

Cat vs. Dog

I. 问题的定义

项目概述

猫和狗是日常生活中常见的动物，它们的照片随处可见。从照片中我们可以一眼认出照片里是猫还是狗。在 Kaggle 里就有 dogs-vs-cats 这样一个项目，提供猫狗照片，期望参与人员能够用各种方法识别照片汇总是猫还是狗。

在该项目中，我从 Kaggle 上将训练和测试数据下载下来，用训练集进行训练得到分类模型，在测试集上进行测试。

应用平台：Windows、Jupyter Notebook

问题陈述

该项目是一个有监督学习的二分类问题，因为训练集中可以知道该图片是猫还是狗，并且只有这两个类别。

以下为大概的流程：

- 1、 从 Kaggle 中下载该项目的训练数据和测试数据；
- 2、 数据预处理：
 - a) 将训练集分为训练集和验证集；
 - b) 将图片归一化；
- 3、 建立模型
 - a) 用到 VGG19、ResNet 等预训基础模型；
 - b) 用多模型融合方法进行训练调试
- 4、 训练

5、 测试

评价指标

对数损失(Log loss)亦被称为逻辑回归损失(Logistic regression loss)或者交叉熵损失(Cross-entropy loss)。

当预测准确是 $\text{cost}=0$ ，当预测错误时 cost 会很大，小的 $\log \text{loss}$ 意味着更好的分类效果：

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

其中， n 表示测试样本的数量； \hat{y}_i 表是预测概率； y_i 为样本的真实值，狗为 1，猫为 0。

II. 分析

数据的探索

数据分为训练集 (train) 和测试集 (test1)。训练集包含 12500 张猫的图片 and 12500 狗的图片，图片名字如 cat. 248. jpg 或 dog. 123. jpg 类似的格式在一个文件夹中。测试集中包含 12500 张猫狗图片，图片的命名规则如：14. jpg。



