猫狗大战

机器学习工程师 纳米学位 毕业项目

陈睿嘉

2018 年 3 月 21 日

目录

1. 定义.....1
1.1 项目概述.....1
1.2 问题陈述.....1
1.3 评价指标.....2
2. 分析.....2
2.1 数据描述.....2
2.2 算法和技术.....4
2.2.1 深度学习.....4
2.2.2 卷积神经网络.....5
2.2.3 解决方案.....6
2.3 基准指标.....7
3. 方法.....9
3.1 模型选择.....9
3.2 数据清洗.....9
3.3 单模型迁移学习.....11
3.3.1 模型搭建.....11
3.3.1 实验结果.....12
3.4 单模型 Fine-tune.....13
3.4.1 模型搭建.....13
3.4.2 实验结果.....14
3.5 多模型融合.....15
3.5.1 模型搭建.....15
3.5.2 实验结果.....16
3.6 改进.....16
4. 结论.....17
参考文献.....18

1. 定义

1.1 项目概述

猫狗大战源自 Kaggle 于 2013 年举办的一个娱乐竞赛项目，提供了 25000 张猫狗训练数据以及 12500 张测试数据，数据主要为真实场景采集的猫狗照片。

项目要求分辨给定测试图片是猫还是狗，属于计算机视觉领域图像分类问题，适合使用以卷积神经网络（Convolutional Neural Network，CNN）构建模型、最小化对数损失函数作为策略、梯度下降与反向传播作为算法的深度学习方法。

1.2 问题陈述

猫狗大战本质上属于监督学习，假设样本数据服从伯努利分布，学习一个条件概率分布模型 $p(y|x)$，输入图片像素值向量 x，输出样本 x 表示狗的概率 $\hat{y} = p(y = dog|x)$，那么样本表示猫的概率就为 $1-\hat{y}$。

然而，图像数据的特征不易提取，甚至不易理解，人类本身也无法准确描述分辨猫狗的核心特征。但若使用整个像素向量作为特征向量，以像素向量的高维度性，使用传统机器学习方法将带来维度灾难（Curse of dimensionality），需求像素维度指数级别的均匀样本，以及指数增长的模型规模与计算量。

针对维度灾难，深度神经网络（DNN, Deep Neural Network）的层次结构可以将模型容量（Capacity）增长为样本维度的指数级别，而没有增加太多的模型参数，并且，高维度的图像数据向量并不是每种组合都有意义，模型需要学习的模式通常为数据向量的某种低维流形表示，随机像素组合大都没有意义，能表示猫狗甚至有意义的图像的像素向量只占极少数，并不需要样本维度指数级别的训练样本，使用深度网络模型满足解决此问题的理论基础。然而，对于图像数据，DNN 模型引入的参数太多，随着网络规模增加，几乎无法训练，由此，根据像素值往往只与附近区域的像素高度相关这一先验特征，CNN 模型成为了当前最适合进行图像分类的先验模型。

图像分类问题的代表性数据集为 ImageNet，需要从超过 100,0000 张图片的训练数据中学习识别 1000 种图像分类（包括猫狗等动物）。通常在 ImageNet

问题上表现优秀的模型，对其他图像分类问题甚至不限于分类问题具有良好的泛化性能。由于猫狗分类属于 ImageNet 的子问题，适合使用迁移学习（Transfer Learning）方法，使用预训练模型在 ImageNet 中学习到的经验参数表示猫狗图片的低维特征向量，并学习对这些特征向量进行分类。

1.3 评价指标

二分类模型输出单元通常使用 Sigmoid 激活函数，损失函数采用对数损失函数，即伯努利分布下的交叉熵：

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

$$Cost = \frac{1}{m}\sum_{i=1}^{m} L(y_i,\hat{y}_i)$$

损失函数的形式可以通过对模型输出 $\hat{y} = p(y = dog \mid x)$ 取极大似然估计导出，并且其对数形式可以解决 Sigmoid 输出单元两端饱和导致梯度消失的缺陷。

模型在测试数据的分类结果可以上传到 Kaggle 进行评价、排名。Kaggle 对于测试结果的评价指标也使用对数损失函数，表示不仅要求分类准确率尽量高，还要求正确分类置信度尽量高。

2. 分析

2.1 数据描述

项目使用了 Kaggle 竞赛提供的数据集，其训练数据集包括 25000 张图片，并在文件名中标注了图片为猫还是狗，猫狗数量各占一半，测试数据集包括 12500 张图片，没有标注类别，文件以数值 ID 命名。

