猫狗大战

# 一、定义

## 项目概览

猫狗大战（Dogs vs Cats）是源自kaggle上2013年的一个竞赛项目。这个项目主要是为了使用机器学习算法进行图片分类，尤其是深度学习。项目中提供了一个数据集，包括25000个已标定的数据和12500个未标定数据。通过这25000个已标定数据来构建相应的预测算法模型，并使用这个算法模型对12500个未标定的数据进行预测，并将预测的结果提交到kaggle上，以得到一个综合评分，以评价预测算法模型的优良水平。

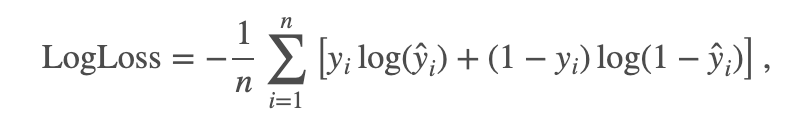
近些年，随着深度学习的突破性进展，整个机器学习领域也发展到了新的高度，尤其是卷积神经网络（CNN）在图片识别方面，其识别的准确性已经超过人类的识别水平。卷积神经网络可追溯到上个世纪80年代，Fukushima首次提出neocognitron模型，可以看作是卷积神经网络的第一个实现。经过若干年的发展，直到2006年深度学习的理论被提出后，卷积神经网络的表征能力得到了关注，并随着数值计算设备的更新开始快速发展。自2012年的AlexNet开始，卷积神经网络多次成为ImageNet大规模视觉识别竞赛（ImageNet Large Scale Visual Recognition Challenge, ILSVRC）的优胜算法，包括2013年的ZFNet，2014年的VGGNet和GoogleNet，2015年的ResNet等。

## 问题说明

猫狗大战提供的图片源自真实拍摄，分辨率差异较大。图片中猫和狗的颜色丰富，类型多样，姿态迥异，同时还有复杂的背景，极大的增加了识别的难度。这是一个典型的监督学习二分类问题。本项目将使用基于深度学习的卷积神经网络（CNN）来构建预测算法模型，通过25000个已标定数据（train data）来训练神经网络，然后使用训练好的神经网络模型来预测未标注的数据（test data），来完成预测准确度排名进入到kaggle上猫狗大战Public Leaderboard排名前10%的项目目标。

## 评价指标

在机器学习领域，通常会对算法模型设定损失函数作为评价指标，以便更好的评价算法模型的优良。常用的分类问题，都会使用交叉熵作为损失函数，猫狗大战属于二分类图像识别问题，需要使用了二分类交叉熵（binary\_crossentropy）作为损失函数。具体公式如下：



二分类交叉熵，需要在神经网络的最后一层使用sigmoid作为激活层来配合使用，以便可以对结果进行很好的分类。

# 二、分析

## 数据研究

为了规范化后面的算法使用，这里需要对下载的数据集进行解压，并满足如下目录结构。或者直接使用kaggle的api进行下载，也会满足如下目录结构。

├── bottleneck\_features.ipynb

├── README.md

├── requirements/

│   └── ...

├── sample\_submission.csv

├── test.zip

├── train.zip

└── transfer\_learning.ipynb

## 数据可视化

为了更好的对数据有直观的了解，将内存中的图片可视化出来。如下：



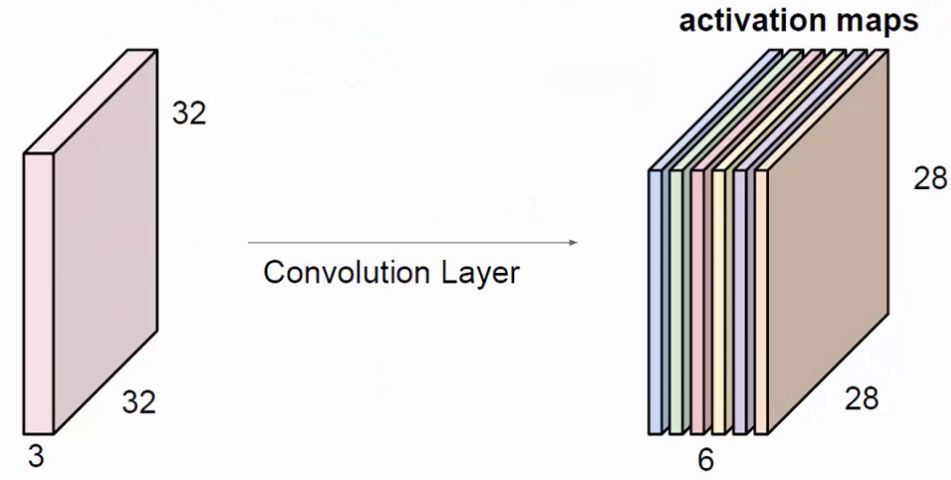
通过上图可以看出，训练所用的数据，尽管有各种复杂的背景，尽管猫和狗的姿势和形态也比较多样性，但整体来看图片都比较清晰，算是比较理想的数据集。

## 算法与方法

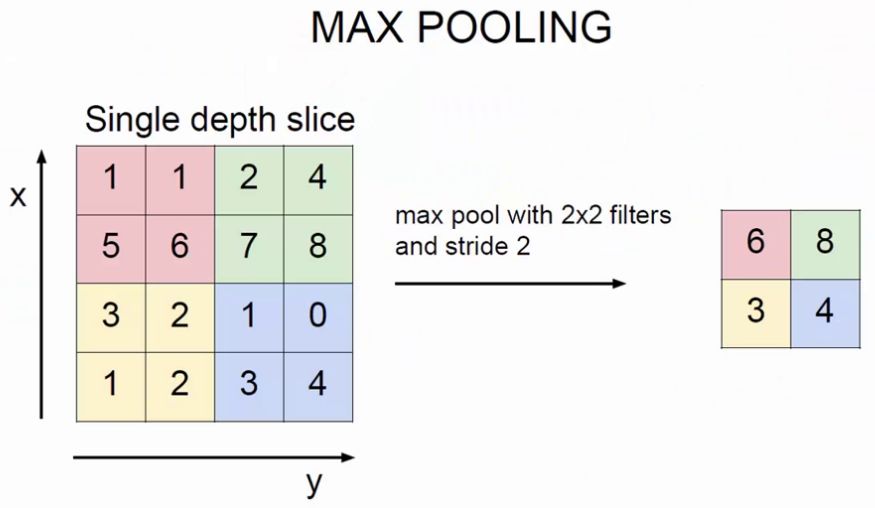
### 卷积神经网络

卷积神经网络（Convolutional Neural Network，CNN）是一种带有卷积结构的深度神经网络，属于监督学习中的一种。它的神经元可以响应一部分覆盖范围内的周围单元，对于大型图像处理有出色的表现。它包括：卷积层（convolutional layer）和池化层（pooling layer）。