cat.6531.jpg



cat.6532.jpg



cat.6533.jpg

图 1 训练集中猫图片



图 2 训练集中狗图片

探索性可视化

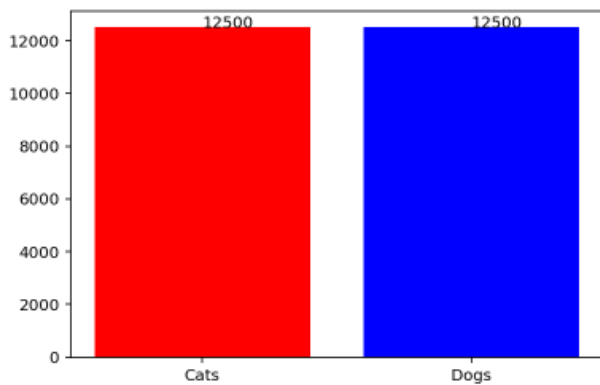


图 3 训练样本中猫和狗的数量

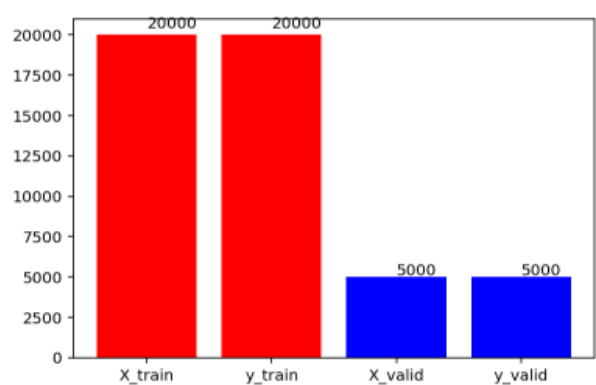


图 4 训练样本数据划分

算法和技术

该项目是属于图像分类中的二分类问题，运用神经网络提取物体的特征进行监督训练构建模型。

1、卷积神经网络介绍

卷积是对两个实变函数的一种数学运算。连续卷积函数如下：

$$s(t) = \int x(a)\omega(t-a)da$$

但在图像中数据都是离散的点，所以离散卷积函数如下：

$$s(t) = \sum_{-\infty}^{\infty} x(a)\omega(t - a)$$

在卷积神经网络中，卷积常被用来计算图像的特征，计算过程如图 5 所示，在图像中的处理如图 6 所示：

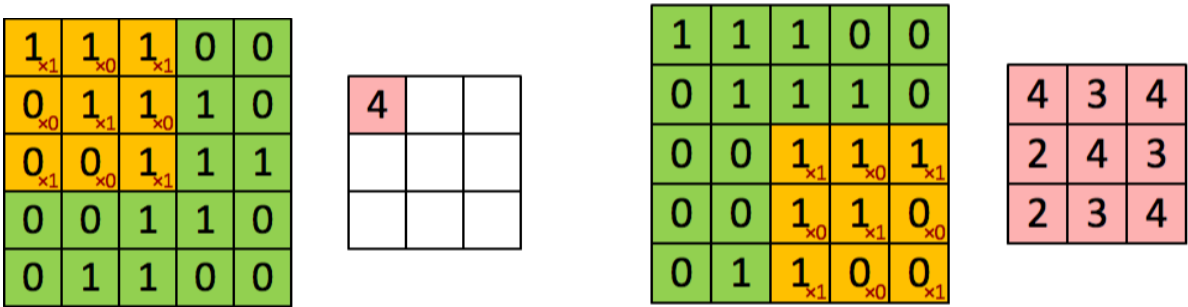


图 5 卷积计算过程



图 6 左图为卷积处理后的特征 右图为原始图片

从图 5 可以看出，卷积网络为局部连接而不是全连接，不是 flatten 之后全连接，卷积网络保留了图像数据的结构信息。卷积核的深度与输入图像一样。与全连接相比，卷积神经网络可以通过参数共享大大减少网络参数。

2、激活函数

激活函数用 Relu 函数，公式为 $h = \max(0,wx + b)$ 。Relu 函数在 $x>0$ 部分为线性，可以减少梯度消失，而 sigmoid 函数把数据压缩在 $(-1,+1)$ 范围之内造成梯度消失。

3、池化层

通常，在连续的卷积层之间会周期性地插入一个池化层。它的作用是逐渐降低数据空间尺寸，这样的话就能减少网络中参数的数量，使得计算资源耗费变少，也能有效控制过拟合^[5]。

在池化层有 max pooling 和 mean-pooling，即对邻域内特征点只求平均，max-pooling，即对邻域内特征点取最大。一般来说，mean-pooling 能减小由邻域大小受限造成的估计值方差增大产生的误差，更多的保留图像的背景信息，max-pooling 能减小由卷积层参数误差造成估计均值的偏移产生的误差，更多的保留纹理信息^[4]。

中间层中插入最大池化，对于分类问题而言，更在乎特征有没有，而不关心特征位置在哪个像素点上，因此用最大池化，把局部区域的特征分布缩减为一个数字，来表示该局部有么有特征。

而在最后一层用了 GlobalAveragePooling2D，对整个输入矩阵池化，对于分类任务而言 Global-AveragePooling 可以输出所占该特征图比例大的物体。该层是对平面计算使得利用了空间信息，对于图像在空间中变化更具鲁棒性，同时对整个网络从结构上做正则化防止过拟合^[6]。

4、损失函数

因为评价指标里我们用到了准确率和对数损失函数，所以在用 Keras 编译的时候我们用到了 binary_crossentropy 和 accuracy 这两个参数，优化器选择 adadelta 优化器^[2]：

