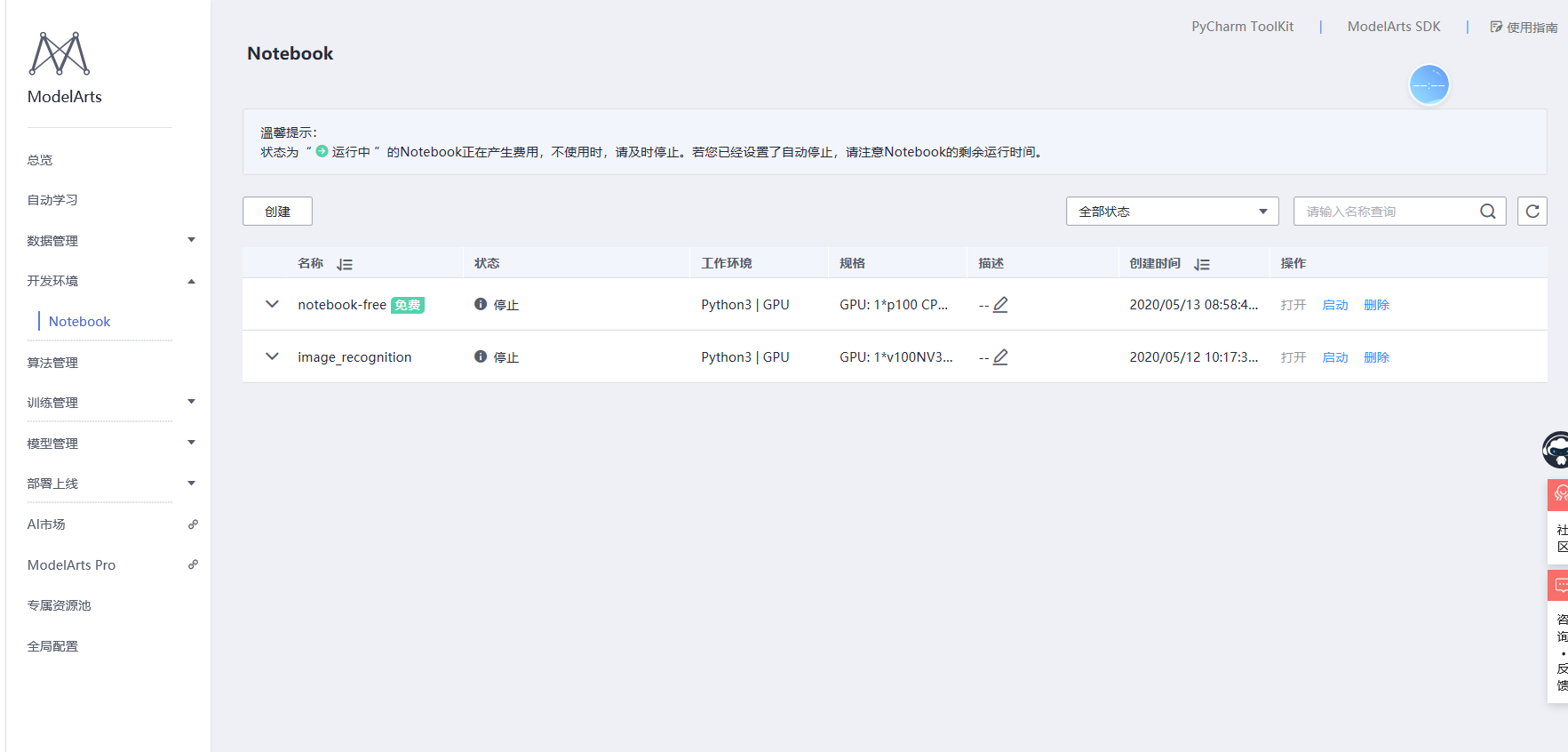
