机器学习纳米学位毕业项目

猫狗大战 (Dogs vs. Cats)

Chen Yifan

Aug 8th, 2018

1 定义

1.1 项目概述

是kaggle的一个竞赛项目，目标是建立一个模型，将给定的图片分辨为猫或狗。这是一个图像分类问题，是典型的计算机视觉问题。

图像分类是计算机视觉研究中的经典问题，基于图像分类的研究成果和方法广泛应用在诸如目标检测已经图像摘要生成等领域。从 2010 年开始举办的 ImageNet 大规模视觉识别挑战赛[^1] (ILSVRC)代表了这些领域的世界先进水平。2012年，以卷积神经网络[^2] (Convolutional Neural Network, CNN)为代表的深度学习方法开始在挑战赛中独领风骚；此后几年，基于CNN的神经网络模型不断有新的研究成果出现，并且连续几年获得挑战赛冠军，可见CNN在计算机视觉领域的巨大优势。

作为一个典型的图像分类问题，本项目计划使用CNN网络来构建模型，考虑到训练CNN网络需要用到巨大的计算资源，拟采用Keras的Applications模块提供了带有预训练权重的深度学习模型以减少对计算资源的要求. Keras是一个高层神经网络API，Keras由纯Python编写而成并基Tensorflow、Theano以及CNTK后端。Keras提供的应用于图像分类的预训练模型，其权重训练自ImageNet。

本项目采用Kaggle竞赛项目《Dogs vs. Cats Redux: Kernels 》的数据集，图片分为train和test两个数据集；train中共有25000张图片，其中猫和狗各有12500张 ，test中包含12500张未标注的图片. 对于test中的每一张图片，模型需要预测出图像是狗的概率(1.0 代表狗，0 代表猫).

1.2 问题陈述

Kaggle竞赛提供的数据是从真实世界采集的猫和狗的图片，图像分辨率差异较大，质量参差不齐，图片中猫和狗品质多样，颜色和姿态各异，背景多变，这些都增加了图像分类的难度。

项目的目标是建立一个模型，将给定的图片分辨为猫或狗，这是典型的图像二分类问题; 通过训练集中大量已经标注为猫或狗的图片对模型进行训练，用训练好的模型预测未知的图片，模型的输出为图片是狗的概率，概率为1.0表示狗，0表示猫 。

1.3 评价指标

本次猫狗大战(Dogs vs. Cats Redux: Kernels Edition)中使用logloss[2]作为评价指标, 分数计算如下:

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

其中:

- n是图片的数量
- $\hat{y_i}$是模型预测为狗的概率
- y_i 是类别标签，1对应狗，0对应猫
- log 是自然对数

LogLoss是一个连续值, 取值范围 是0至无穷大, 相比Accuracy, LogLoss能对模型提供更细致的评价. 在深度学习成为主流的今天, 模型对图像分类的准确率都非常高, 模型之间的性能差异较小, 使用LogLoss作为评价指标能以更细微的视角观察到模型性能之间差异

2 分析

2.1 数据探索

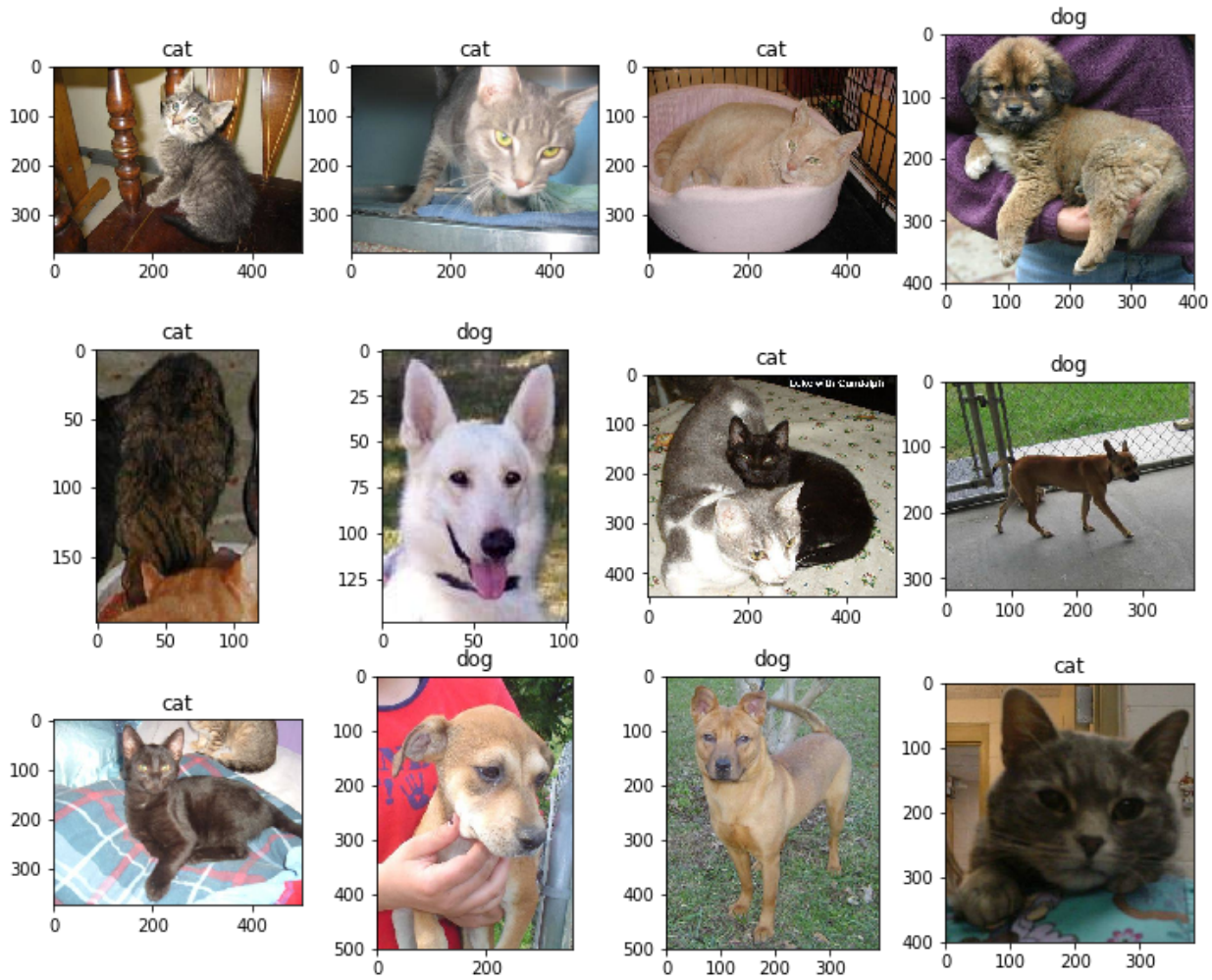
项目的训练集和测试集全部来自于Kaggle竞赛项目《Dogs vs. Cats Redux: Kernels 》，图片分为train和test两个数据集，绝大部分都是各种猫和狗的图片；train中共有25000张图片，其中猫和狗各有12500张；所有图片都已经在文件名中标注为猫或狗，如cat.xxx.jpg, dog.xxx.jpg, 由于猫狗比例相等，因此模型不需要考虑类别不平衡问题。