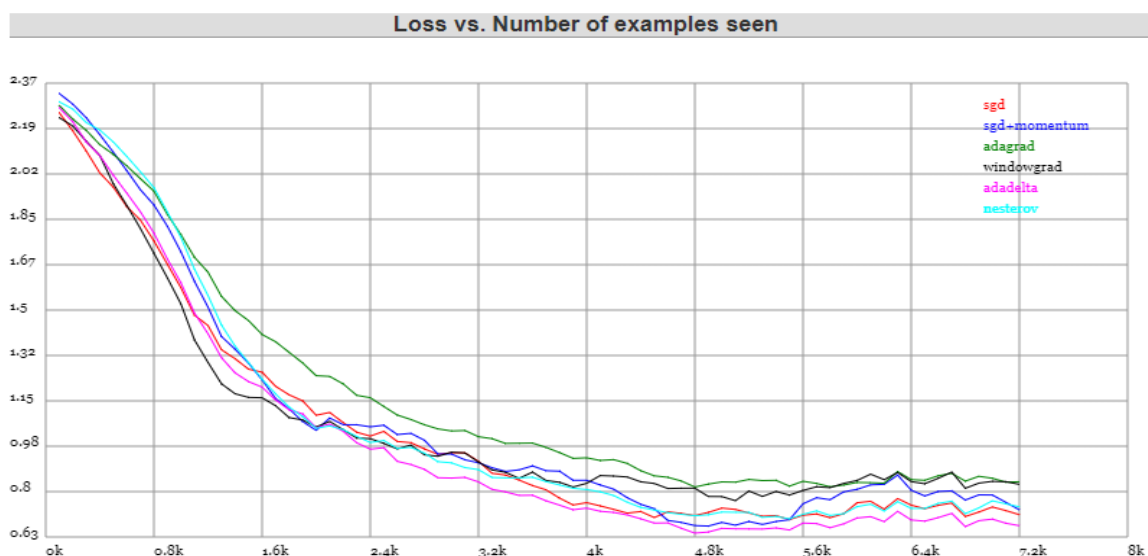


图 7 各种优化器下的损失

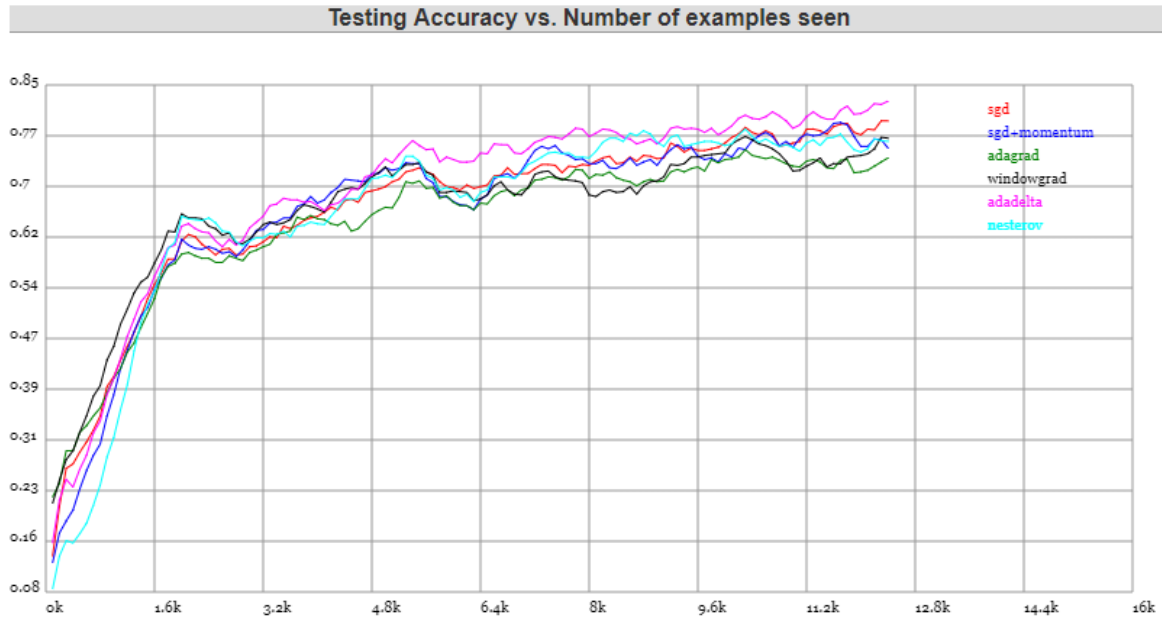


图 8 各种优化器下测试集上准确率

5、迁移学习

该项目用到了迁移学习。转移学习是机器学习中的一个研究问题，其重点在于存储在解决一个问题时所获得的知识，并将其应用于不同但相关的问题^[7]。

在实践中，很少有人从头开始训练整个卷积网络，因为拥有足够大小的数据集相对来说比较少见。预训练模型是已经训练好且已经收敛的模型能够加快训练速度且有比较好的效果。

基准模型

Kaggle 中该项目第 100 名 Loss 值：0.05629。

III. 方法

数据处理

训练数据需要划分为训练集、验证集：

在训练和测试数据集中图像的大小不一，如图 1 和图 2 所示。在 VGG、ResNet 模型中，图片输入尺寸为 224×224 进行训练的（在 InceptionResNetV2、InceptionV3、Xception 模型中图像的输入为 299×299 ），所以在处理训练数据时需要将输入图像进行尺寸变换为 224×224 （或 299×299 ）大小载入矩阵中。由于在训练的 25000 张图像然中，前 12500 为猫，后 12500 为狗的图像。所以在标签向量中，将前 12500 幅猫的图像标记为 0，后 12500 幅狗的图像标记为 1。下图为尺寸变换后的图片：



