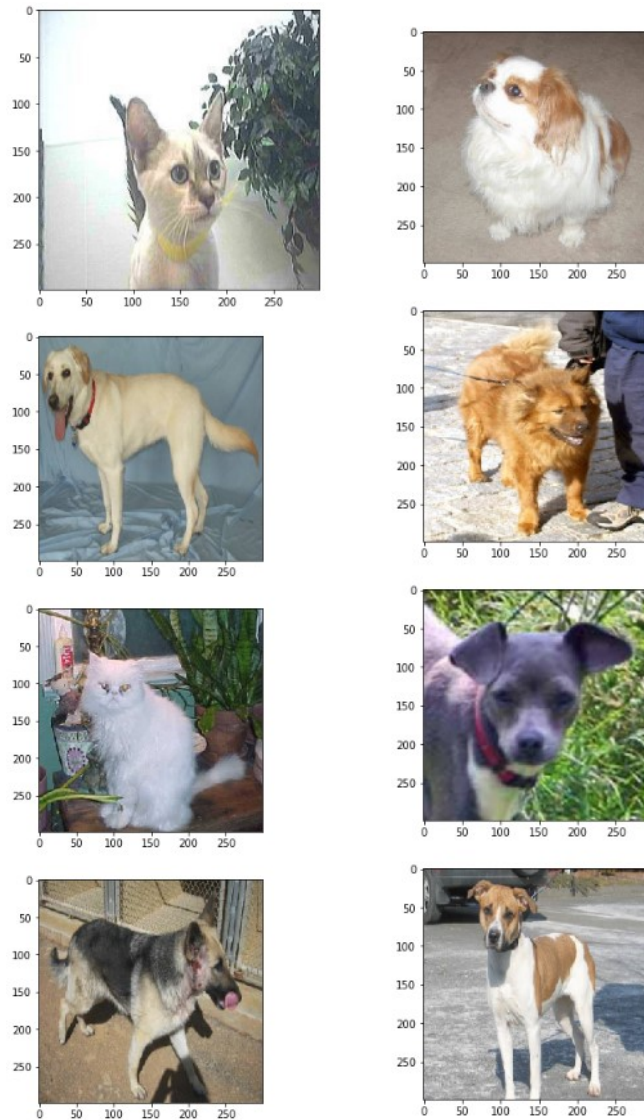


模型对测试集进行预测，并将结果上传至 Kaggle，获得得分为 0.03982。

八. 结果分析

从 Inception、ResNet50、Xception 和 Inception_Resnet_v2 训练结果上看，Inception_Resnet_v2 的对测试集的得分是最高的，甚至高于 Inception、Xception 和 Inception_Resnet_v2 融合模型的得分。为了进一步改进措施，可以使用数据增强，比如图像平移、旋转以及裁剪，或者再进行训练其他模型融合并找到更好效果的模型。

进行可视化识别出未能成功预测的图片，即查看对测试集的概率值小于 0.53 以及大于 0.47 的图片。其图片如下所示：



虽然该模型可以判断出图片是狗还是猫,但是模型并不确定,其中概率在 0.4 到 0.6 之间的测试图片共有 33 张,若可以发现不确定该 33 张图片为何无法肯定的原因,应该会让自己对神经网络模型更加熟悉。通过这个项目,我从自己安装 Ubuntu 系统,到配置深度学习环境 (CUDA 和 Cudnn),再到搭建谷歌云服务器,其过程虽痛苦,但最后实现的那一刻真心感到兴奋,非常感谢。

九、主要参考文献

- [1] Andrew Ng, Deep Learning.ai, Coursera
- [2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for image recognition, arXiv:1512.03385v1
- [3] Gao Huang, Zhuang Liu, Laurens van der Maaten. Densely Connected Convolutional Networks, arXiv:1608.06993v5
- [4] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens.

- Rethinking the Inception Architecture for Computer Vision, arXiv:1512.00567v3
- [5] Francois Chollet. Xception Deep Learning with Depthwise Separable Convolutions, arXiv:1610.02357v3
- [6] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning, arXiv:1602.07261