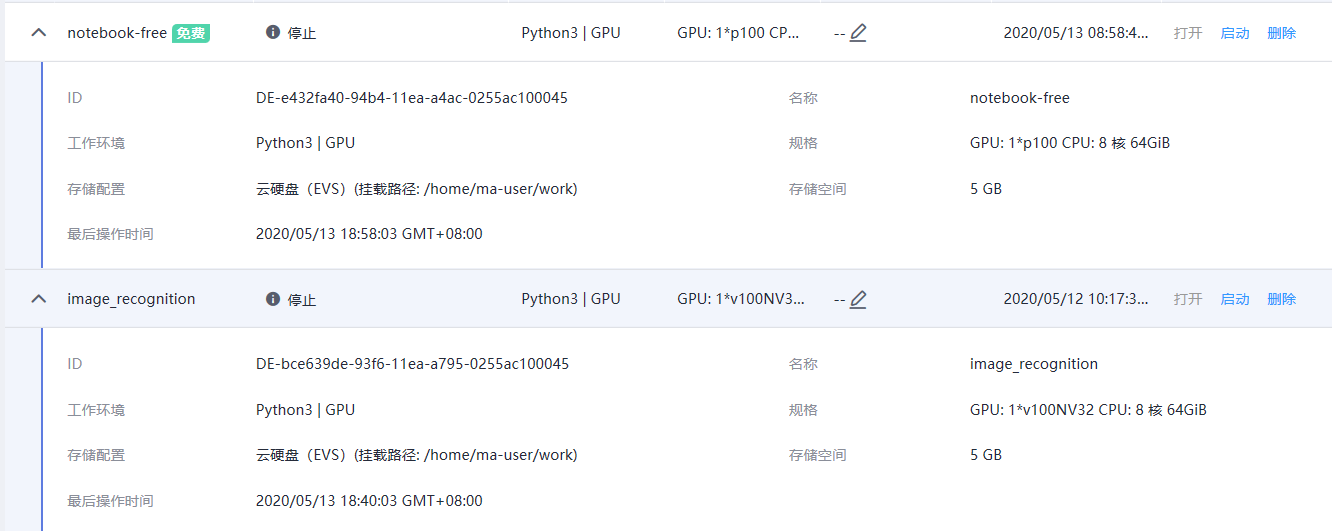
## Notebook案例-猫狗图像识别实践