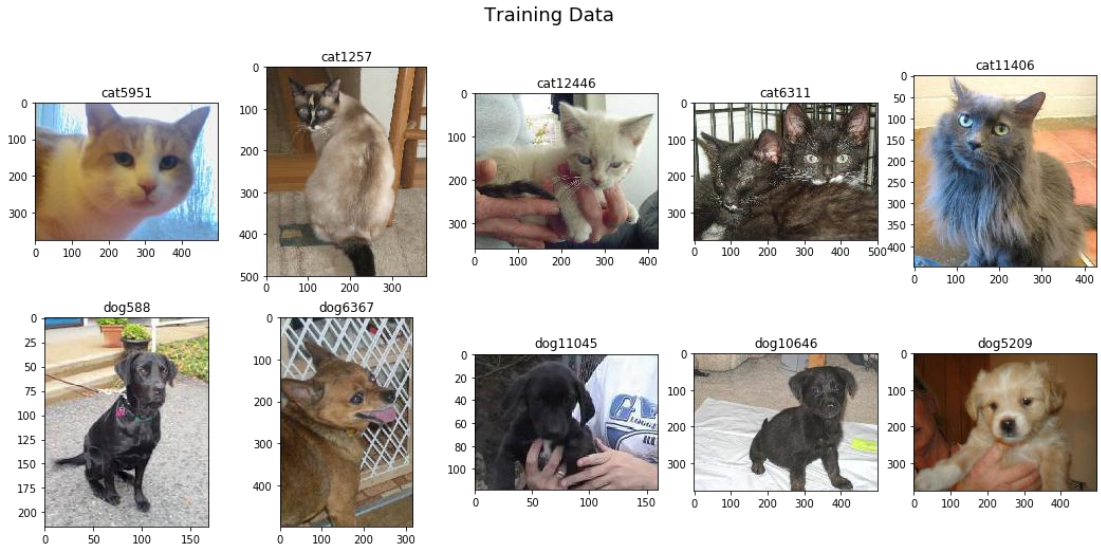

图 1. 训练数据按类别随机抽样

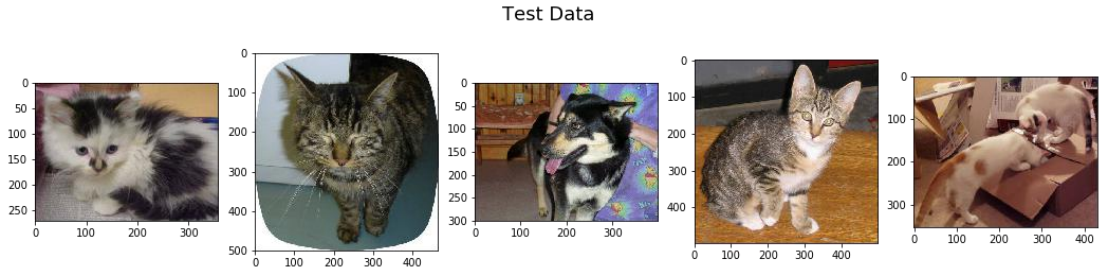

图 2. 训练数据随机抽样

如上图所示，图 1 为从训练数据集中随机抽样猫狗各 5 个样本，图 2 为从测试数据集中随机抽样 5 个样本，可见数据内容具备多样性与复杂性。如：图片尺寸参差不齐，不同图片包括了猫/狗身体不同角度，不同的猫/狗数量，有的图片没有包含全身，甚至没有正面脸部，图片背景元素也具备充分的干扰性。

需要注意，由于猫狗数据样本比例相当，在进行数据抽样、分割等操作时，必须保证不同子数据集猫狗样本也各占一半。

图片样本特征十分抽象，很难使用数据统计方法剔除异常值，如下对比较直观的特征包括图像宽、高、宽高比进行了箱型图统计：

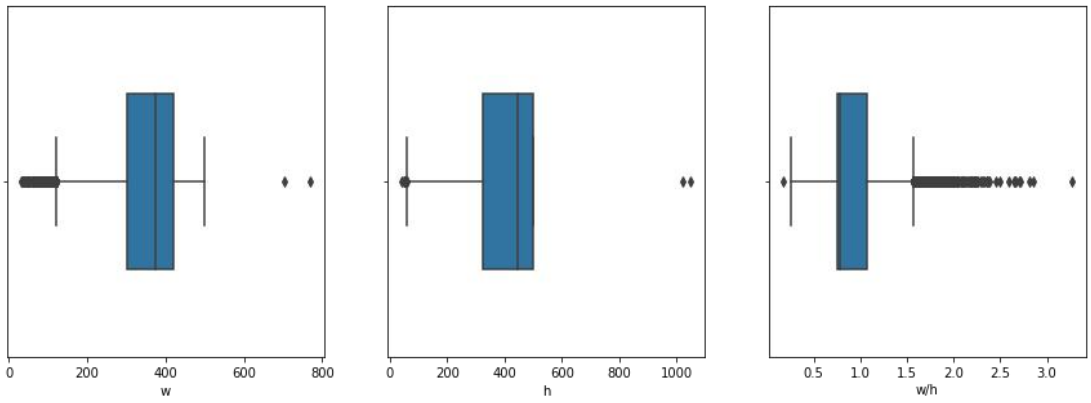

图 3. 图像尺寸数据箱型图

如图 3 所示，虽然大部分数据分布在正常尺寸，但仍然存在大量异常样本，由于数量太大，逐个进行裁剪、填充需要大量时间，并且深度学习对于此类异常样本具有较强鲁棒性。所以，需要使用预训练的分类模型对数据集进行初步分类，识别非猫狗异常样本，然后再对数量较少的异常样本集合进行人为判断，对内容正确但尺寸异常的数据进行裁剪、填充。

2.2 算法和技术

2.2.1 深度学习

针对传统机器学习方法对于高维度相关性数据（如图片、语音、文字等）上表示能力的不足，深度学习着重解决了由此带来的维度灾难以及计算复杂度问题。

深度神经网络结构提出隐藏层的概念，并且由万能近似定理（Universal Approximation Theorem）证明，若不考虑训练难度，只要给予足够数量的隐藏单元，模型可以以任意精度近似一个从有限维度映射到另一个有限维度的可测函数，并且，模型的表示能力是网络深度的指数级函数[1]。

对于高维度相关性数据，以图片为例，若对每个维度根据均匀分布随机选择像素值，最终有非常大的概率得到噪声图像，说明数据分布通常集中在某个坐标区域，可以使用一个低维流形坐标系表示，深度学习模型可以很好的近似这一坐标映射函数。