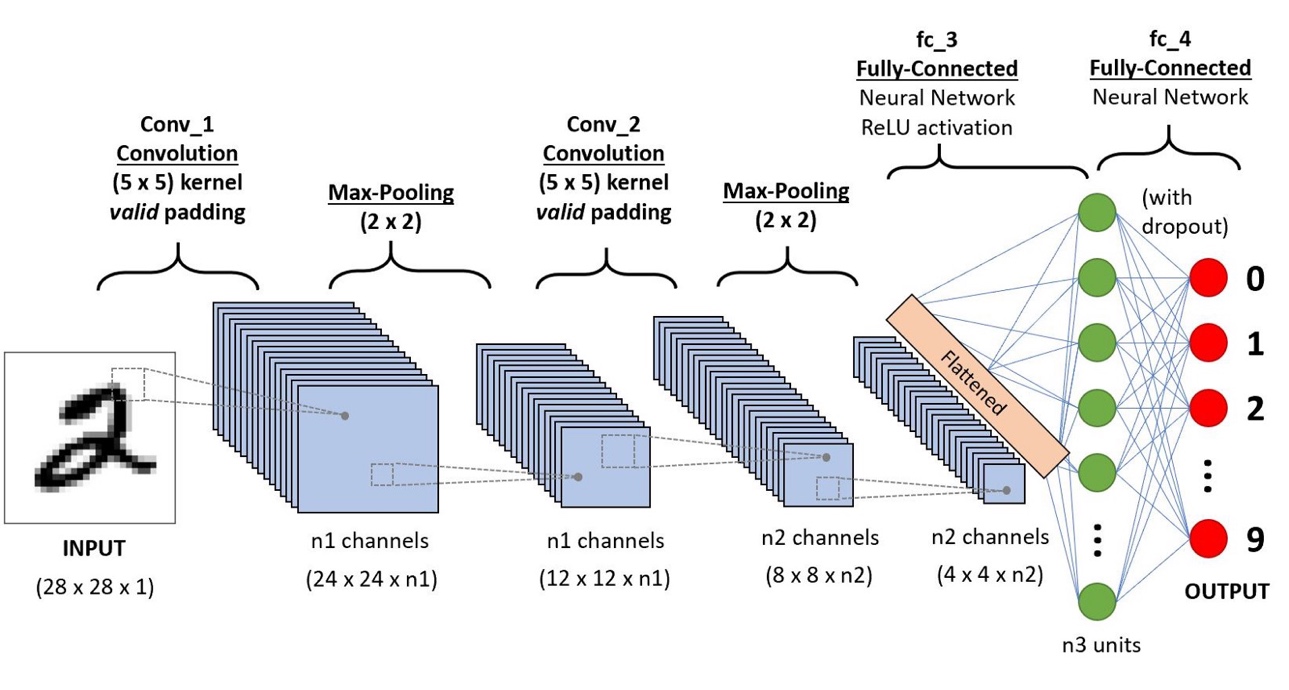
卷积，是在原始的输入上进行特征的提取。特征的提取简言之就上，在原始输入上一个小区域一个小区域进行特征的提取。



池化，就是对特征图进行特征压缩的过程，也称作降采样。选择原来的某个区域的max或mean代替那个区域，整体就浓缩了。



将卷积层和池化层结合起来，就得到了如下的过程。随着不断的卷积和不断的池化操作，最后得到了一个分辨率很低，但特征很多的图片，然后通过全连接

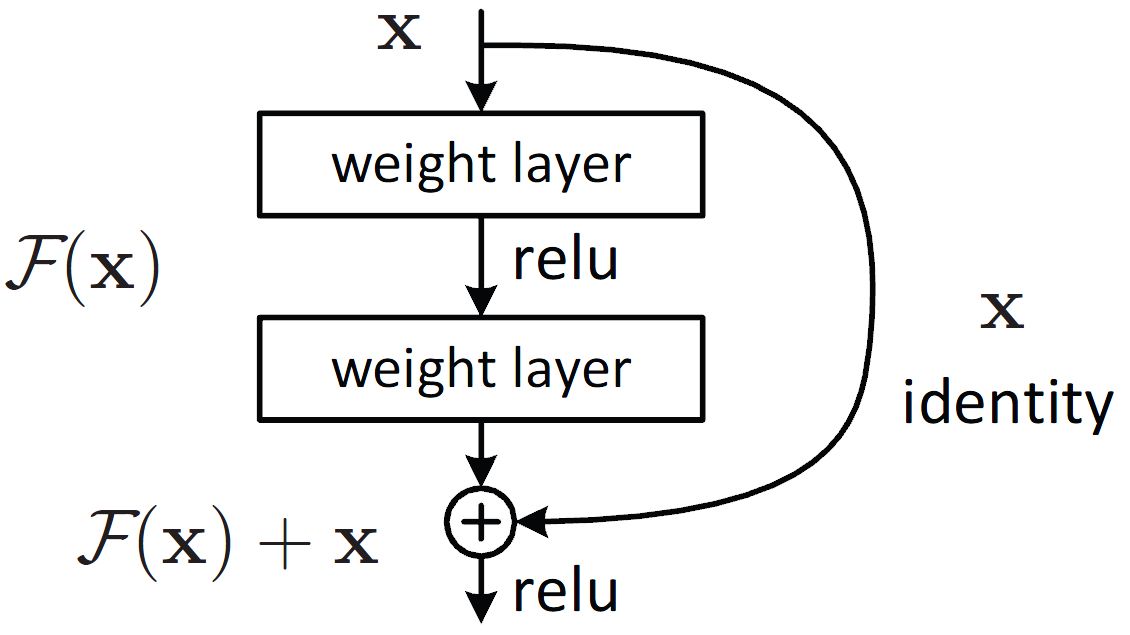


### 残差网络

ResNet（Residual Neural Network）由微软研究院的Kaiming He等四名华人提出，通过使用ResNet Unit成功训练出了152层的神经网络，并在ILSVRC2015比赛中取得冠军，在top5上的错误率为3.75%，同时参数量比VGGNet低，效果非常突出。

传统的卷积网络或者全连接网络在信息传递的时候或多或少会存在信息丢失、损耗等问题，同时还会导致梯度消失或者梯度爆炸，导致很深的网络无法训练。ResNet在一定程度上解决了这个问题，通过直接将输入信息直接输出到下一层，保存了信息的完整性。

ResNet网络上由若干个ResidualBlock和IdentityBlock构成的，比较常用的分别是Resnet50、Resnet101和Resnet152。



### 迁移学习

这里我们为了采用比较简单的Resnet50模型来构建我们的神经网络。鉴于整个网络的训练时间比较长，我们使用ImageNet的数据集预先训练好的模型，通过迁移学习来训练我们自己的神经网络。

所谓迁移学习，就是运用已有的知识对不同但相关领域问题进行求解的新的一种机器学习方法。在传统的分类学习中，为了保证训练得到的分类模型具有准确性和高可靠性，都必须满足两个基本的条件：

1. 新的数据集与原始数据集的相似程度；
2. 新的数据集的大小情况；

使用迁移学习总共有四种情况：

1. 新的数据集较小，且与原始数据相似度较高；
2. 新的数据集很小，但与原始数据相似度较低；
3. 新的数据集很大，且与原始数据相似度较高；
4. 新的数据集很大，但与原始数据相似度较低；

本项目中使用的dogs\_vs\_cats与ImageNet整体上来看，相似度还是比较高的，且dogs\_vs\_cats只提供了25000个训练数据（含验证数据）和12500个测试数据，样本偏小，比较适合第一种情况。

## 基准测试

按照udacity课程的要求，需要保证训练的模型的准确度在kaggle的Publish Leaderboard中的排名达到前10%。由于当前kaggle中的总排名人数为1314，所以要达到的排名需在1—131之间。根据排名的分数，测试集的评分需小于0.06127。