### 1. 猫狗大战基本实验

#### 1.1 环境搭建

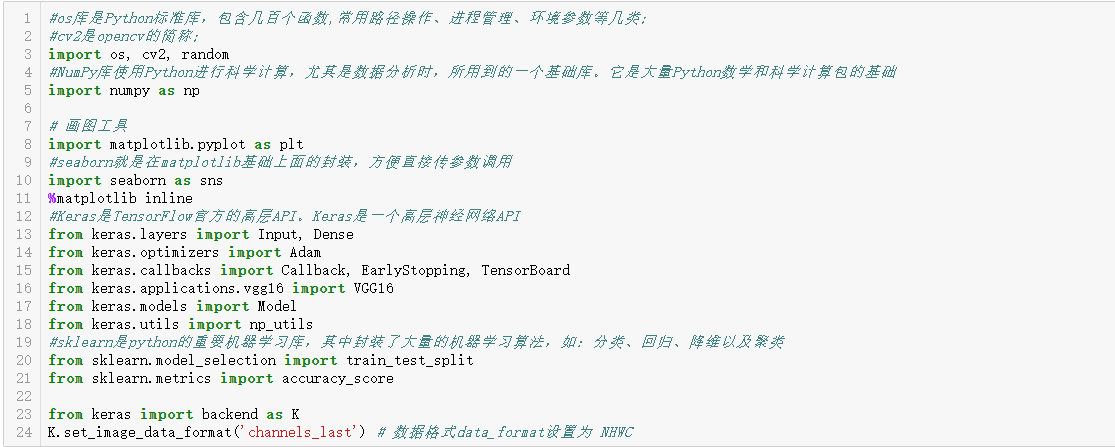


创建 notebook 开发环境

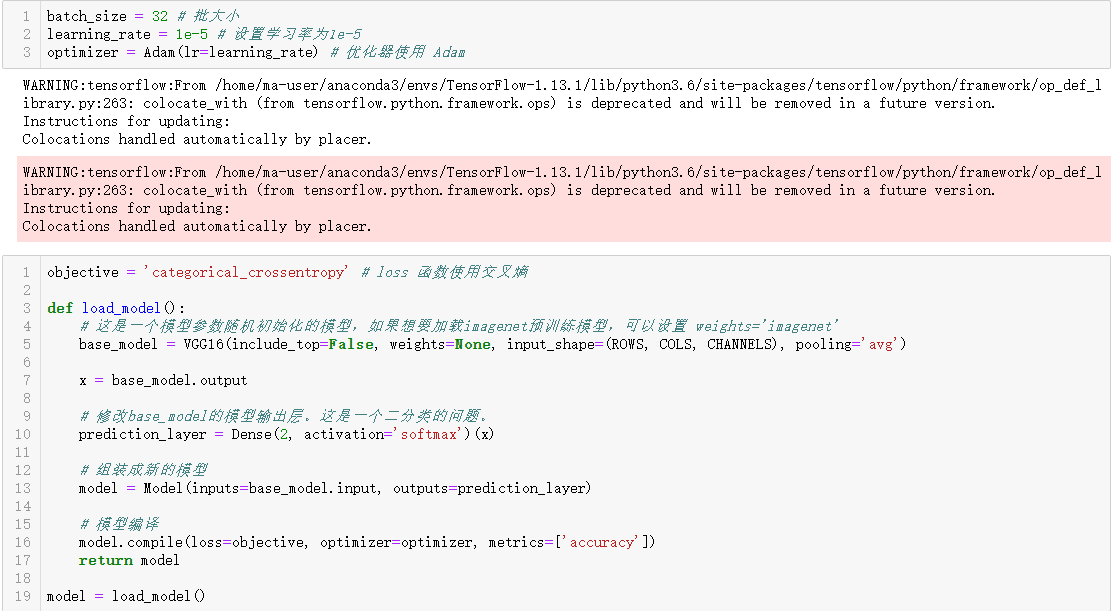


notebook 参数配置

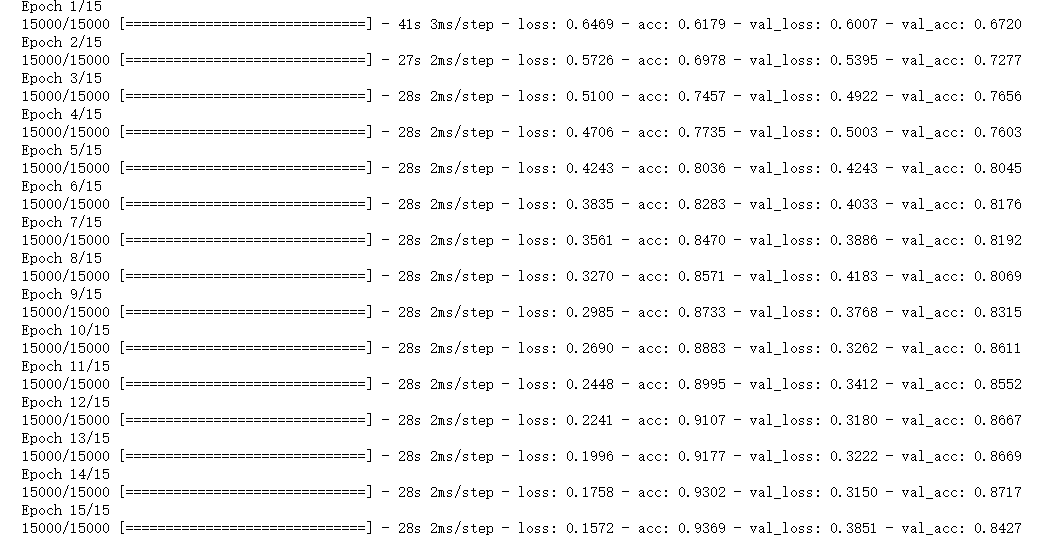
#### 1.2 猫狗大战基础实验



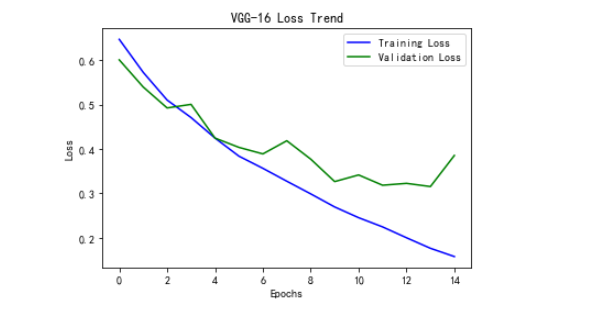
导入依赖库



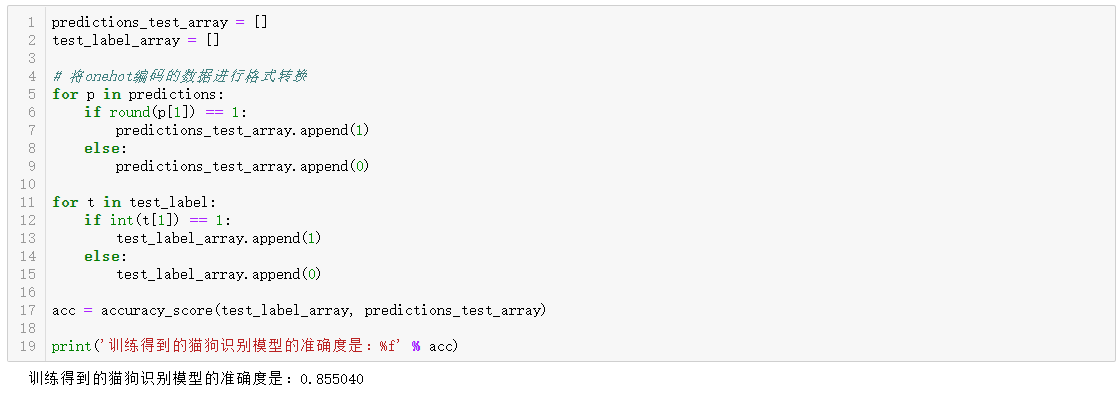
构建神经网络



15轮训练过程



可视化损失函数



模型评估-准确率85.5%

### 2. 调整参数，提高猫狗识别精度

#### 2.1 网络结构

##### 2.1.1 激活函数

选择合适的函数防止梯度消失/爆炸的情况，如果出现这些情况，你的模型得不到更新，这个网络相当于是死的网络，对与训练来说是非常不利的。

##### 2.1.2损失函数

更好的拟合预测与真实的值

##### 2.1.3优化器的选择

也就是梯度下降的方式，尽可能快的去优化目标函数达到全局最小值。提高训练速度，找到最优的解。