图 9 归一化后的图片

执行过程

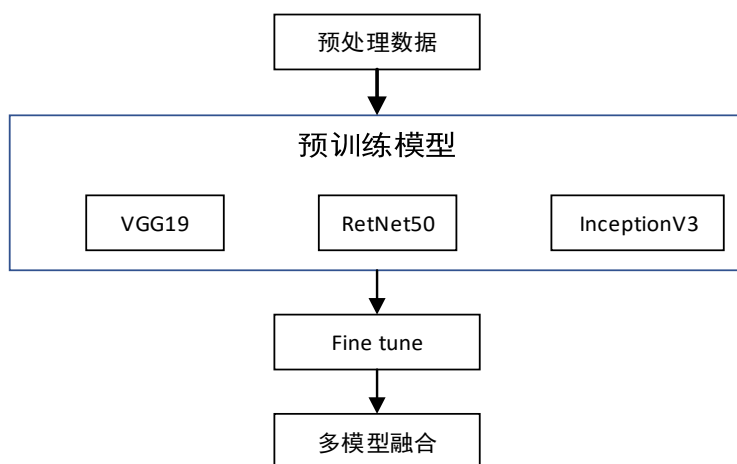


图 10 执行过程流程图

1、预训练模型

(1) VGG19

如下图所示，VGG19 由 19 层组成^[8]，最后三层为全连接层。最后一个全连接层相当于分类器，图中为 1000 个分类，但在该项目中只有猫和狗两类，所以最后一层需要改为只有两个结果的全连接层。

VGG19					
序号	层结构				
1	conv1-1	1	24	conv4-3	11
2	relu1-1		25	relu4-3	
3	conv1-2	2	26	conv4-4	12
4	relu1-2		27	relu4-4	
5	pool1		28	pool4	
6	conv2-1	3	29	conv5-1	13
7	relu2-1		30	relu5-1	
8	conv2-2	4	31	conv5-2	14
9	relu2-2		32	relu5-2	
10	pool2		33	conv5-3	15
11	conv3-1	5	34	relu5-3	
12	relu3-1		35	conv5-4	16
13	conv3-2	6	36	relu5-4	
14	relu3-2		37	pool5	
15	conv3-3	7	38	fc6(4096)	17
16	relu3-3		39	relu6	
17	conv3-4	8	40	fc7(4096)	18
18	relu3-4		41	relu7	
19	pool3		42	fc8(1000)	19
20	conv4-1	9	43	prob(softmax)	
21	relu4-1				
22	conv4-2	10			
23	relu4-2				

图 11 VGG19 结构图

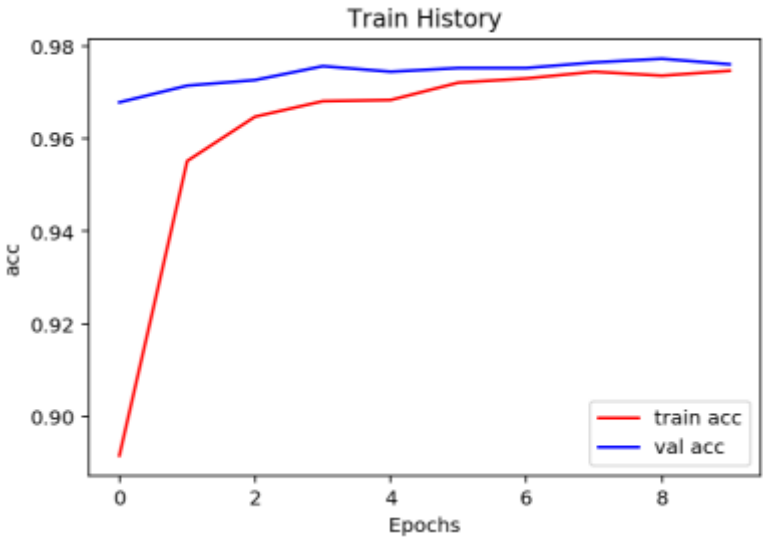


图 12 VGG19 模型

Name	Submitted	Wait time	Execution time	Score
y_pred_VGG19.csv	just now	0 seconds	0 seconds	0.07314
Complete				