测试集中包含12500张未标注的图片, 图片内容都是各种猫和狗.

2.2 探索可视化

随机选取部分图片进行观察，图片基本都是RGB的彩色图片，包含不同品种的猫和狗，姿态，背景各异；其中有猫狗单独的照片，有人类牵引，拥抱的合照， 也有猫和狗的合照；大部分图片分辨率和质量较好，个别图片分辨率和质量较差。

- 训练样本展示



- 分辨率分布图

2.3 算法和技术

2.3.1 深度学习

深度学习作为机器学习算法研究中的一个新的技术，其动机在于建立、模拟人脑进行分析学习的神经网络。深度学习是相对于简单学习而言的，目前多数分类、回归等学习算法都属于简单学习或者浅层结构，浅层结构通常只包含1层或2层的非线性特征转换层，典型的浅层结构有高斯混合模型(GMM)、隐马尔科夫模型(HMM)、条件随机域(CRF)、最大熵模型(MEM)、逻辑回归(LR)、支持向量机(SVM)和多层感知器(MLP)。浅层结构学习模型的相同点是采用一层简单结构将原始输入信号或特征转换到特定问题的特征空间中。浅层模型的局限性对复杂函数的表示能力有限，针对复杂分类问题其泛化能力受到一定的制约，比较难解决一些更加复杂的自然信号处理问题，例如人类语音和自然图像等。而深度学习可通过学习一种深层非线性网络结构，表征输入数据，实现复杂函数逼近，并展现了强大的从少数样本集中学习数据集本质特征的能力。 深度学习通过学习一种深层非线性网络结构，只需简单的网络结构即可实现复杂函数的逼近，并展现了强大的从大量无标注样本集中学习数据集本质特征的能力。深度学习能够获得可更好地表示数据的特征，同时由于模型的层次深、表达能力强，因此有能力表示大规模数据。对于图像、语音这种特征不明显（需要手工设计且很多没有直观的物理含义）的问题，深度模型能够在大规模训练数据上取得更好的效果。相比于传统的神经网络，深度神经网络作出了重大的改进，在训练上的难度（如梯度弥散问题）可以通过“逐层预训练”来有效降低。

2.3.2 卷积神经网络

卷积神经网络（Convolutional Neural Network, CNN）是一种前馈人工神经网络，一般情况下，卷积神经网络由一个或者多个卷积层，池化层，激活函数以及顶端全连接层组成。与其他深度学习结构相比，卷积神经网络在图像和语音识别方面能够给出更好的结果；相比较其他深度、前馈神经网络，卷积神经网络需要考量的参数更少，使之成为一种颇具吸引力的深度学习结构.

- 卷积层(Convolutional Layer): 卷积神经网路中每层卷积层由若干卷积单元组成，每个卷积单元的参数都是通过反向传播算法最佳化得到的。卷积运算的目的是提取输入的不同特征，第一层卷积层可能只能提取一些低级的特征如边缘、线条和角等层级，更多层的网路能从低级特征中迭代提取更复杂的特征 .
- 池化层(Pooling Layer): 池化（Pooling）是卷积神经网络中另一个重要的概念，它实际上是一种形式的下采样。池化层有多种形式的池化函数, 如 “最大池化（Max pooling）”, "平均池化(Average Pooling)"等。以最大池化为例, 它是将输入的图像划分为若干个矩形区域，对每个子区域输出最大值。直觉上，这种机制能够有效

地原因在于，在发现一个特征之后，它的精确位置远不及它和其他特征的相对位置的关系重要。池化层会不断地减小数据的空间大小，因此参数的数量和计算量也会下降，这在一定程度上也控制了过拟合。通常来说，CNN的卷积层之间都会周期性地插入池化层。

- 全连接层(Fully Connected Layer): 是一个传统的多层感知器, “全连接”表示上一层的每一个神经元，都和下一层的每一个神经元是相互连接的。 卷积层和池化层的输出代表了输入的高级特征。完全连接层的目的是利用这些基于训练数据集得到的特征，将输入分为不同的类.

2.3.3 技术实现

项目使用深度学习库 Tensorflow 和 基于 Tensorflow 的高层神经网络API Keras 进行开发。

Tensorflow 是由Google 开发的开源机器学习库。最初由Google Brain小组开发，用于机器学习和深度神经网络方面的研究，但这个系统的通用性使其也可广泛用于其他计算领域.