##### 2.1.4学习率（learning rate或作lr）

是指在优化算法中更新网络权重的幅度大小。学习率可以是恒定的、逐渐降低的，基于动量的或者是自适应的。不同的优化算法决定不同的学习率。当学习率过大则可能导致模型不收敛，损失loss不断上下震荡；学习率过小则导致模型收敛速度偏慢，需要更长的时间训练。通常lr取值为[0.01,0.001,0.0001]

##### 2.1.5正则化项

防止过拟合的一个参数。

在深度学习里，除了可以在目标函数里加L1,L2的正则化，一般来说，我们在构架网络模型的时候，可以在两层网络之间，加一层dropout层，通过这一层之后，可以让上一层的神经元部分失活，随机让某一部分的神经元进入到下一部分的神经元进行计算，所以我们在构建相对比较深的网络的时候，可以去加一些dropout层。

#### 2.2 训练过程

##### 2.2.1 batch\_size

就是每次我们放多少图片进行训练，在卷积神经网络中，大批次通常可使网络更快收敛，但由于内存资源的限制，批次过大可能会导致内存不够用或程序内核崩溃。bath\_size通常取值为[16,32,64,128]

##### 2.2.2 Epoch

迭代次数是指整个训练集输入到神经网络进行训练的次数，当测试错误率和训练错误率相差较小时，可认为当前迭代次数合适；当测试错误率先变小后变大时则说明迭代次数过大了，需要减小迭代次数，否则容易出现过拟合。