图 13 VGG19 训练后 Kaggle 评分

VGG 网络简单，并且能够取得较好的效果。

(2) ResNet

ResNet 引入了残差网络结构（residual network），通过残差网络，可以把网络层弄的很深，据说现在达到了 1000 多层，最终的网络分类的效果也是非常好，残差网络的基本结构如下图所示：

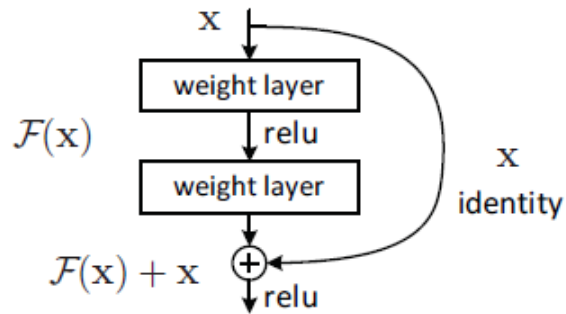


图 14 残差网络

通过在输出个输入之间引入一个 shortcut connection,而不是简单的堆叠网络，这样可以解决网络由于很深出现梯度消失的问题，从而可可以把网络做的很深，ResNet 其中一个网络结构如下图所示^[9]：

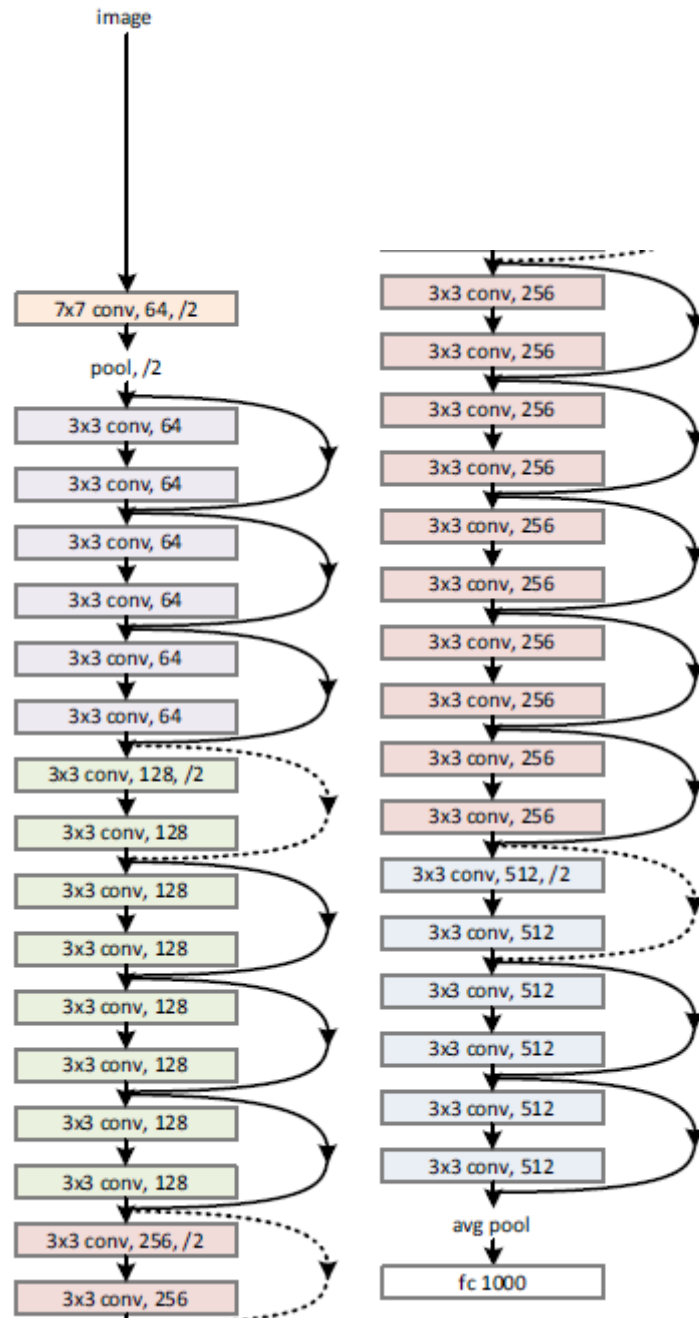


图 15 ResNet 网络结构

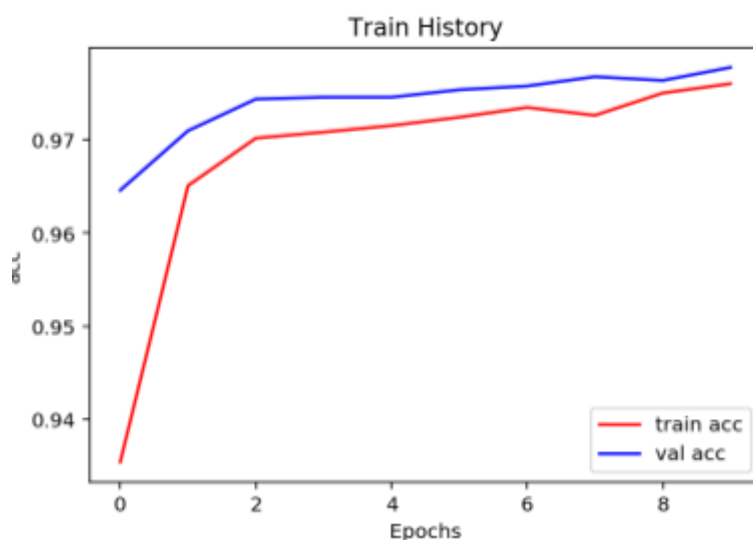


图 16 ResNet50 模型

Name	Submitted	Wait time	Execution time	Score
y_pred_ResNet50.csv	just now	0 seconds	0 seconds	0.07286
Complete				

图 17 ResNet50 训练后 Kaggle 评分

ResNet 网络中加入残差可以解决退化问题。

(3) InceptionV3

type	patch size/stride or remarks	input size
conv	$3 \times 3 / 2$	$299 \times 299 \times 3$
conv	$3 \times 3 / 1$	$149 \times 149 \times 32$