# 三、方法

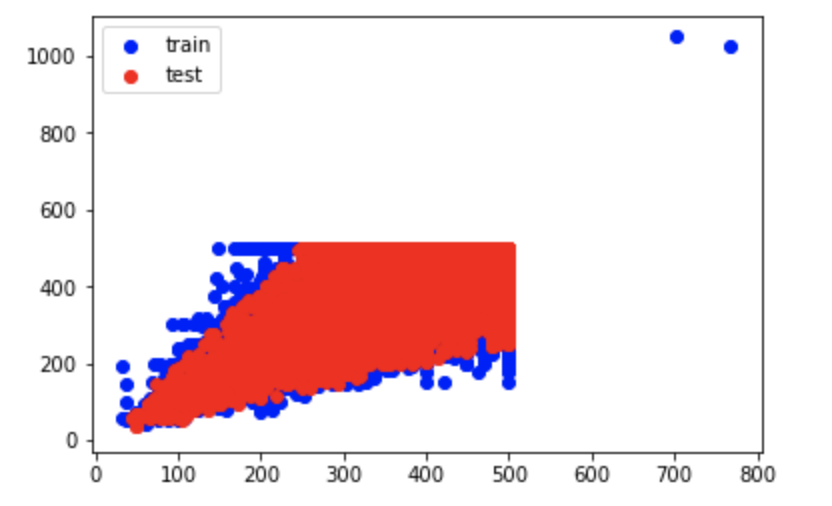
## 数据预处理

从目录结构中可以看出，train数据集的数据标注上包含在文件名中，因此需要获取数据集的相关信息和标注信息。另外，test数据集只提供了文件，并没有数据标注，需要将预测的结果提交到kaggle上才能获得评分。

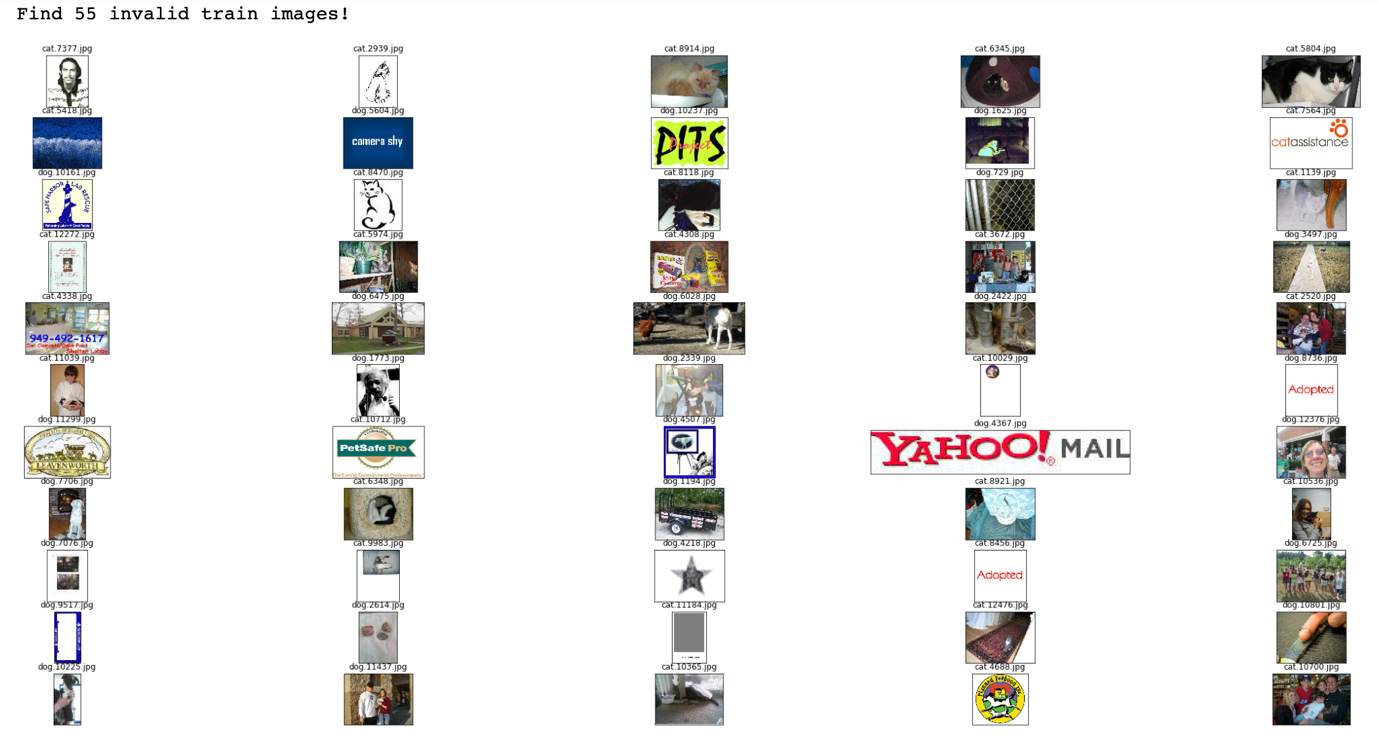
|  |  |
| --- | --- |
| 文件 | 标注（cat=0.0，dog=1.0） |
| datas/train/cat.6938.jpg | 0.0 |
| datas/train/dog.11432.jpg | 1.0 |
| datas/train/cat.433.jpg | 0.0 |
| datas/train/cat.11305.jpg | 0.0 |

*备注：这里需要值得注意的是，kaggle的submission文件中的id是按数字排序的（如：0，1，2，3），而加载数据集获取到的文件名通常是按字符串进行排序的（如：0，1，10，100）。*

通过文件大概查看，发现图片的分辨率都不一致，所以对图片的分辨率进行散点分布查看。通过绘制的散点图可以看出，大部分图片的分辨率都比较集中，除了两个特殊的图片。进一步确认后，发现只是分辨率比较大，毕竟所有的图片都会统一缩放为（224，224），所以不影响训练模型。



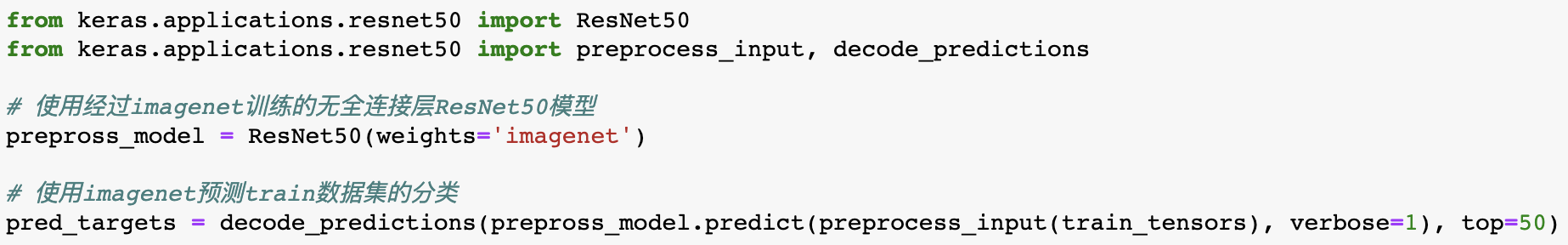
通常数据集中会包含一些错误的数据，会极大的影响模型的训练，由于项目本身是用来训练猫和狗的分类，所以对于非猫非狗的图片，就是错误的数据，需要剔除掉。下面是通过算法筛选出来的错误图片。