Tensorflow 它是一个采用数据流图（data flow graphs），用于数值计算的开源软件库。数据流图用“结点”（nodes）和“线”(edges)的有向图来描述数学计算。“节点” 一般用来表示施加的数学操作，但也可以表示数据输入（feed in）的起点/输出（push out）的终点，或者是读取/写入持久变量（persistent variable）的终点。“线”表示“节点”之间的输入/输出关系。这些数据“线”可以输运“size可动态调整”的多维数据数组，即“张量”（tensor）。 理论上，只要能将计算表示为一个数据流图，就可以使用Tensorflow, 因此卷积神经网络也可以由Tensorflow 来搭建.

Keras是一个高层神经网络API，可以基于Tensorflow、Theano以及CNTK作为后端。Keras支持CNN和RNN, 或二者的结合, 提供了高度模块化和易于使用的API, 可以非常方便和快速的进行模型设计. 此外, Keras的Application模块提供了带有预训练权重的Kears模型, 如ResNet50, Xception, InceptionV3等, 这些模型可以用来进行预测,特征提取和finetune, 应用这些预训练模型可以极大的简化模型设计.

使用CNN处理高分辨率彩色图像所需的计算量非常大, 项目使用GPU进行计算加速. 对于没有GPU资源的个人用户, 可以选择 Amazon 提供的云计算服务 EC2 (Elastic Compute Cloud), 综合考虑性价比, 最终选择p3.2xlarge实例作为项目的运行平台, 该实例配备1个NVIDIA Tesla V100 高性能GPU, Intel Xeon E5-2686 v4 处理器, 61GB内存, 以及16GB现存.

2.4 基准模型

最近几年，在ImageNet挑战赛中涌现了许多优秀的卷积神经网络模型，如ResNet50, Xception, InceptionV3, InceptionResNetV2等。这些模型对ImageNet数据集上的1000个类别进行分类，最好的Top-5准确率高达0.92以上。 本次竞赛仅对猫狗进行分类，若想准确率到达0.92以上，则对数损失必须控制在0.1以内；kaggle排名前10%的成绩达到了0.06114, 排名100的成绩是0.05629，项目设定的目标是LogLoss分数小于0.056, 即进入排名100以内.

3 方法

3.1 模型选择

如表-1所示, 在 ImageNet[^3] 竞赛中取得优异成绩的CNN模型, 其深度常常超过100层, 从零开始训练一个类似的模型, 并让模型习得优异的性能, 不仅需要大量计算资源, 同时也需要巨量的训练样本来训练模型; 使用 Keras 提供的带有预训练权重的模型来构建迁移学习模型, 既能大幅减少对计算资源的要求, 提供训练效率, 同时也能减少因项目训练集不够而引起模型"过拟合"的风险.

Keras 提供的应用于图像分类的预训练模型，其权重训练自ImageNet. ImageNet数据集有超过1400万张被标注过的图片, 涵盖2万多个类别; 其中也包含许多猫狗的图片(包括狗品种118种, 猫7种), 利用训练自ImageNet的模型来预测猫狗类别是可行的. 在 Keras 提供的预训练模型中, ResNet50, Xception, InceptionResNetV2在实际应用中表现出非常好的性能, 且这些模型有公开模型结构和训练参数. 基于这些预训练模型, 采用迁移学习方法来实现本项目是一个可行的选择.

表-1

模型	大小	Top1准确率	Top5准确率	参数数目	深度
Xception	88MB	0.790	0.945	22,910,480	126
VGG16	528MB	0.715	0.901	138,357,544	23
VGG19	549MB	0.727	0.910	143,667,240	26
ResNet50	99MB	0.759	0.929	25,636,712	168
InceptionV3	92MB	0.788	0.944	23,851,784	159
InceptionResNetV2	215MB	0.804	0.953	55,873,736	572
MobileNet	17MB	0.665	0.871	4,253,864	88

表-1 列出了不同CNN模型之间性能(Top1/Top5准确率), 参数量以及深度的对比, 可以看到, 早起的VGG-16/19模型, 由于使用了较大的卷积核和较多的全连接层, 虽然网络深度不大,但参数量及计算量较大, 且准确率也逊色于后来的ResNet和Inception等一系列模型. ResNet50, Inception等模型, 分别设计了残差快(Residual Block) 和 Inception Module 的概念, 不仅减少了参数量,提升了运算效率,同时网络深度也提高了许多, 从而带来了性能的提升 (Top1准确率达到80%左右, Top5准确率达92%以上).

综合考虑各模型的表现, 项目选取ResNet50, Xcepiton, InceptionResNetV2这三个模型作为迁移学习的基础模型.