conv padded	$3 \times 3 / 1$	$147 \times 147 \times 32$
pool	$3 \times 3 / 2$	$147 \times 147 \times 64$
conv	$3 \times 3 / 1$	$73 \times 73 \times 64$
conv	$3 \times 3 / 2$	$71 \times 71 \times 80$
conv	$3 \times 3 / 1$	$35 \times 35 \times 192$
3×Inception	As in figure 5	$35 \times 35 \times 288$
5×Inception	As in figure 6	$17 \times 17 \times 768$
2×Inception	As in figure 7	$8 \times 8 \times 1280$
pool	8×8	$8 \times 8 \times 2048$
linear	logits	$1 \times 1 \times 2048$
softmax	classifier	$1 \times 1 \times 1000$

图 18 Inception 结构图^[10]

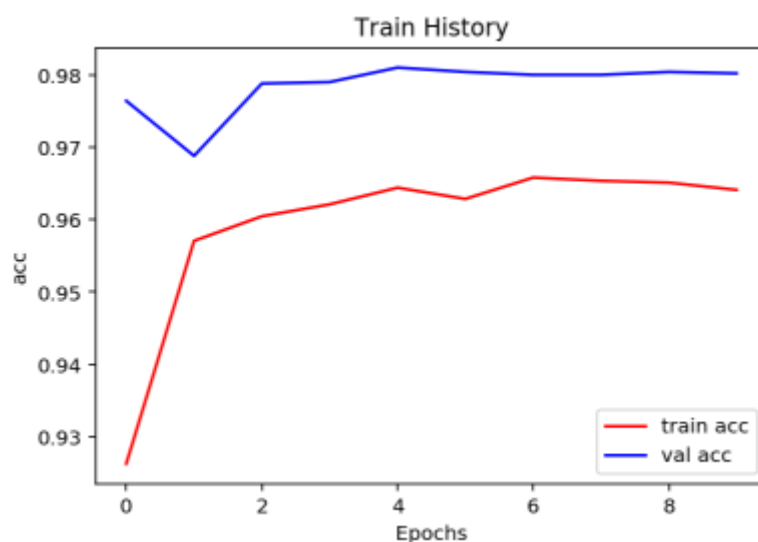


图 19 InceptionV3 模型

Name	Submitted	Wait time	Execution time	Score
y_pred_InceptionV3.csv	just now	0 seconds	0 seconds	0.07740
Complete				