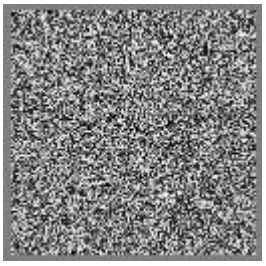

图 4. 随机生成的噪声图像

传统机器学习方法如支持向量机（SVM, Support Vector Machine），其核函数也可以将数据隐式映射到合适的维度，但很难针对不同需求找到合适的核函数，深度学习可以通过 DNN 结构近似表达此种显式映射函数，以解决传统方法的不足。

2.2.2 卷积神经网络

卷积神经网络根据生物视觉神经中感受野（Receptive Field）概念的启发，使用卷积核与上层的稀疏连接取代普通 DNN 层与层之间的全连接权重，极大降低了网络参数数量，而理论上相比全连接网络表示能力只是略微降低，并且更容易训练。由于计算机视觉相关问题数据结构的特殊性（图像、视频等），往往每个像素只与附近区域的像素高度相关，非常适合使用 CNN 模型。

卷积是一种数学运算，其标准定义为：

$$x(t) * w(t) = \int_{-\infty}^{\infty} x(a) \cdot w(t-a) da$$

卷积运算的实际意义需要应用到不同场景，若将 t 理解为时间，令 a<0 时 w=0，可以理解为从过去到时刻 t，x 以 w 为权重的累加和。扩展到离散场景时，可定义为：

$$x(t) * w(t) = \sum_{a=-\infty}^{\infty} x(a) \cdot w(t-a)$$

在 CNN 图像应用场景中，卷积可以扩展为更为直观的二维形式：

$$I(i,j) * K(i,j) = \sum_{m=1}^{W} \sum_{n=1}^{H} I(m,n) \cdot K(i-m, j-n)$$

其中 I 表示输入图像，i、j 为图像像素坐标，K 表示卷积核，W、H 为卷积核尺寸，即对输入图像中每个像素值（暂不考虑步长）及其附近 W、H 区域的像素值进行加权累加和，以得到一副新的“图像”。

通过卷积计算得到的新“图像”，可以表示出原图像的某些特征，比如传统的边缘检测算子就是一种人为设计权重的卷积核，将如 3×3 大小的卷积核分别对原图像每个像素进行卷积运算，从而得到如图 5 所示的边缘检测效果，而 CNN 模型则是根据代价函数和梯度下降让卷积核自己学习需要的权重，让模型自己学习需要提取的图像特征。并且，随着模型深度的增加，卷积核的视野逐渐增大，从而可以提取更为高级、抽象的特征，如从边缘到毛发，最终到身体轮廓。

图 5. 边缘检测算子

相对于 DNN 结构，CNN 最大的优势就是通过稀疏交互和参数共享极大降低了网络参数数量，并且保留了特征的平移不变性。对于图像数据的维度规模，大型 DNN 结构带来的参数量与计算量使模型几乎无法训练，并且将会严重过拟合，相对的，CNN 的缺点是增加了额外的需要人为调整的超参数，如卷积核大小、步长等。

2.2.3 解决方案

项目使用 Python3.6 开发，通过对 Tensorflow 进一步封装的 Keras API 进行模型实现与训练。Tensorflow 是 Google 提供的深度学习开源框架，其主要功能是自动完成了计算图的梯度计算与反向传播过程，使开发者可以专注于模型搭建与训练，并且将前馈计算、反向传播等密集浮点运算交由效率更高的语言完成，

支持 GPU 加速计算。由于 GPU 的构架特点，其浮点运算性能远超 CPU，使用 NVIDIA GPU 加速计算，还需要安装 CUDA 以及 cuDNN。Keras 对 Tensorflow 进行了进一步封装，实现了常用的层次结构，使调用接口更加人性化，并且提供了一些搭建好的预训练模型，以提高开发效率，避免重复造轮子。

项目主要使用 Keras 提供的 ImageNet 数据集预训练模型进行迁移学习，通过 fine-tune 卷积层、多模型拼接等方法提高模型性能，以及通过 Dropout、图像增强等方法解决过拟合。

需要注意，由于 Kaggle 评估指标为损失函数，进行多模型拼接时，不宜使用投票表决法、均值法等提高准确度指标的方法；同时，选择解决过拟合方法时，也不宜使用 L1/L2 Regularization，前者会导致权重稀疏，后者会导致权重衰减，两者都会使预测时输出层 Sigmoid 函数接收的激活绝对值变小，从而输出概率不够自信。Dropout 方法在每次训练时随机丢掉一部分特征，使当前某些强特征随时会被在下次训练时丢弃掉，从而让模型不会过分依赖某几个强特征，虽然也会降低训练时的输出层激活绝对值，但预测过程中将不会对特征进行丢弃，非常适用于此项目；图像增强方法是通过让模型见到更多人造数据以提高泛化性能，但会相应增加每代训练时间，需要根据实验结果进行取舍。

又由于 loss 对数函数的性质，当样本预测分类错误时，若模型输出过于自信，使对数函数输入接近 0，将导致对数函数值接近负无穷，而样本分类正确时取值于对数函数饱和端，略微降低自信度影响不大，所以在提交 Kaggle 评估得分时，应当对输出概率进行截断。令预测错误率为 a，输出截断范围[x, 1-x]，该优化问题可近似为：

$$\arg\max_{x}[a\log x+(1-a)\log(1-x)]$$

求导得到 x=a，由于模型在验证集正确率约为 0.996，这里采取对输出截断至范围[0.005, 0.995]。

2.3 基准指标

ImageNet 的分类评价指标一般为 top-5 test error，即根据概率从大到小排名前 5 的分类结果中没有包括正确分类的百分比错误率。

VggNet 使用最深为 19 层的传统 CNN 结构，在 ImageNet 中达到单模型 7.0%，组合模型 6.8%的 top-5 error[2]。

GoogLeNet 即第一版 Inception 模型，增加网络宽度，减少了选择卷积核尺寸等参数的人为因素，直接将不同尺寸卷积核以及池化层放在同一层作为一个 block，并拼接为一个输出层，然后为了降低计算参数，使用 1×1 卷积核进行降维，达到单模型 7.9%，组合模型 6.7%的 top-5 error[3]。