3.2 数据预处理

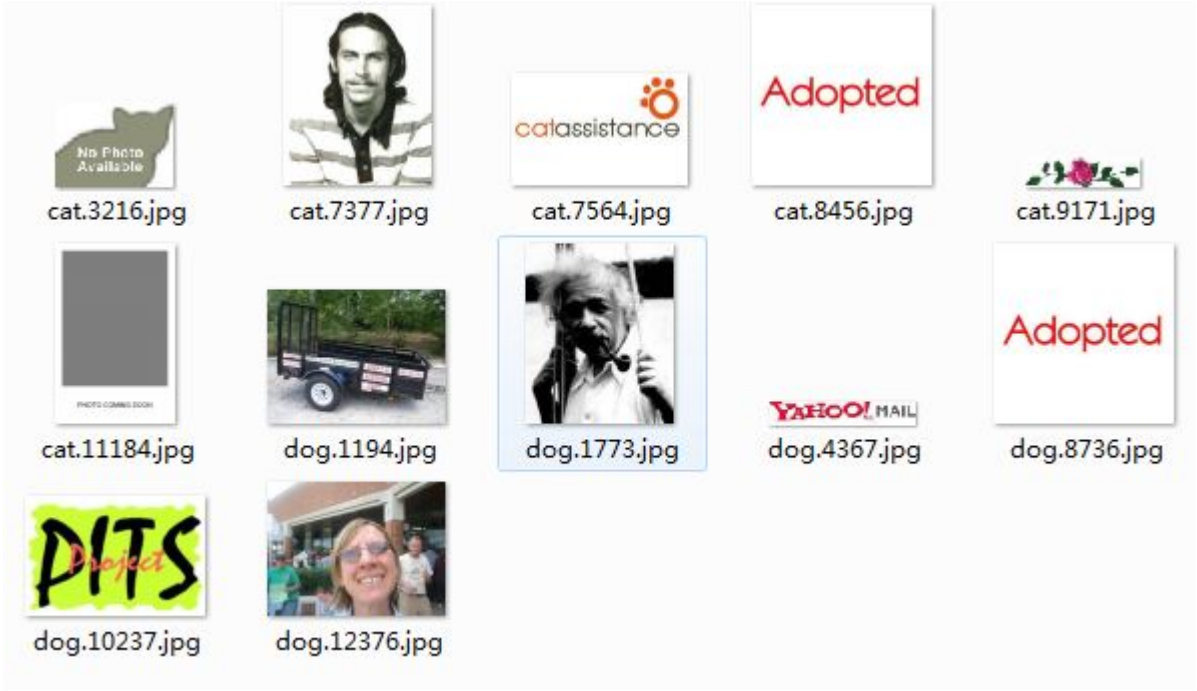
3.2.1 异常数据处理

异常值的存在会让训练误差增大，在训练模型前对训练集的异常值进行清理是很有必要的; 在ImageNet中，猫的品种有7种，狗的品种共有118种，使用在ImageNet上有着很高准确率的预训练模型对训练集的图片进行预测，分析预测结果的Top-N（N可以是20，30，50或更大的数）， 如符合下面两种情况，则当作异常值挑选出来：

- Top-N 里既没有猫，也没有狗
- Top-N 里只有猫，但图片被标注为狗，或Top-N 里只有狗，但标注为猫

使用预训练模型ResNet50, Xception, InceptionResNetV2分别对训练集进行预测，生成各自的异常图片集，然后取3个模型的并集， 经人工辨认后，最终确定61张图片为异常图片，并将异常图片从训练集中剔除。

非猫狗图片



被错误标注的图片

3.2.2 数据预处理

对数据集进行预处理以便将图像转换成模型能用的图像数据,并给训练集中的每个图像数据创建标签, Keras 的图片生成器 ImageDataGenerator 能方便地将图片转换成模型能用的数据. 为训练集的数据创建相应的标签. ImageDataGenerator 要求将不同种类的图片分在不同的文件夹中, 因此我们将数据集的路径结构做如下转换: 转换前:

```
|-- test [12500 images]
|-- train [25000 images]
```

转换后:

```
|-- pre-test
     |-- test [12500 images]
|
|-- pre-train
      |-- cats [12500 images]
      |-- dogs [12500 images]
```

不同的预训练模型对输入图片的大小有不同的要求, 本项目使用各模型默认的输入图片大小, 即:

模型	输入图片尺寸
ResNet50	224*224
Xception	299*299
InceptionResNetv2	299*299

3.3 执行过程

3.3.1 模型结构

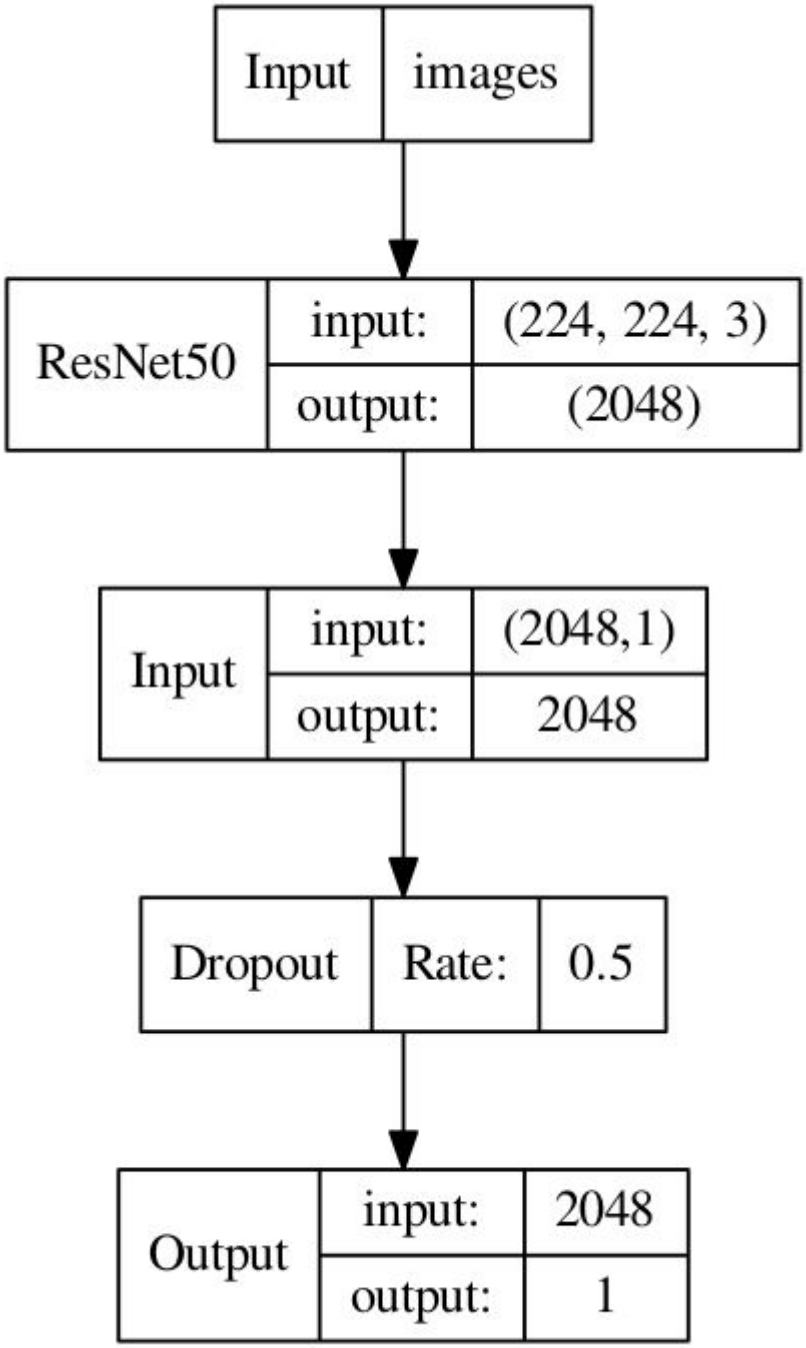
整体模型搭建如图3-3-1，具体如下：

1. 先将图片预处理为预训练模型的默认输入维度：

模型	输入维度
ResNet50	(224,224,3)
Xception	(299,299,3)
InceptionResNetv2	(299,299,3)