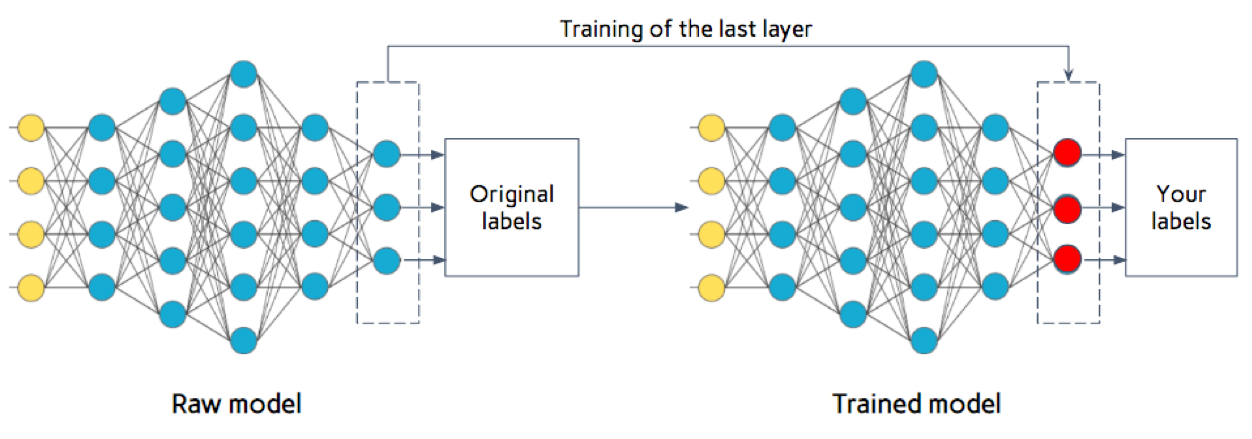
在进行算法建模之前，需要先把用到的图片数据加载到内存中，由于后续的神经网络模型，这里需要将图片分辨率统一为（224，224，3）。为了在训练神经网络模型时对模型进行验证，以避免出现过拟合的现象，需要将train数据集进行拆分，分为train数据集和valid验证集。

## 算法实施细节

在开始模型的训练之前，需要对训练集的数据进行预处理。首先，需要筛选错误的图片，以免影响模型的准确度。由于图片，难以通过常规的方法进行数据分析，这里使用了一种特别的方法。先使用完整的Resnet50模型，对训练集数据进行预测，由于Resnet50使用了ImageNet的权重，那么对于项目中的数据集会有一定的分类结果。对于非猫非狗的图片，果断剔除掉。为了尽可能的发现这些特征，我使用了top=50的参数。



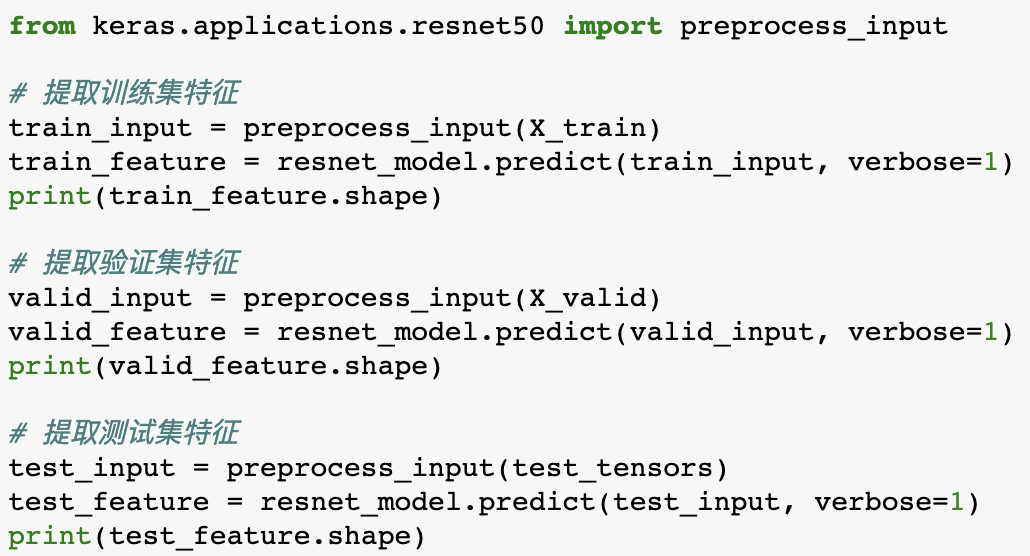
由于我们使用的预训练模型是基于ImageNet数据集进行训练的，而项目中的训练集和ImageNet的相似度比较高，同时项目中使用的数据集算是相对比较小的，所以需要对全连接层进行重新训练。



首先，构建Resnet50模型，使用ImageNet的权重初始化网络，并且去掉全连接层；使用keras接口时，只需要指定include\_top为False，就会自动去掉全连接层；将weights设置为imagenet，即使用ImageNet的权重初始化网络。



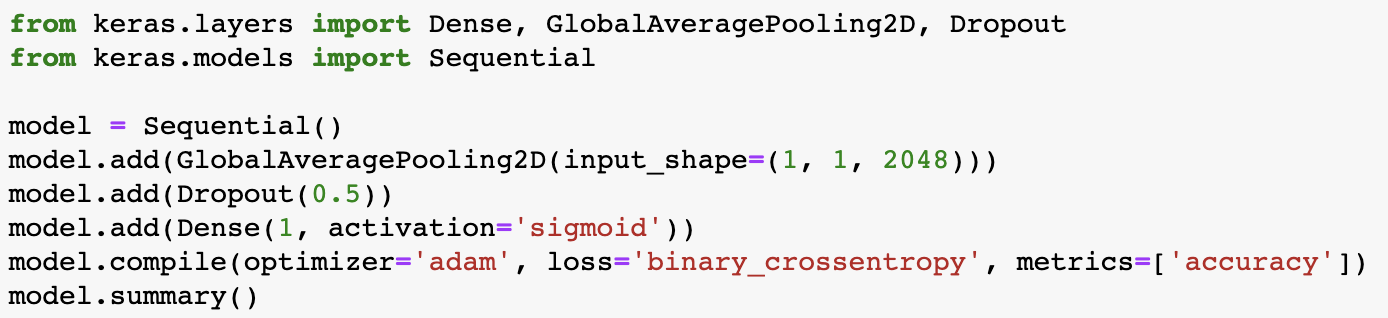
接着，将预先加载的训练集、验证集和测试集数据传入预训练模型，然后将输出的无全连接层的数据特征，以及训练集和验证集的标注信息存储到bottleneck文件中。这里使用了preprocess\_input来对图片的像素进行预处理，比如：将图片像素的值由0～255变换为-1～1内，进行逐样本均值消减的归一化等。



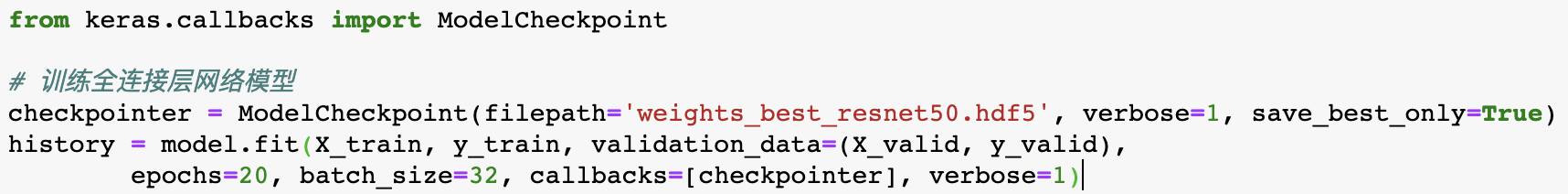
然后，构建新的全连接层神经网络，首先需要一个GAP层作为起始，然后连接一个Dense层，以便输出最终结果；通常为了避免过拟合，会在GAP之后加上Dropout层。