ResNet 则着重解决网络深度增加难以训练的问题。以解决输入到输出的直接映射为核心思想，将输出函数改为源目标函数 H(x)与输入的残差（Residual）F(x)，再将输入层无参数连接（或维度映射）到输出层，使最终输出保持为 H(x)=F(x)+x，使网络能够达到极高的深度而几乎不增加额外参数。ResNet50 与 ResNet152 分别达到 6.7%与 5.7%，组合模型甚至达到 3.7%的 top-5 error[4]。

Inception v3 在 GoogLeNet 的基础上，将大型卷积核分解为多个小型甚至一维卷积核，并加上非线性激活单元，在降低参数数量的基础上提高了模型表现，组合模型最好表现达到 3.58%的 top-5 error[5]。

Inception v4 将 block 结构设计的更加复杂，使单模型 top-5 error 达到 3.8%，同时提出结合 ResNet 的 Inception-ResNet v2，达到 3.7%的 top-5 error 并且训练更快[6]。

Xception 将图像空间信息与 channel 信息分开处理，使用 depthwise separable convolution 方法对每个 channel 分开卷积，然后使用 1×1 卷积核进行合并，进一步降低参数数量，提高模型表现能力，在 ImageNet 上单模型表现接近 Inception v4，作者将此归为 Inception 的复杂设计对 ImageNet 存在过拟合[7]。

ResNeXt 针对 Inception 的 block 内部结构太过复杂而引入过多先验设计，设计一种子结构相同的 block，在复杂度降低一半的情况下达到比 Res200 更好的表现[8]。

使用 ResNet50、Inception v3、Xception 组合模型进行迁移学习，对输出的特征向量进行逻辑回归（Logistic Regression），可以在验证集上达到 99.6%的正确率，在测试集的损失函数值达到 0.04141，提交 Kaggle 排名为 20/1314，要求项目结果不低于此标准。

3. 方法

3.1 模型选择

在 Keras 目前提供的所有预训练模型中，InceptionResNetV2、Xception、InceptionV3 模型在 ImageNet 验证集表现位列前三，所以选用此三个模型分别进行实验。

在将图片数据输入模型之前，需要通过重采样或插值方法将尺寸映射到模型需要的尺寸，所选择的三个模型输入尺寸都为 299×299，并需要将像素值从[0, 255]的取值范围映射到[-1, 1]。

3.2 数据清洗

由于 ImageNet 分类中已包含 118 种狗类别和 7 种猫类别，使用预训练模型分别对各 12500 个猫狗训练样本进行预测，并输出前 n 个预测结果分类，若猫样本的 n 个预测结果中没有 7 种猫类别，或狗样本的 n 个预测结果中没有 118 种狗类别，即判定该样本为异常样本，n 值的大小经过反复实验设定为猫样本 60，狗样本 20。最终使用 InceptionResNetV2、Xception、InceptionV3 三个模型分别预测后，对三种结果取并集得到共 116 个异常样本。

图 6. 部分异常样本

可以发现，其中大部分图片如“dog.1773.jpg”、“cat.10712.jpg”判定为异常样本毫无疑问，必须剔除，而如“cat.12424.jpg”、“cat.252.jpg”属于分辨率过低，也可以剔除，又如“dog.7805.jpg”此种类型，背景干扰太大。为了进一步筛选异常样本，在将此 116 个异常样本剔除后，本次项目从中进一步人为判断筛选出 11 个轻度异常样本，并将其裁剪、填充至模型需要的尺寸，项目选用的三个模型输入尺寸都为 299×299，如图 6 中“cat.6655.jpg”，将其填充为图 7 所示：

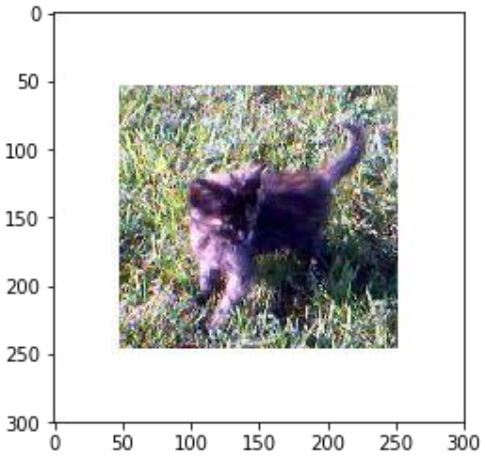

图 7. 裁剪填充结果

清洗后数据剩余 24895，将处理后的数据按猫狗等比例分割，其中 80%作为训练集，共 19916，20%作为验证集，共 4979。

3.3 单模型迁移学习

3.3.1 模型搭建

卷积特征随着卷积层深度的增加由具体到抽象，而类似的图片数据通常具有可共用的特征，如边缘都是由横线、竖线、斜线的组合表示。迁移学习利用预训练模型在大规模数据集中学习到的卷积核权重，可以直接提取出类似数据集中具有较强泛化性能的高级抽象特征，从而节省大量训练时间。通常方法是保留预训练模型的深层卷积结构，去掉后几层池化、输出层，然后针对新需求重新构建输出结构。