2. 然后使用一个预训练的模型（也使用其在imagenet上的预训练权重），对图片进行特征的提取；
3. 使用GlobalAveragePooling2D进行池化，是的输出特征的维度变为（nb_samples, 2048）；
4. Doupout层，防止过拟合；
5. 全连接层，激活函数为'sigmoid'，输出维度为（nb_samples, 1），也就是每个图片的输出都是（0,1）区间的一个数，就是进行分类。

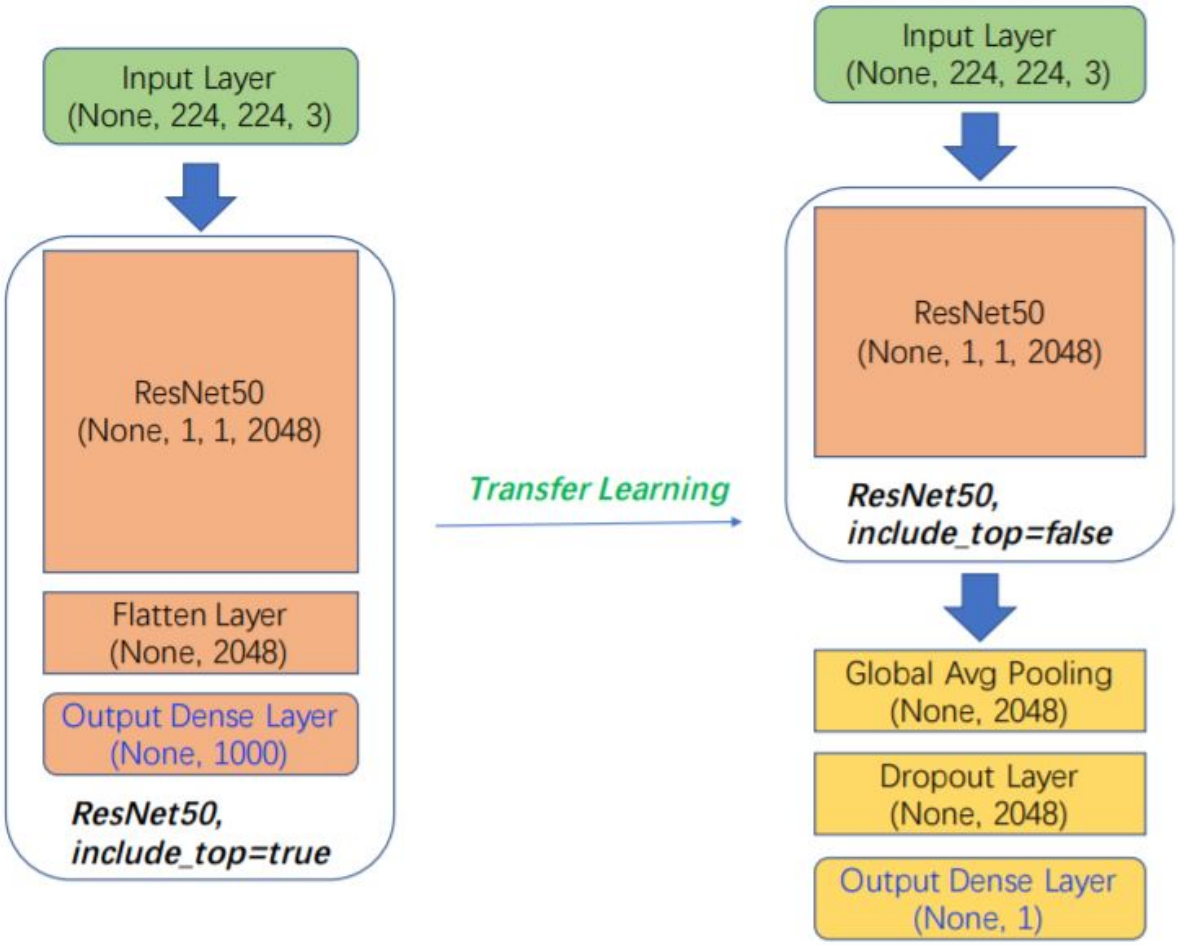
- 图3-3-1



3.3.2 特征提取

如前文所述，项目使用迁移学习方法，基于预训练模型进行微调，以ResNet50为例，其原理如图3-3-2, Xception 和 InceptionResNetV2 使用同样的方法构建模型

- 图3-3-2



以ResNet50为例，Keras Application 模块提供的ResNet50模型API， 传递参数 include_top = True 时，保留顶层的3个全连接网络。这个原始的预训练模型，输入为(224,224,3)的张量，输出层的激活函数是softmax，最终输出为ImageNet 1000类概率。 迁移学习是，传递参数 include_top = False，不使用预训练模型的顶层3个全连接层，输出的是预训练模型从训练集中提取的特征。