图 20 InceptionV3 训练后 Kaggle 评分

Inception 网络加了网络的宽度，另一方面增加了网络对尺度的适应性。

完善

Fine Tune

1、Dropout

使用 Dropout 防过拟合^[12]：dropout 掉不同的隐藏神经元就类似在训练不同的网络，整个 dropout 过程就相当于 对很多个不同的神经网络取平均。而不同的网络产生不同的过拟合，一些互为“反向”的拟合相互抵消就可以达到整体上减少过拟合。减少神经元之间复杂的共适应关系，因为 dropout 程序导致两个神经元不一定每次都在一个 dropout 网络中出现，迫使网络去学习更加鲁棒的特征^[11]。

2、使用 clip 将预测结果限制在[0.005, 0.995]之间，由于 Kaggle 的评估标准是 LogLoss，对于预测正确的样本， $\ln 0.995$ 和 $\ln 1$ 相差无几；但是对于分类错误的样本，0 和 0.005 差别还是相当大的，改为 0.005 是一个不错的经验超参数。

单模型参数微调之后并没有达到理想的情况。

Fine Tune 的参数具体如下：

- 1) 将 ResNet50 的 100 层之后的 trainable 设置为 True，使得变量可以被训练；
- 2) 将 Dropout 参数设置为 0.2，即保留 0.8。

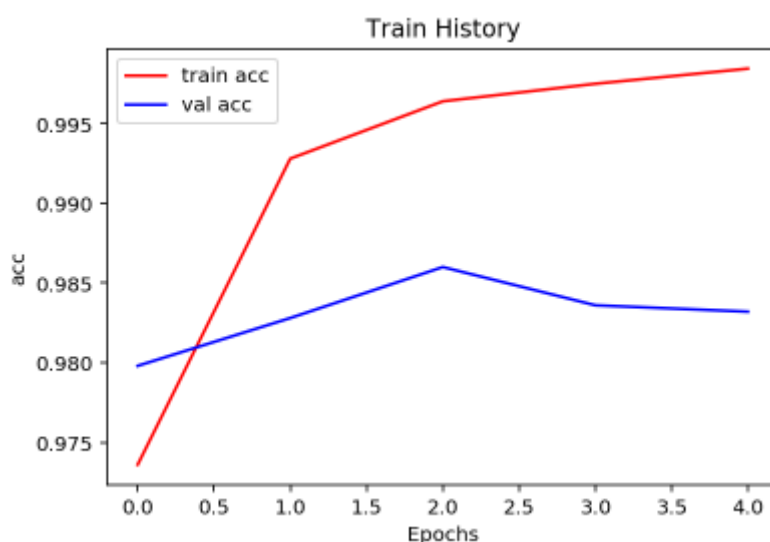


图 21 调整后的模型

Name	Submitted	Wait time	Execution time	Score
y_pred_finetune_ResNet50.csv	just now	3 seconds	0 seconds	0.05706
Complete				

图 22 参数微调 Kaggle 评分

2、多模型融合

选择 InceptionResNetV2、InceptionV3、Xception 这三个模型融合是因为生成的模型小巧且准确率高^[3]，增加特征数量。

首先用分别用 InceptionResNetV2、InceptionV3、Xception 直接 predict 得到预测结果作为特征，将其拼接在一起送入模型中进行训练。后续步骤跟单一模型类似。结果如下图所示：