由于所选的预训练模型结构已经十分复杂，使更简单的猫狗数据通过这些特征表示后几乎线性可分，选择对模型输出特征图进行全局池化（GlobalAveragePooling2D）后直接使用逻辑回归。因为只有最后的 Dense 层参数可训练，可以先使用模型对所有清洗后的训练数据、测试数据进行预测，输出全局池化后的特征向量作为数据载体，仅使用 Dropout 层和 Dense 层进行逻辑回归，极大简化模型，节省训练时间，其缺陷是无法使用图像增强。模型分为提取特征向量、逻辑回归两部分：

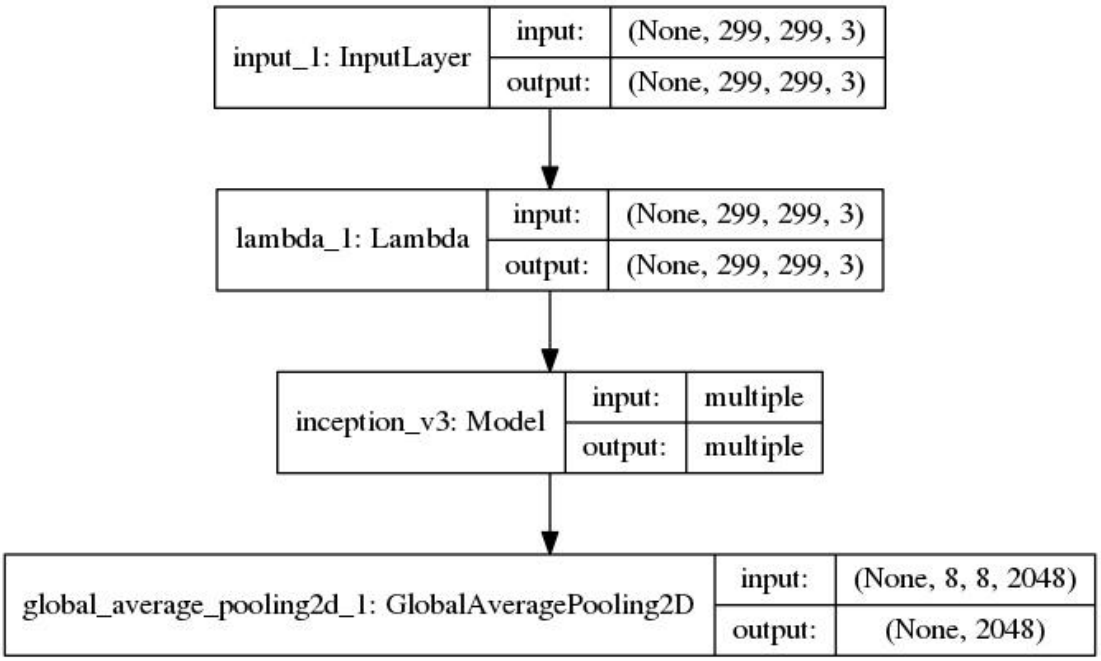

图 8. InceptionV3 特征提取

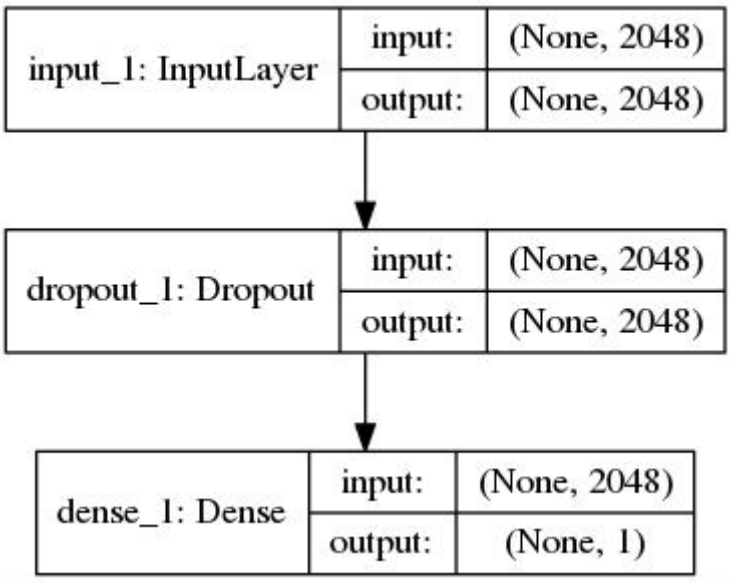

图 9. 特征向量逻辑回归模型

3.3.1 实验结果

训练使用 128 个样本作为 batch size，Adam 作为优化方法，学习率1^{-3}，以及使用衰减率1^{-6}的 Keras 默认衰减函数：

$$lr = \frac{lr}{1 + decay\_rate * batch\_num}$$

将 Early Stop 设置为验证集 loss 连续 5 代没有降低即停止，保留 loss 最低的模型，学习曲线如下：

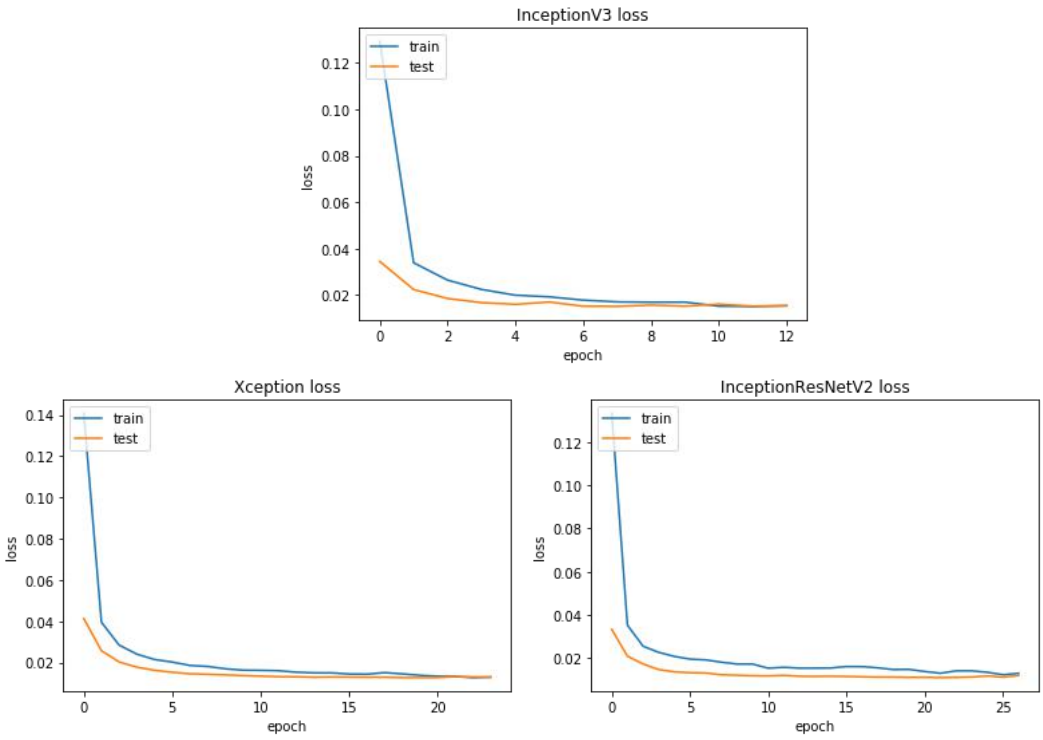