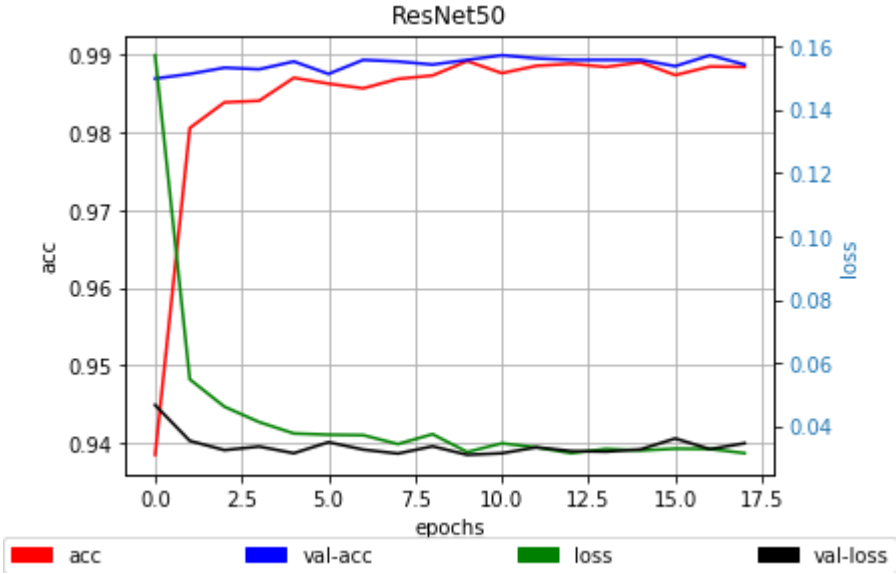
预训练模型输出的特征向量仍然很大， 为了减少特征数量，且降低过拟合的风险，对预训练模型的输出使用GlobalAveragePooling2D， 对输出的每个特征向量施加平均池化再保存为h5文件。

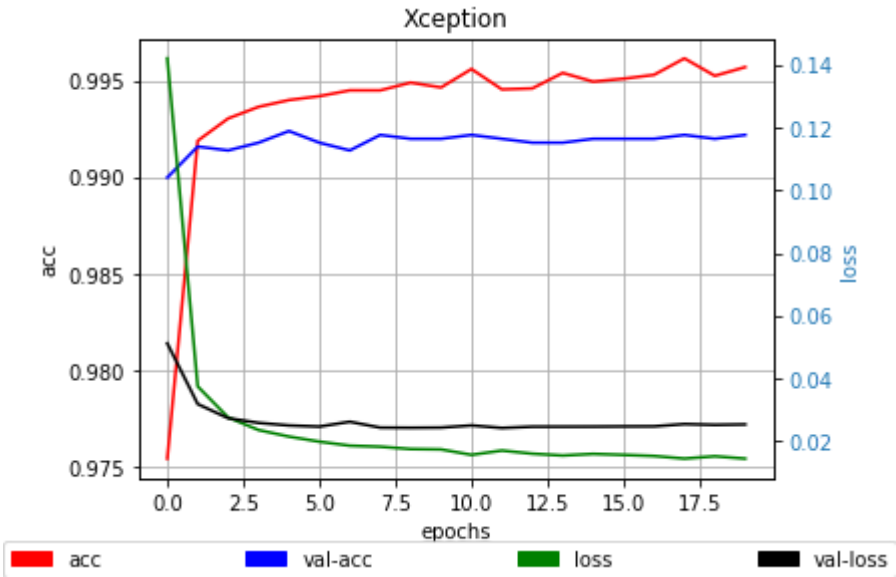
模型使用 predict_generator 从图像数据生成器上读取数据并提取特征。分别从训练集和测试集提取出特征并保存为h5文件。

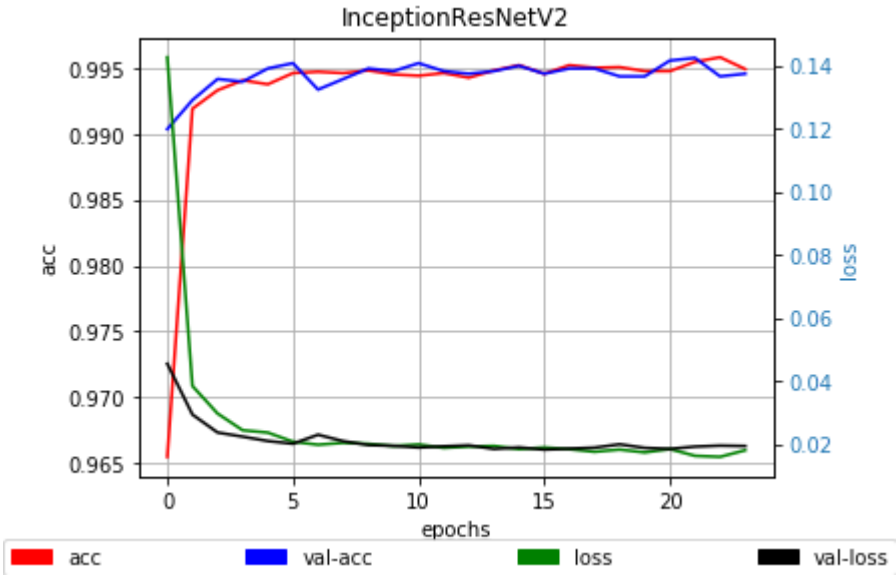
3.3.3 载入特征向量并训练

读取由预训练提取的特征后，为了在训练时有效划分训练集和测试集，先将提取的特征随机排序后，再在上面应用：Dropout层，Dense层，Dense层使用输出二分类的激活函数'sigmoid'，输出（0, 1）区间的单个值，正好用于分类及分类自信度的表达。优化器选择'adadelta', 损失函数选择'binary_crossentropy'.

设置验证集的大小为20%, 即每一轮训练，20%的数据作为验证集，80%数据作为训练集. 程序设置了EarlyStopping ， 监视 'val_loss', 当val_loss连续8轮没有改善时，则结束训练.