##### 2.2.3 Train\_test\_split

划分训练集和测试集的比例，验证泛化性能。这个比例合适的时候，我们就可以通过训练得到模型，去验证集上去验证它的表现好不好，如果表现好的话，就可以拿到测试集上去测试它的泛化性能，一般训练和测试的比例是7：3，在训练里面还可以分为训练集和验证集，这都属于在训练的过程，验证集只是在训练过程中把它作为训练集的测试，如果整个数据集按10份的话，训练集、验证集、测试集的比例一般是6：2：2，或者是6：1：3，前面两个部分，训练和验证都是为了测试模型的效果好不好，通过在训练集上的训练，我们可以得到这个模型，之后我们拿到验证集上去验证，如果说这个验证的效果还可以，就可以把这个模型放到测试集上去测试，看真实的数据效果怎么样，

#### 2.3 调参实践



##### 2.3.1 修改loss函数和优化器

常见的三种优化器：Momentum, RMSProp, Adam，实验参考中采用Adam，Adam（Adaptive Moment Estimation）算法是将Momentum算法和RMSProp算法结合起来使用的一种算法。

常用的loss函数如下表，由于实验参考中的网络模型原本是用于1000个分类，故使用的是多类别交叉熵损失函数（categorical\_crossentropy），但在这个猫狗大战项目中，最后的目的是二分类，所以采用二分类交叉熵损失函数（binary\_crossentropy），效果应该会更好。



这次调参最大的改动是将Adam优化器调整为RMSProp，采用二分类交叉熵损失函数（binary\_crossentropy）。



准确度提高到90.3%（原85.5%）

##### 2.3.2 修改loss函数

使用理论上效果更好的Adam优化器，采用适合二分类的'binary\_crossentropy'。



准确度提高到92.0%（原85.5%）

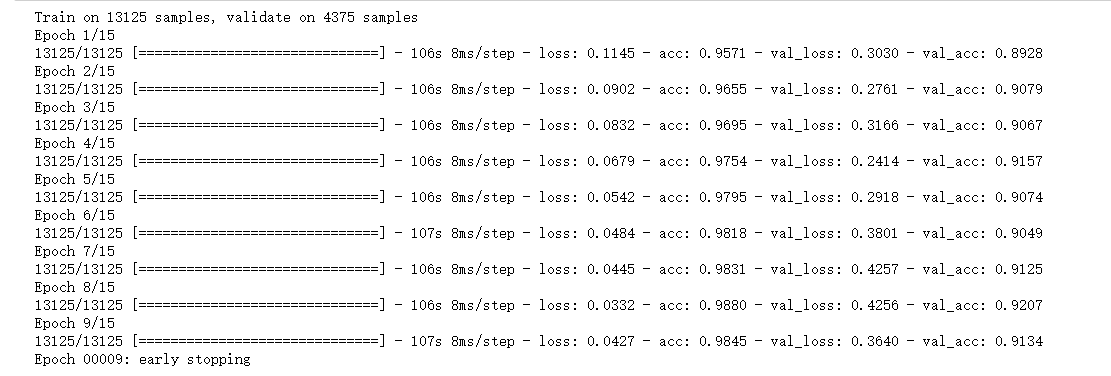
##### 2.3.3 修改图像尺寸，调整验证集比例

将图像尺寸由原来的128\*128输入调整为224\*224输入

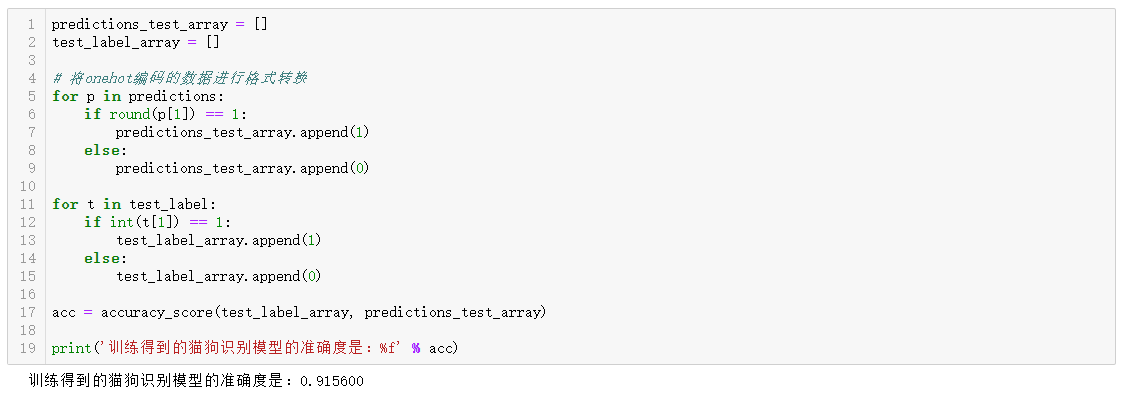
1. # 目标缩放尺寸
2. ROWS = 224
3. COLS = 224
4. CHANNELS = 3