图 10. 学习曲线

将预测结果提交 Kaggle 之后，InceptionResNetV2 表现最佳，最好能得到 0.03929 的 loss，InceptionV3 则最好得到 0.04121 的 loss，Xception 模型虽然在验证集表现强于 InceptionV3，但在测试集只得到了最好 0.04188 的 loss，说明 Xception 对验证集的过拟合更严重。

3.4 单模型 Fine-tune

3.4.1 模型搭建

Xception 模型使用最少的参数数量达到了差不多的性能，并且也引入了残差结构，理论上需要的训练时间最短，项目选择 Xception 模型进行 fine-tune 实验，对 block14 的卷积层进行微调，包括输出层 2049 个参数，一共 4750849 个可训练参数。由于微调卷积层时必然会趋向于过拟合，模型搭建也采用直接 Dropout 连接逻辑回归输出层。

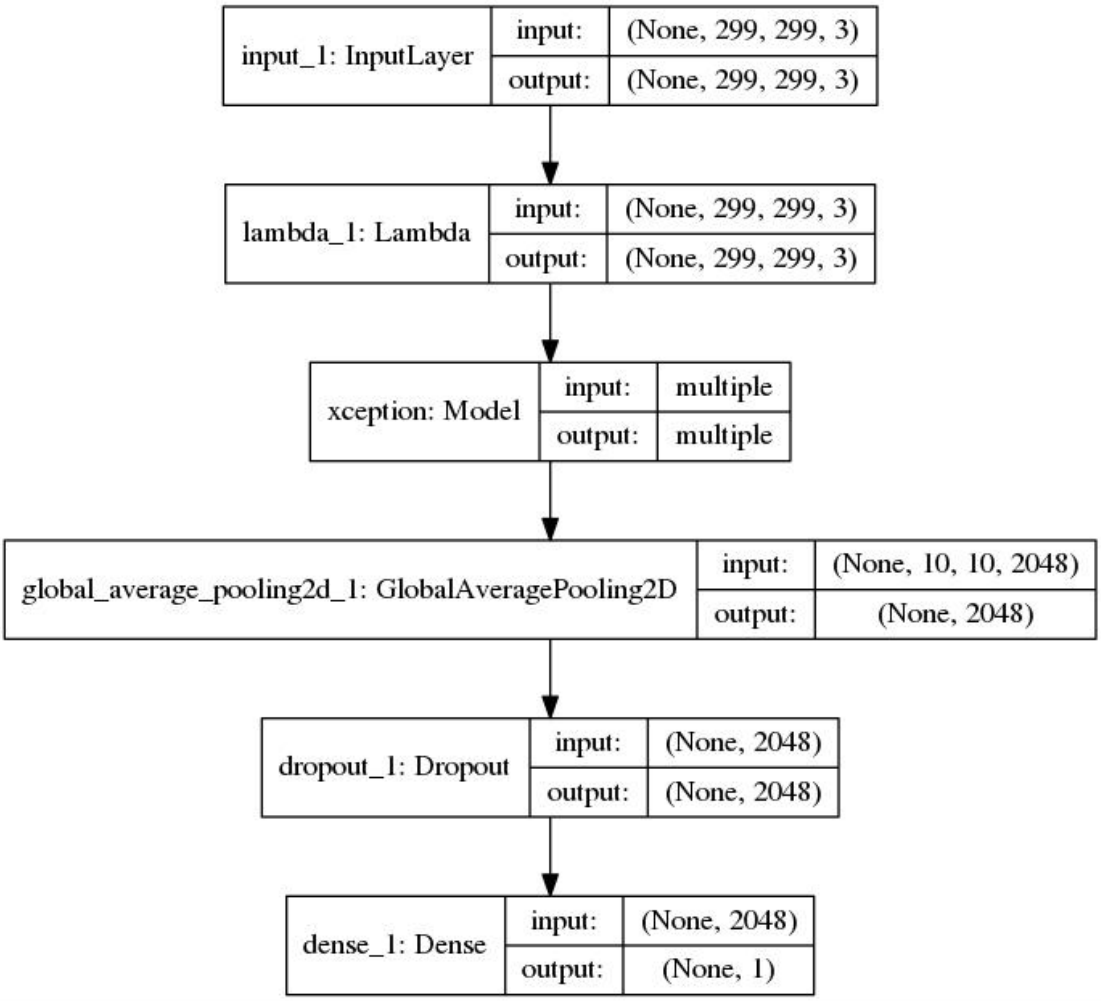

图 11. Fine-tune 模型

实验时先将 19916 张训练数据按猫狗等比例分割出 10%即 1991 张用于 Dense 层预训练，预训练方法同 3.3 使用的迁移学习逻辑回归，然后再使用剩余 90%即 17925 张数据进行卷积层微调。若使用预训练见过的数据微调卷积层，将会加剧过拟合程度。