ResNet50
acc
loss
epochs
0.99
0.98
0.97
0.96
0.95
0.94
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
acc
val-acc
loss
val-loss


Xception
acc
loss
epochs
0.995
0.990
0.985
0.980
0.975
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
acc
val-acc
loss
val-loss


InceptionResNetV2
acc
loss
epochs
0.995
0.990
0.985
0.980
0.975
0.970
0.965
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0
5
10
15
20
acc
val-acc
loss
val-loss

从图3-3-3可以看出， InceptionResNetV2 模型的训练效果最好，因此项目将选择 InceptionResNetV2 作为最终的预训练模型，使用基于 InceptionResNetV2 的模型对测试集进行。

3.3.4 预测

读取测试集的特征文件后，输入到训练后的模型对进行预测，并将预测结果按要求写入csv文件中。由于输出为预测为狗的概率，评价指标是对数损失，因此对输出进行截取会稍微改善最终的结果。

使用这3个模型分别对测试集进行预测，将结果写入csv文件，上传到 Kaggle 竞赛项目，3个模型获得的分数如下：

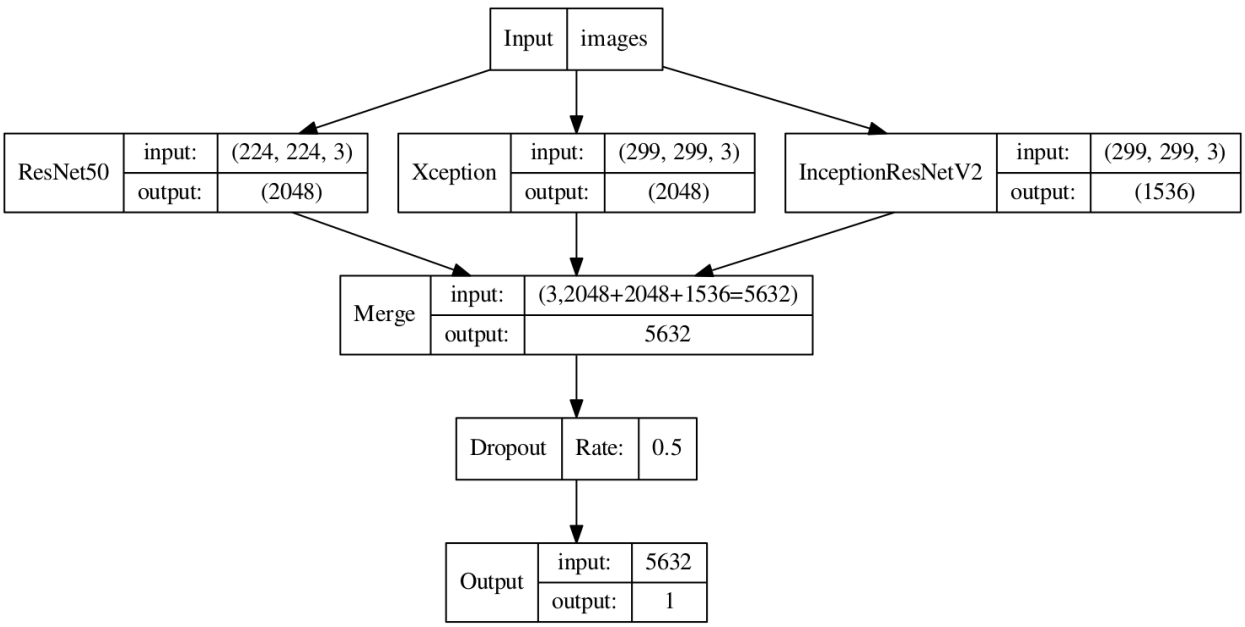
模型	分数
ResNet50	0.05470
Xception	0.04126
InceptionResNetV2	0.03847

InceptionResNetV2 模型的预测结果最好，获得的得分是 0.03847, 排名在12名。另2个模型的表现也很出色，完全达成了 LogLoss 分数小于0.056, 排名100以内的既定目标.

3.4 完善

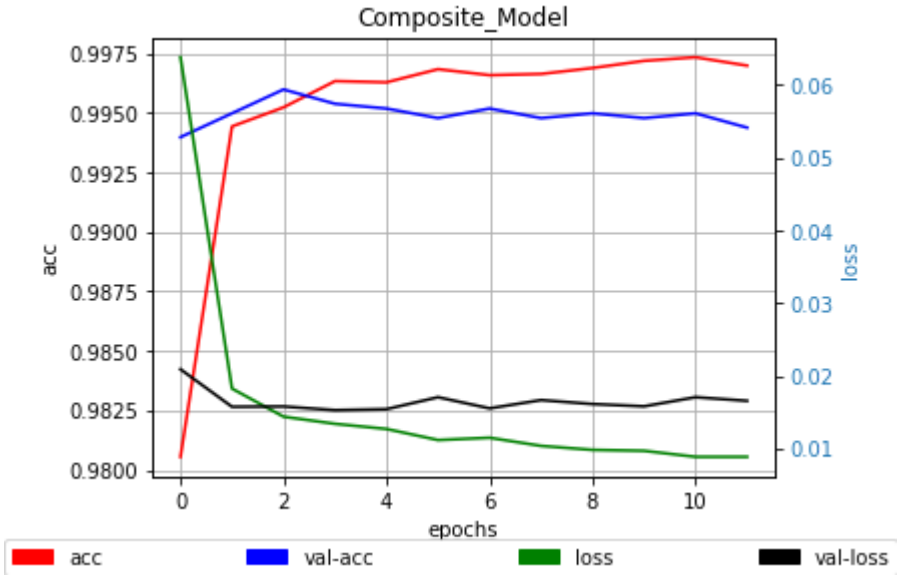
InceptionResNetV2 模型的效果最好，但三个单预训练模型的结果均达到了预期，下一步将使用多模型进行优化。这三个网络的网络结构差异较大，对图像特征的提取也必然存在差异, 从特征层面进行组合可以更全面地概括一个图形的内容，从而提高模型的性能。

组合模型结构如下图：



训练集和测试集经三个预训练模型分别输出2048维、2048维和1536维特征，将这些特征组合在一起产生新的样本，每个新样本包含5632维特征。分类器依然使用一个Dropout层加一个Sigmoid输出层， 用训练集对应新特征样本训练分类器，训练完成后对测试机的新特征集进行预测。