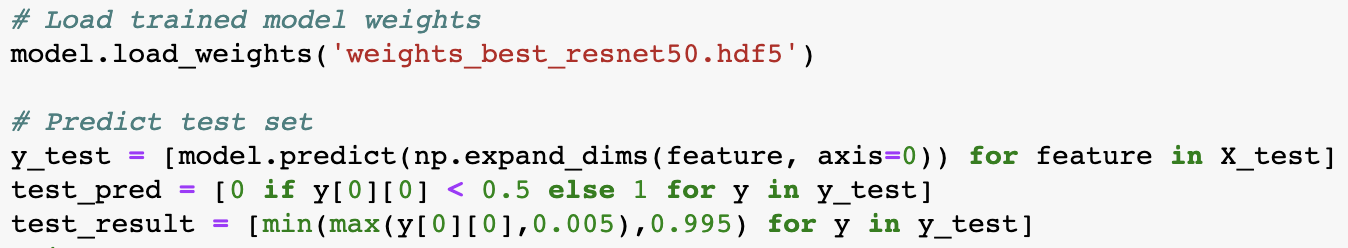
由于项目本身上一个监督学习的二分类问题，所以这里使用sigmoid作为最后一层的激活函数，同样也使用binary\_crossentropy作为损失函数；常用的优化器算法有SGD、Adagrad、RMSProp、Adam等可供选择，但由于Adam算法效果更好，收敛更快，所以这里直接使用了Adam算法作为优化器；由于去掉了全连接层的Resnet50所提取的数据集的输出特征数据的shape为（1，1，2048），所以GAP层的输入参数为（1，1，2048）；并且由于二分问题的类型，Dense的输出纬度为1。



将先前提取的训练集和验证集的bottleneck特征数据作为输入，进行网络的训练，并将训练的最佳权重保存到hdf5文件中。经过多次实验发现，基本上epoch=20和batch\_size=32足够得到最终的结果；并且由于在训练网络时，使用了ModelCheckpoint并指定参数save\_best\_only=True，可以将训练最好的权重保存下来。



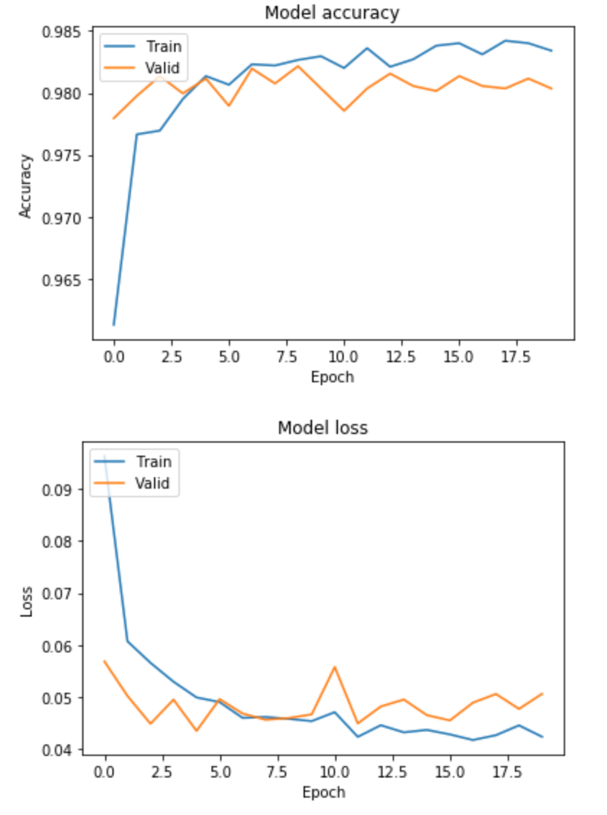
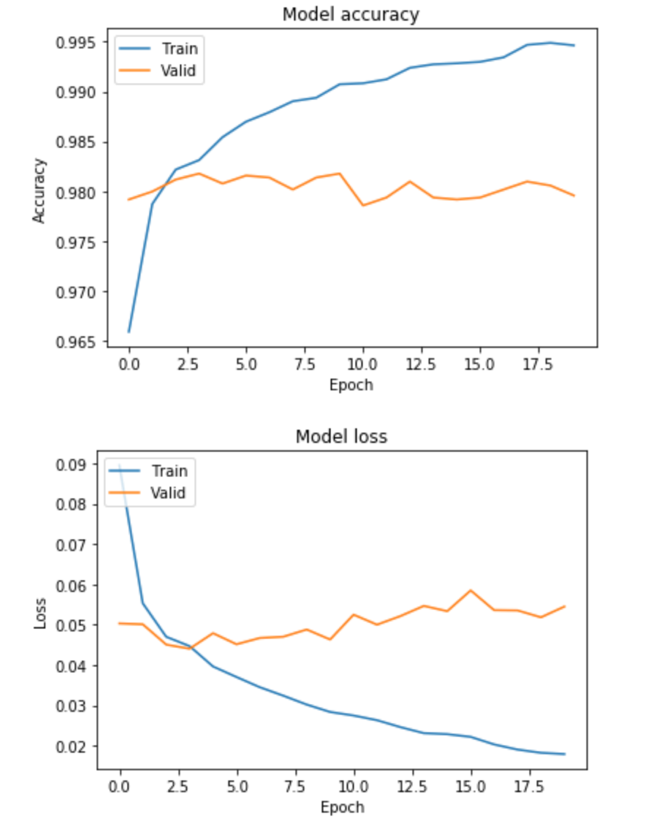
最后，使用训练好的神经网络，对先前提取的测试集的bottleneck特征数据进行预测，并将预测结果保存到submission.csv文件中。这里需要注意的是submission.csv模版数据中的id是和测试集的文件名是匹配的，要保证预测的结果和id是匹配的（详细在数据处理部分已经提到过）。



## 模型的改进方法

在新的全连接层增加了Dropout层。想要提高CNN的表达和分类问题，最直接的方法就是使用更深的网络和更多的神经元。但是复杂的网络也意味着更加容易过拟合。Dropout层会随机在网络的传递过程中丢掉一些神经元节点。这样在多次的训练过程中，会产生不一样的网络计算，可以有效的避免神经网络的过拟合。