3.4.2 实验结果

训练先使用 128 的 batch size，学习率1^{-3}的 Adam 优化方法进行预训练，使验证集 loss 达到 0.0179。

在进行 fine-tune 时，首先实验了使用 Xception 论文[7]中给出的参数配置：学习率 0.045（fine-tune 降低十倍至 0.0045）、每 2 个 epoch 衰减 0.94 的 SGD 优化器，但没有得到可接受的实验结果；经过反复实验，使用 0.8 的 Dropout、图

像增强、patience 为 10 的 Early Stop 以及 Adam 优化器，最终验证集 loss 达到 0.0098，测试集最佳得分为 0.04140。图像增强参数为：20 度以内随机旋转、0.2 比例水平/垂直偏移、0.2 以内错切弧度、0.2 比例缩放、随机水平反转；优化器参数为1^{-4}学习率和1^{-6}衰减率的默认衰减函数。

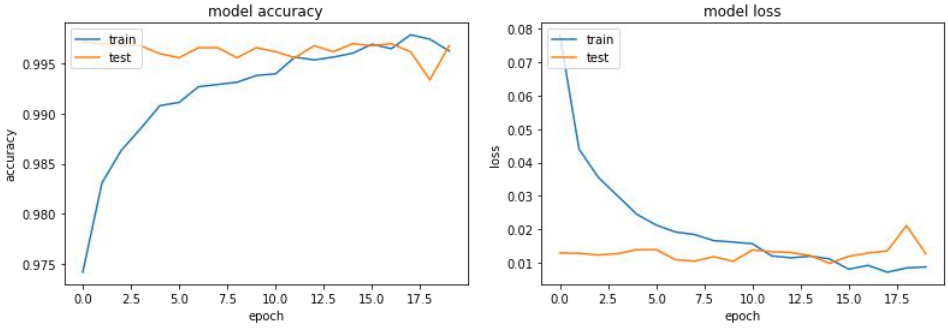

图 12. Fine-tune 学习曲线

可见，在本问题上 fine-tune 和图像增强对模型性能提升不大，并且极容易产生过拟合，考虑到其大量耗时性以及调参难度，不推荐优先使用。

3.5 多模型融合

3.5.1 模型搭建

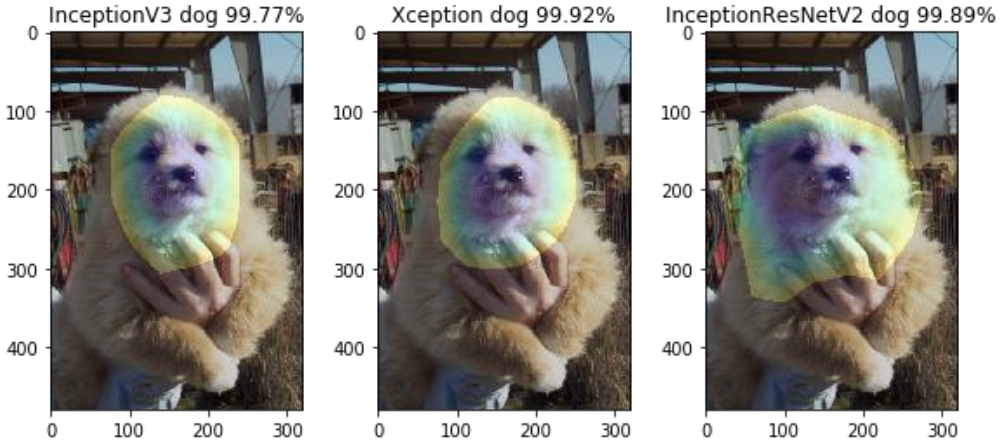

图 13. Class Activation Map

如图 13 所示，不同模型预测时关注的特征相似但不完全相同，在进行多模型融合时，可以将从不同模型提取的特征向量拼接合并为一个特征向量，以增加

模型性能。用于提取特征向量的模型结构同图 8，之后构建模型对特征向量拼接、Dropout 和逻辑回归。

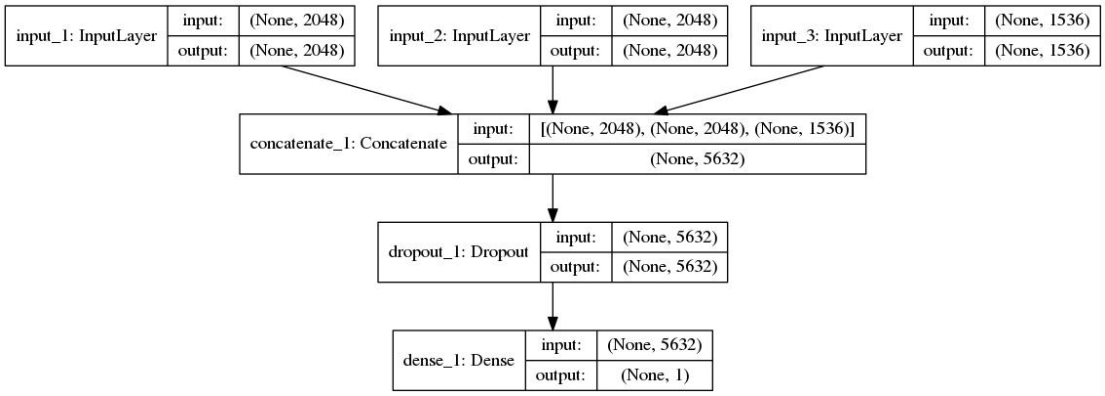

图 14. 多模型融合

3.5.2 实验结果

训练使用 0.5 的 Dropout，128 的 batch size，学习率1^{-3}、衰减率1^{-6}默认衰减函数的 Adam 优化器，patience 为 5 的 Early Stop，使验证集 loss 达到了 0.0063，测试集得到 0.03713。

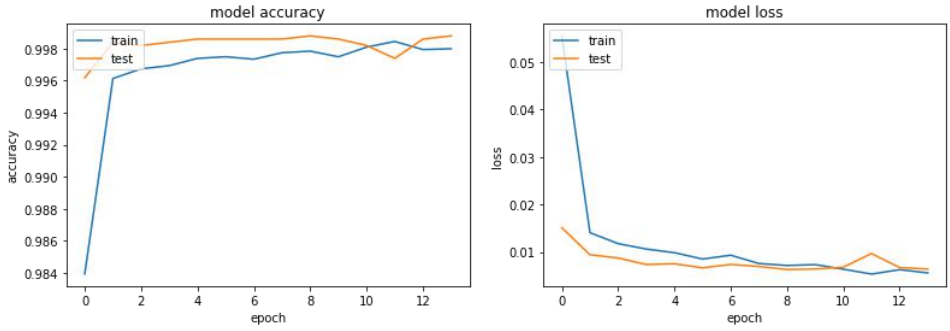

图 14. 多模型学习曲线

3.6 改进

使用 K-Fold 验证法，将 24895 个训练样本分割为 5 份，分别取每一份作为验证集，其余作为训练集训练 5 个模型，选择验证集 loss 最低的模型作为最终模型。由于数据混洗的原因，最终虽然在验证集得分只有 0.0087，但测试集得分为 0.03699，超过了改进前的性能，可见，K-Fold 验证法可以消除部分由于数据混洗、分割带来的偶然性。

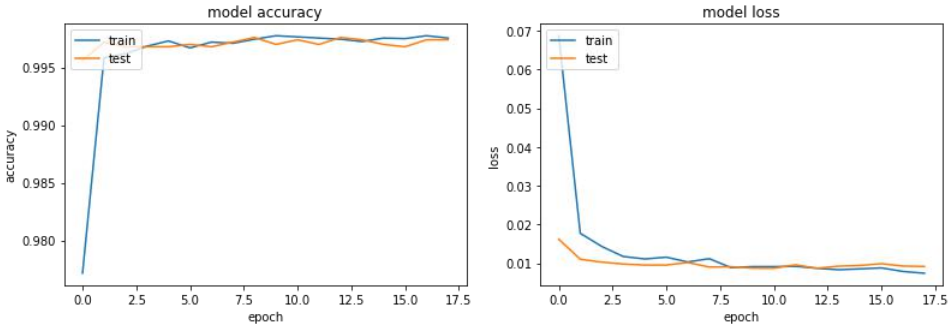

图 15. K-Fold 最佳模型学习曲线

4. 结论