Train\_test\_split将验证集比例调整为30%

1. #调整验证集比例
3. train\_data, test\_data, train\_label, test\_label = train\_test\_split(images, labels, test\_size=0.30, random\_state=10)

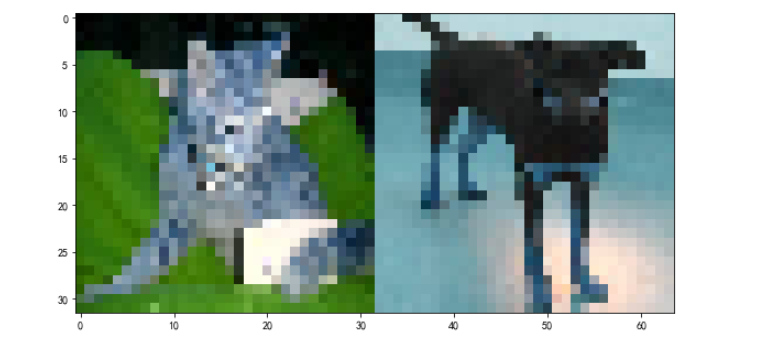


显然，由于输入图像尺寸变大，训练时间变长（一轮106s，原28s）



准确度提高到91.5%（原85.5%）

为了使对比效果更明显，我们减小输入图片尺寸至32\*32，显然图片显示已经模糊不清，特征提取效果可想而知并不好。



图像模糊，特征不明显

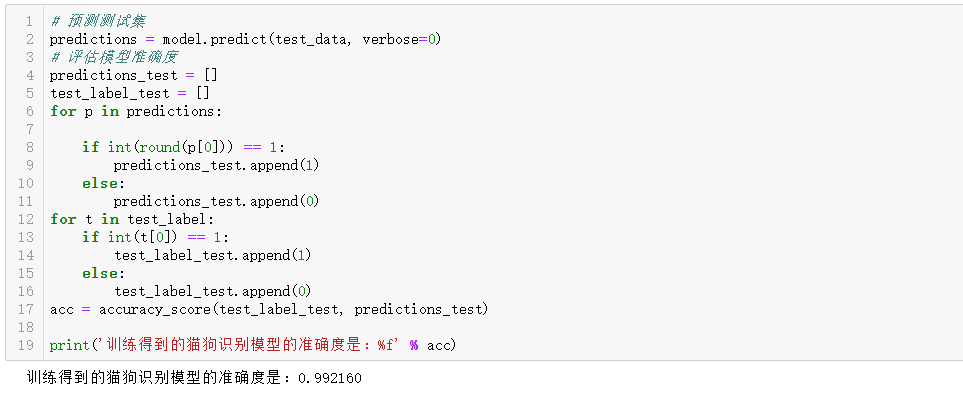


最后的识别率也只有79.3%

##### 2.3.4 直接更换网络模型

将vgg16算法改用xception算法以求提高准确度:

xception算法又叫extreme inception是因为它是inception的进化版,主要改进在于base\_model = Xception(weights="imagenet", include\_top=False, pooling=None, input\_shape=(ROWS, COLS, CHANNELS), classes=2)它将会访问并下载 https://github.com/fchollet/deep-learning-models/releases/download/v0.2/xception\_weights\_tf\_dim\_ordering\_tf\_kernels\_notop.h5 下载速度主要受受本地网速和华为云用户多少影响，至于华为云本身下载速度还是蛮快的。



准确度提高到99.2%！！！（原85.5%）