模型训练曲线如下：



4 结果

4.1 模型的评价与验证

从单个模型来看，项目使用的三个模型都属于卷积神经网络，他们构建网络的思路却不相同； ResNet提出了残差连接的概念，使网络用来训练残差而非卷积输出，优点是可以令网络具有非常深的深度的同时不至于难以优化，从而达到更好的性能； Xception 和 InceptionResNetV2 都是由Inception发展而来，其中，Xception基于Ineption V3 网络，其特点是采 用 一种称为深度 方向上的可分离卷积 方法，即先对每个通道分别卷积，再使 用1×1逐点进行 卷积，同时也应 用了 残差的结构；InceptionResNetV2主要利 用残差来改进Inception V3的结构，基本思想就是使 用 Inception module替代原本ResNet中的卷积单元 .

相比任何单一模型，从多个模型中提取特征并将其组合的方法可以全面利用不同网络提取的不同信息，因而可以得到更好的泛化能力。

- 表4-1

模型	分数
ResNet50	0.05470
Xception	0.04126
InceptionResNetV2	0.03847
Composition-Model	0.03649

4.2 结果分析

- ResNet50 模型得到 0.05470 的分数，在3个模型中表现最差。
- Xception 模型的得分是0.04126，显著优于 ResNet50, 提升幅度约 24.6%。
- InceptionResNetV2 模型的分数可以在排行榜中排名第 12 位，Xception 模型的分数能排在 18 位，而 ResNet50 模型的分数只能排在 92 位。

- 组合模型 Composition-Model 的分数为0.03649, 这个得分超过这个竞赛的第七名，同时比表现最好的单模型 InceptionResNetV2 得分提高了 5.1%, 在 InceptionResNetV2 排名已经非常靠前的情况，足可验证组合模型的性能非常出色。因此本项目选择组合模型作为最终模型。

5 项目结论

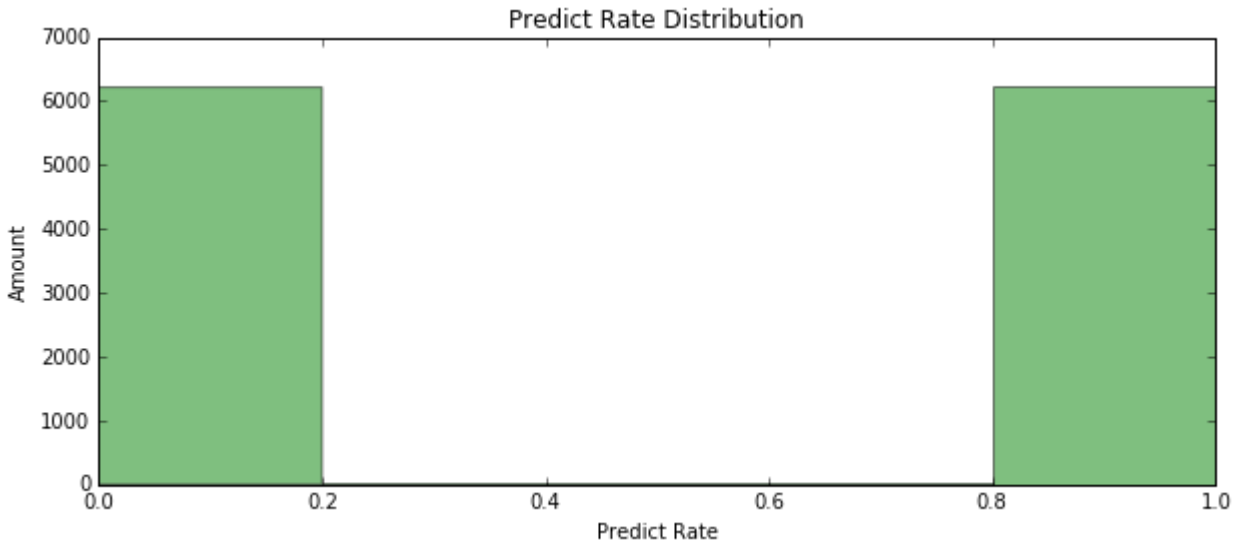
5.1 结果可视化

从测试集随机挑选 一些图片进行可视化，如图5-1。可以看到，对于大部分测试图片, 模型不仅判断正确, 而且对结果非常肯定。如 '3674.jpg', '6657.jpg' 等背景复杂的图片，模型也能给出非常肯定的预测。

- 图5-1 测试结果展示1

- 图5-1 测试结果展示2

预测概率的大部分都分布在 0~0.2, 和0.8~1.0的区间内，模型的表现已经达到了预期



5.2 思考

根据项目数据集的特点，使用迁移学习方法，基于在ImageNet上表现最好的预训练CNN模型，通过Keras完成数据预处理，构建、训练模型，完成预测，并基于结果进行优化。项目所使用的方法和流程，也适用于通用场景下解决图片分类问题。

使用迁移学习，相当于站在巨人的肩膀上可以走得更远。灵活运用Keras API是简单、快速、 高效完成项目的关键。

参考文献

[1] Large Scale Visual Recognition Challenge 2016 [2] Pierre Sermanet, Soumith Chintala and Yann LeCun. Convolutional Neural Networks Applied to House Numbers Digit Classification. The Courant Institute of Mathematical Sciences - New York Universit, 2013. [3] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens. Rethinking the Inception Architecture for Computer Vision. arXiv:1512.00567v3 [cs.CV] 11 Dec 2015. [4] Kaiming He Xiangyu Zhang Shaoqing Ren. Deep Residual Learning for Image Recognition. Microsoft Research, 2015 [5] Franc¸ois Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. arXiv:1610.02357v3 [cs.CV] 4 Apr 2017. [6] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv:1602.07261v2 [cs.CV] 23 Aug 2016