（1）单模型迁移学习实验中，InceptionResNetV2 以 572 层的深度达到最低 loss，InceptionV3 以 159 的深度次之，可见模型性能随着深度的增加而变强；Xception 虽然只有 126 层，但相比 InceptionV3 的优势是参数更少，并且加入了残差结构，理论上更易于训练。

（2）猫狗问题相比与 ImageNet 太过简单，增加额外的隐藏层并没有带来太大提升，并且加剧了过拟合程度。

（3）图像增强和卷积层 fine-tune 方法对模型性能有一定提升，但幅度不大，原因可能在于猫狗数据集跟上属于 ImageNet 数据集的子类，fine-tune 过程太容易产生过拟合，相比其大量耗时性，在此问题上带来的提升并不划算。

（4）不同模型学习到的特征图会有差别，使用多模型特征融合相当于从多个不同的角度综合看待问题，性能提升很大。

（5）K-Fold 验证法能有效消除部分数据混洗、分割造成的偶然性，在简单模型中值得使用，若需要训练更复杂的模型（如 fine-tune 卷积层），其成倍增加的时间成本并不划算。

（6）实验使用单模型（Xception 需要进行 fine-tune）就达到了基准指标，最终实验结果远超基准指标，若需要继续改进，可以考虑如：尝试别的输出截断范围；继续增加融合模型数量；对多个模型卷积层进行微调后再融合训练；增加对训练数据裁剪填充数量；对测试数据集中尺寸异常样本也进行裁剪填充；获取更多训练数据等。可见大多方法都极为耗时，并且与实验方法流程类似。

参考文献

[1] Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning Book.

[2] Karen Simonyan, Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[3] Christian Szegedy, Wei Liu, Yangqing Jia. Going Deeper with Convolutions, 2014.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren. Deep Residual Learning for Image Recognition, 2015.

[5] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe. Rethinking the Inception Architecture for Computer Vision, 2015.

[6] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning, 2016.

[7] François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions, 2017.

[8] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, Kaiming He. Aggregated Residual Transformations for Deep Neural Networks, 2017.