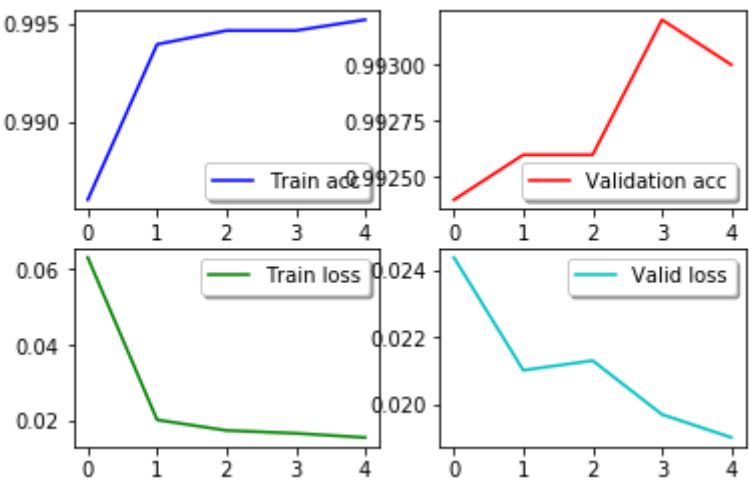


图 23 多模型试验结果



Name	Submitted	Wait time	Execution time	Score	
fetch_extract_feature_submission.csv	an hour ago	0 seconds	0 seconds	0.04372	
Complete					
Jump to your position on the leaderboard ▾					
26	—	lystdo	 0.04357	19	1y
27	—	Kraus Brothers	 0.04395	29	1y

图 24 多模型融合后 Kaggle 评分

IV. 结果

模型的评价与验证



Name	Submitted	Wait time	Execution time	Score
fetch_extract_feature_submission.csv	an hour ago	0 seconds	0 seconds	0.04372
Complete				
Jump to your position on the leaderboard ▼				
26	—	lystdo	 0.04357	19 1y
27	—	Kraus Brothers	 0.04395	29 1y

图 25 多模型融合后 Kaggle 评分

Kaggle 得分为 0.04372, 已经超过 Kaggle 第 100 名的阈值。

V. 项目结论

多融合模型比单一模型效果更好。原因如下^[12]:

- 1) 从统计的方面来看, 由于学习任务的假设空间往往很大, 可能有多个假设在训练集上达到同等性能, 此时若使用单一模型可能因误选而导致泛化性能不佳, 多模型则会减少这一风险;
- 2) 从计算的方面来看, 学习算法往往会陷入局部极小, 有的局部极小点所对应的泛化性能可能很糟糕, 而通过多次运算之后进行结合, 可降低陷入糟糕局部极小点的风险;
- 3) 从表示的方面来看, 某些学习任务的真实假设可能不在当前学习算法所考虑的假设空间中, 此时若使用单一模型则肯定无效, 而通过多模型融合, 由于相应的假设空间有所扩大, 有可能学的更好的近似。

改进

将分类错误的显示出来如图 26 所示, 像第一行前两张图像那样类似的模糊图像, 可以在图像数据中加入图像模糊、图像旋转、图像剪切等处理以强化数据提高泛化能力。

而想第二行中后两幅图像中猫的图像太小, 在用 AveragePooling 对特征图进行操作时提取不到猫的特征, 导致分类错误。



图 26 无法分类的图像

参考文献

- [1] 俞祝良 人工智能技术发展概述[J] 2017 南京信息工程大学学报(自然科学版)
- [2] <http://cs.stanford.edu/people/karpathy/convnetjs/demo/trainers.html>
- [3] Alfredo Canziani, Adam Paszke, Eugenio Culurciello arXiv:1605.07678 [J] An Analysis of Deep Neural Network Models for Practical Applications
- [4] Y-Lan Boureau, Francis Bach, Yann LeCun, Jean Ponce. Learning Mid-Level Features For Recognition
- [5] <http://cs231n.github.io/convolutional-networks/>
- [6] Min Lin¹, Qiang Chen, Shuicheng Yan. Network In Network
- [7] West, Jeremy; Ventura, Dan; Warnick, Sean (2007). "Spring Research Presentation: A Theoretical Foundation for Inductive Transfer". Brigham Young University, College of Physical and Mathematical Sciences. Archived from the original on 2007-08-01. Retrieved 2007-08-05
- [8] K. Simonyan, A. Zisserman Very Deep Convolutional Networks for Large-Scale Image Recognition arXiv technical report, 2014

- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition
- [10] Christian Szegedy, Vincent Vanhoucke, **Sergey Ioffe, Jonathon Shlens**. **Rethinking the Inception Architecture for Computer Vision**
- [11] <https://zhuanlan.zhihu.com/p/23178423>
- [12] Ian Goodfellow, Yoshua Bengio, Aaron Courville. 深度学习[M]. 人民邮电出版社, 2017. 159-165
- [13] 周志华. 机器学习[M]. 清华大学出版社, 2016. 181-181