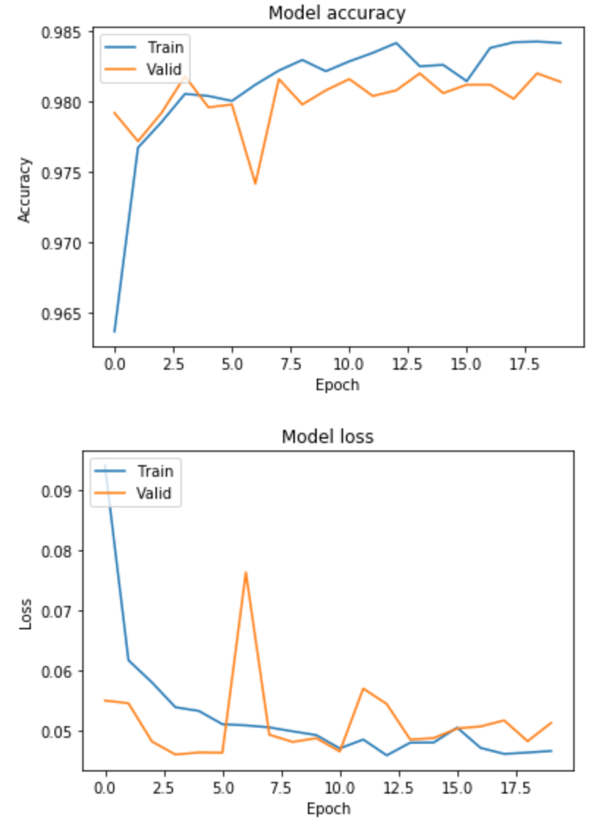
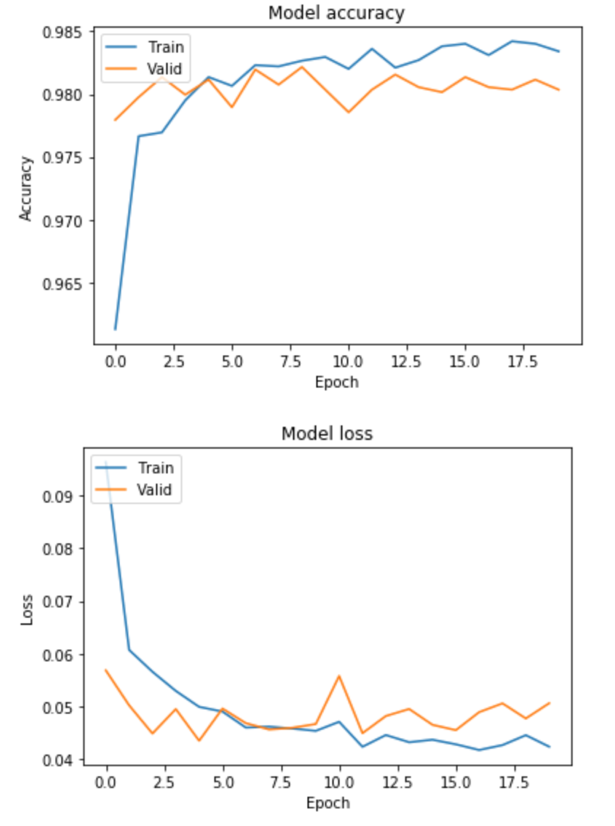
### 不加Dropout层和增加Dropout层的对比结果



**未使用Dropout** **使用Dropout**

从上图可以看出，左图没有使用Dropout层的明显出现了过拟合的情况，并未很好的收敛；右图虽然波动比较严重，但基本上收敛了。所以在模型中使用Dropout层。

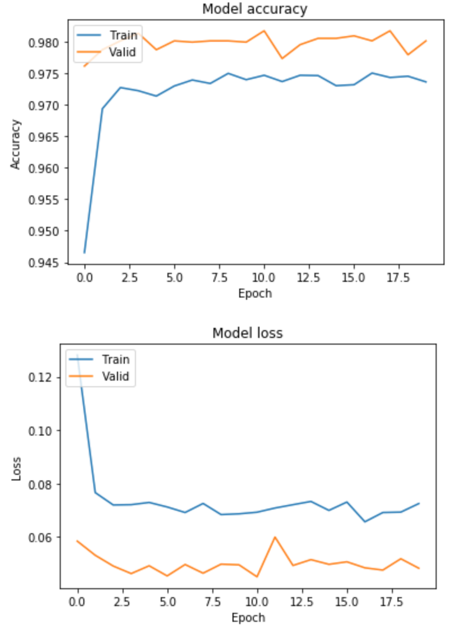
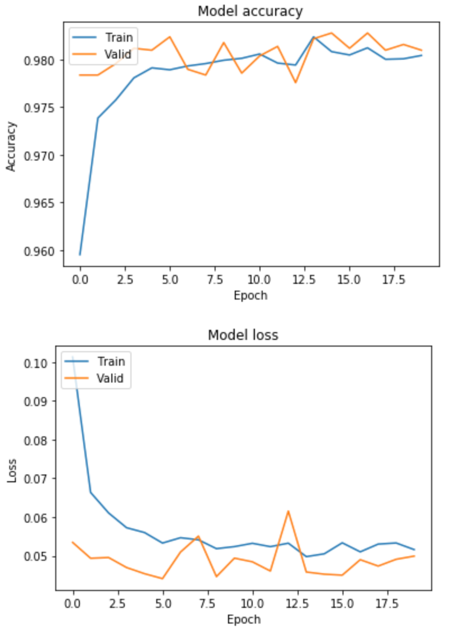
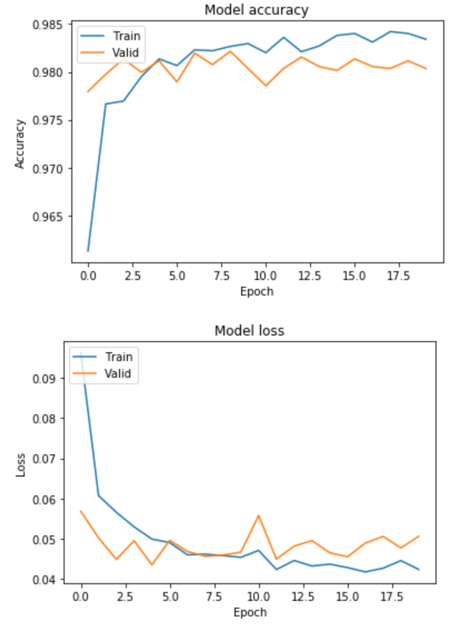
### 对比Adam优化器和RMSProp优化器



**Adam优化器 RMSProp优化器**

从上图可以看出，使用了Adam优化器在Dropout=0.2的情况下，表现会更好一些，所以优化器使用Adam算法。

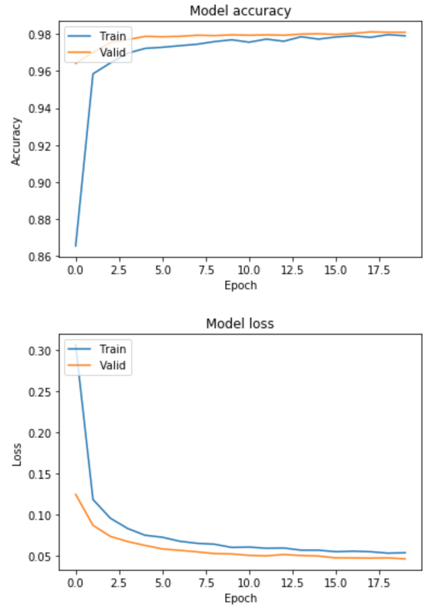
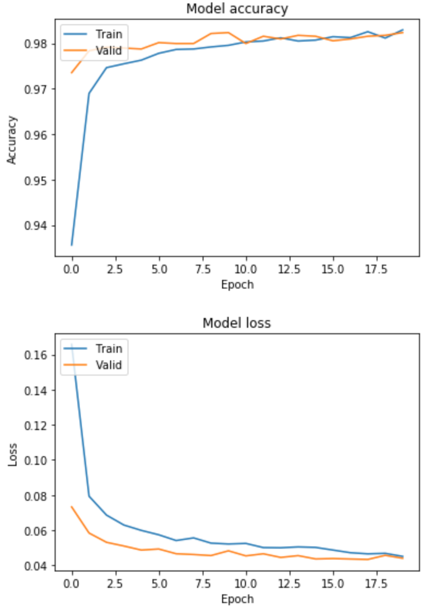
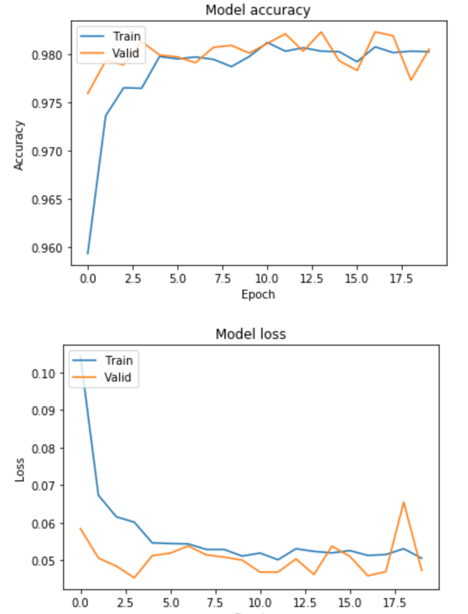
### 对Dropout进行调参（0.2，0.3，0.5）



**Dropout=0.2 Dropout=0.3 Dropout=0.5**

从上图可以看出，Dropout=0.2时，还是会存在一些过拟合的现象，而Dropout=0.5时，会出现无法收敛的情况；当Dropout=0.3时，收敛情况最佳。

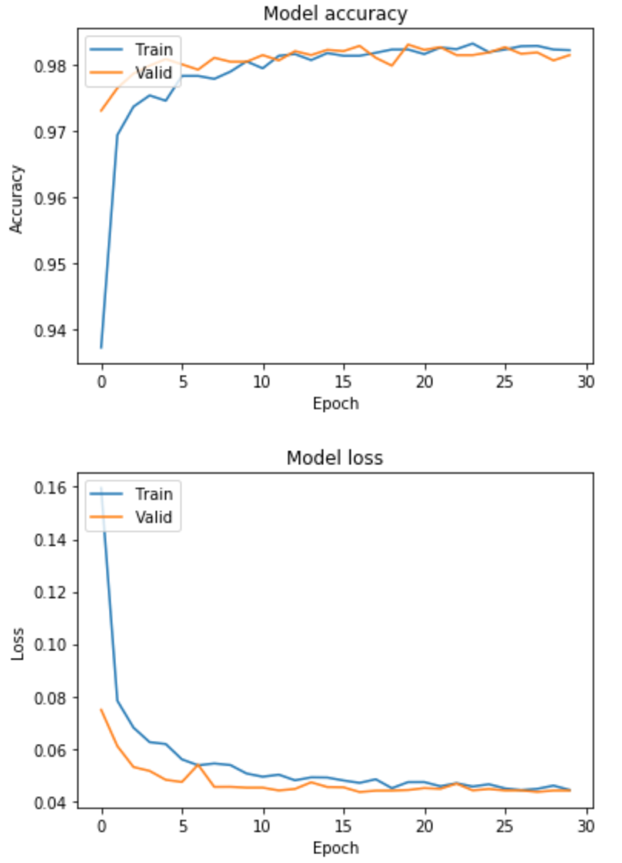
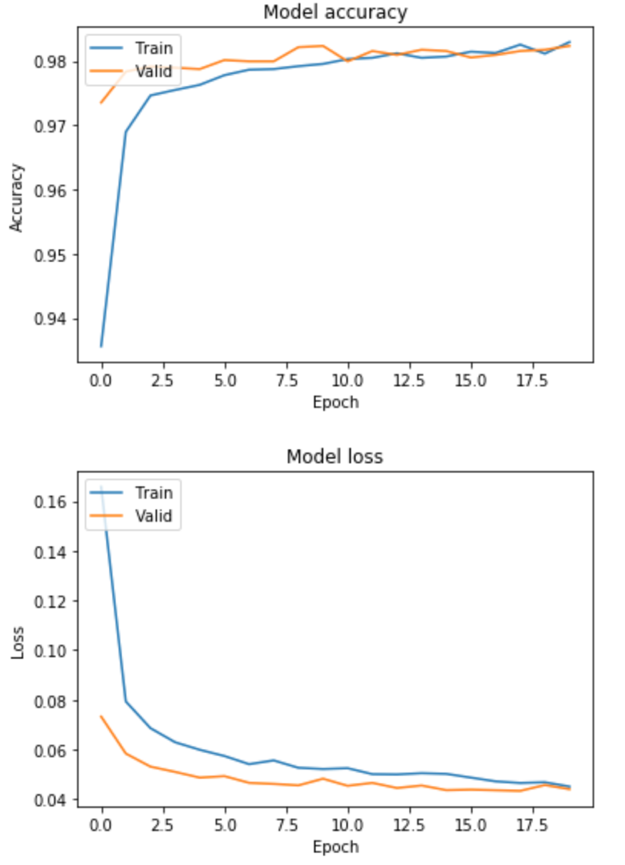
### 对Adam的lr进行调参（0.001，0.0003，0.0001）



**lr=0.001 lr=0.0003 lr=0.0001**

从上图可以看出，当lr=0.001时，收敛曲线波动比较厉害，说明学习步长太大了，当lr=0.0003和lr=0.0001时，基本上差别不大，由于lr=0.0003收敛情况更好，我们使用lr=0.0003这个参数。

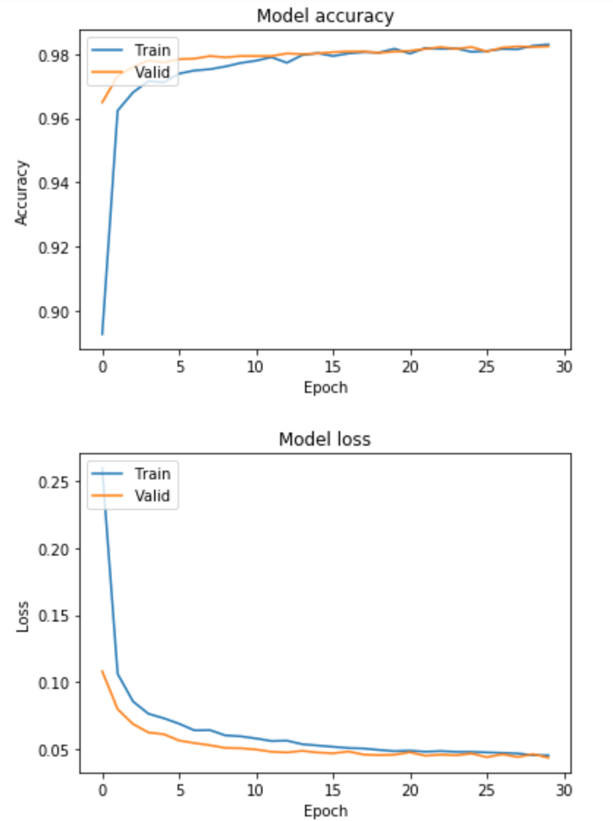
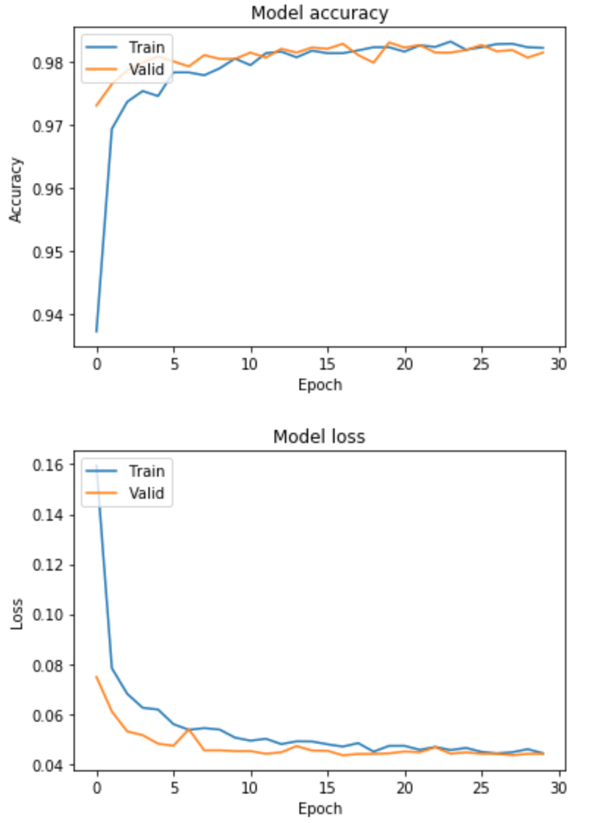
### 对epoch进行调参（20，30）



**epoch=20 epoch=30**

从上对比图可以看出，epoch=30的收敛情况更佳，所以使用epoch=30。

### 对batch\_size进行调参（32，100）



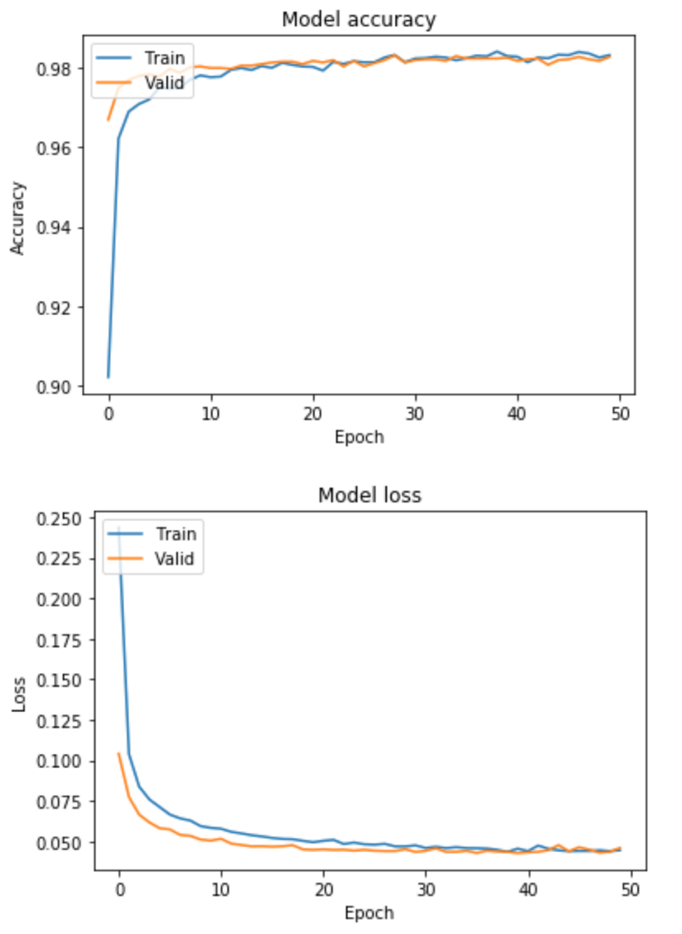
**batch\_size=32 batch\_size=100**

从上图可以看出，batch\_size=100时，不仅收敛情况良好，曲线也更平滑了，所以使用batch\_size=100。

# 四、结果

## 模型评估与验证

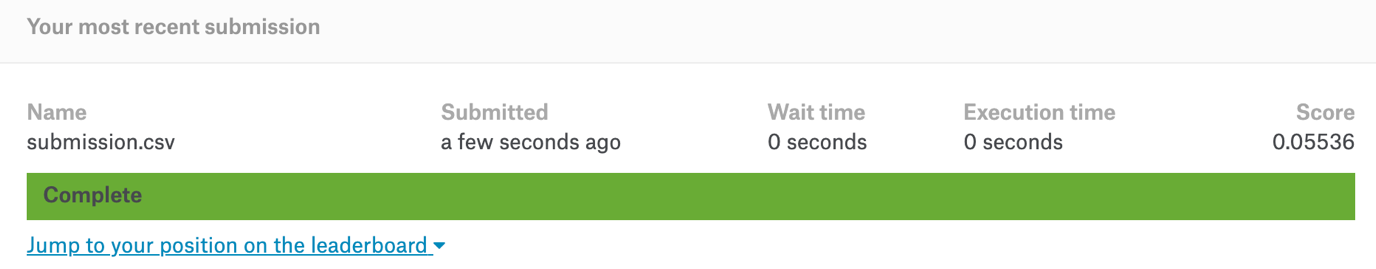
通过输出模型训练过程中的accuracy和loss的曲线，可以看出train数据集随着epoch的增加，准确度越来越高，接近饱和状态，而valid数据集随着epoch的增加，一直围绕饱和状态的train的准确度波动，说明模型训练前，预训练的模型准确度已经非常高了，并且随着不断训练，并没有出现过拟合的状态；train数据集的损失度随着epoch的增加，出现了大幅的降低，并趋于饱和，而valid数据集随着epoch的增加，虽有波动，一直稳定在train的稳定状态，说明预训练的模型的loss非常低了，并且没有随着训练的增加，出现loss大幅增加的变化，说明并未出现过拟合。曲线的变化基本上符合预期。



为了通过可视化的方式验证模型的预测结果，我们共测试集中随机选取了32个图片进行预测。从预测结果来看，准确度很高。



将测试集预测结果提交到kaggle上，最终的预测结果评分为0.05336，小于基准测试标准的评分0.06127，并通过可视化来查看测试集的预测结果，错误率很低。



# 五、结论

从测试集的数据预测情况来看，发现图片的角度和背景的多样性，并没有影响预测的准确性。尽管迁移学习的是ImageNet的权重模型，但通过重新训练全连接层的神经网络，对于dogs\_vs\_cats的数据集，依旧会有很好的表现。

其实，对于模型的准确度的提升，还是有很多提升空间的，可以尝试使用Resnet152来迁移学习，会得到更加准确的预测数据；同时，还可以对数据集中的图片数据做数据增强，以增加数据的多样性，以获得更好的模型。当然，目前也存在一些做法，将多个模型进行融合，以便获得更好的准确度。

# 参考文献

[1] Sumit Saha, A Comprehensive Guide to Convolutional Neural Networks.

[2] CS231n Convolutional Neural Networks for Visual Recognition.

[3] Jason Yosinski, Jeff Clune, Yoshua Benjio, and Hod Lipson, How transferable are features in deep neural networks.

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

[5] Jason Brownlee, How to Check-Point Deep Learning Models in Keras.

[6] Aaditya Prakash, One by One [1 x 1] Convolution - counter-intuitively useful.

[7] Francois Chollet, Building powerful image classification models using very little data.

[8] 凌蓝风，毕业设计Dogs vs Cats For Udacity P7 （异常值检验）