

（深圳）

实验报告

开课学期： 2022年春季

课程名称： 人工智能

实验名称： 深度学习实现花卉识别

实验性质： 课内实验

实验时间： 2022.4.16 地点： /

学生专业： 计算机科学与技术

学生班级： 1901105

学生学号： 190110509

学生姓名： 王铭

同组成员： 肖力炜、郑羿恺、陈柏江

授课教师： 郑海刚

报告成绩：

# 一、实验内容

本实验是以小组的形式，利用当前流行的三种深度学习框架，对给定的花卉图片数据集进行训练，构造将输入图片分为六种类别的神经网络，并在给定的测试集上进行预测分类。在小组作业中，我主要负责的是基于pytorch框架在本地实现神经网络并完成分类任务，完成pytorch在云端训练和本地预测以及TensorFlow的ModelArts在线部署。

# 算法与模型介绍

本模型采用了四层卷积+池化后连接三层全连接层的神经网络结构，完整的模型结构图如图2.1所示。



图2.1

输入图片为RGB三个通道，长、宽均为128，第一个卷积层所用卷积核为9，步长为2，填充为4，可以根据公式输出维度=计算出输出维度为64\*64，输入通道数为3，输出通道数为64，即用了64个卷积核，则经过第一层卷积后输出维度为(64,64,64)。然后经过BatchNorm层进行规范化，选择ReLU激活函数对结果进行激活。经过第一个池化层，采用最大池化，由步长和填充大小可以算出，池化后的输出维度为(64,32,32)。同理经过第二、三、四层卷积+池化后，维度分别为(128,8,8)、(256,4,4)、(512,1,1)。卷积池化操作后，经过Flatten展平层可以将输入转化为长度为512的一维数据，之后连接3层全连接层便可以进行分类。可以发现，相较于将输入图片直接连接全连接层进行计算，通过卷积池化后，在保存了输入图片的特征的条件下，大大降低了输入数据的维度，从而能有效降低全连接层模型的参数总数。

本模型选择的损失函数为较为常见的交叉熵损失函数CrossEntropyLoss，相较于均方损失函数，采用交叉熵损失函数可以避免损失值较小时梯度较小的问题。

本模型选择的优化器为Adam优化器即采用自适应梯度来更新神经网络的参数，相较于随机梯度下降SGD算法来说，Adam的实现更加简单高效，且能够自动调整学习率，便于训练时更好地收敛，因而选用Adam优化器。

# 三、算法实现

**3.1 实验环境**

操作系统：Windows10，CUDA版本：10.2

编程语言：采用Python3.7+pytorch1.9.0，训练过程在GPU上完成

**3.2 数据集处理**

数据读取部分，采用openCV，从训练集中读入的图片，并将读入的图片统一缩放为128\*128，按照读入图片的种类顺序将标签与类别序号对应，并将图片和标签转换成numpy的ndarray，其中图片采用浮点数据类型，标签采用整数型。代码实现如图3.1所示：

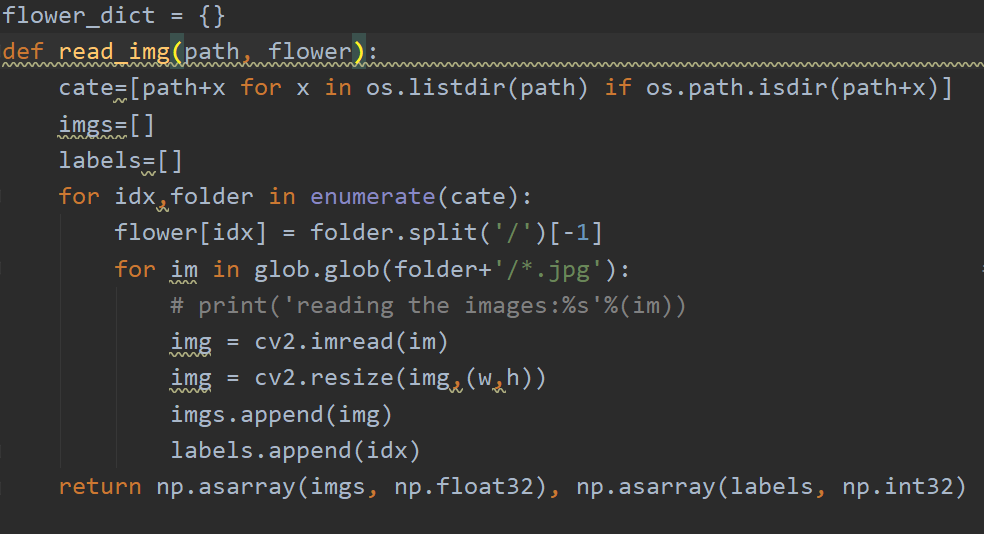


图3.1

按照常用的8：2的比例，将读入的训练图片划分成训练集和验证集，并将图片进行标准化处理，由于输入图片的维度为128\*128\*3，为使用pytorch，需要将其转变为3\*128\*128，同时将数据转换成torch的类型，实现代码如图3.2所示：

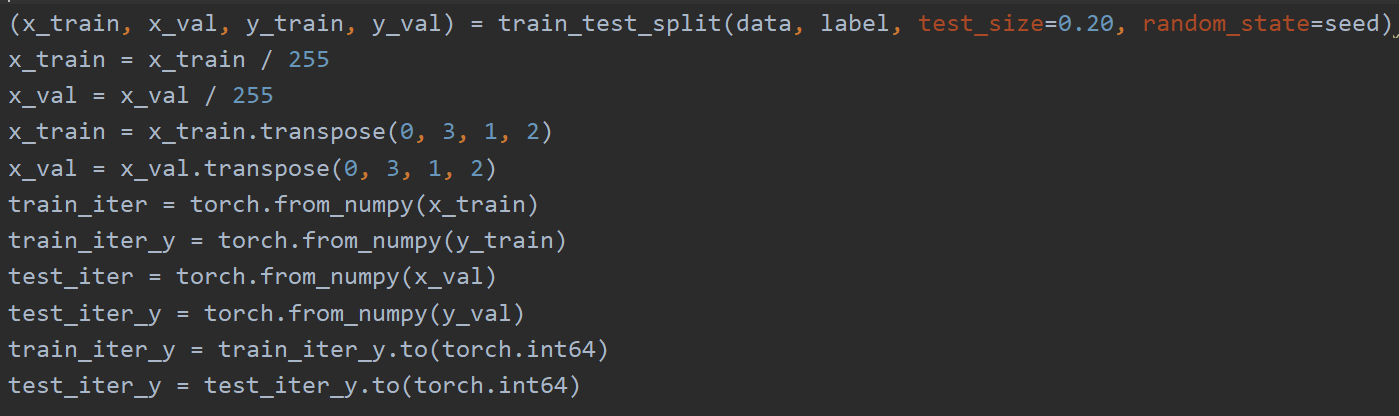


图3.2

**3.3 模型构建与训练**

模型定义代码如图3.3所示：



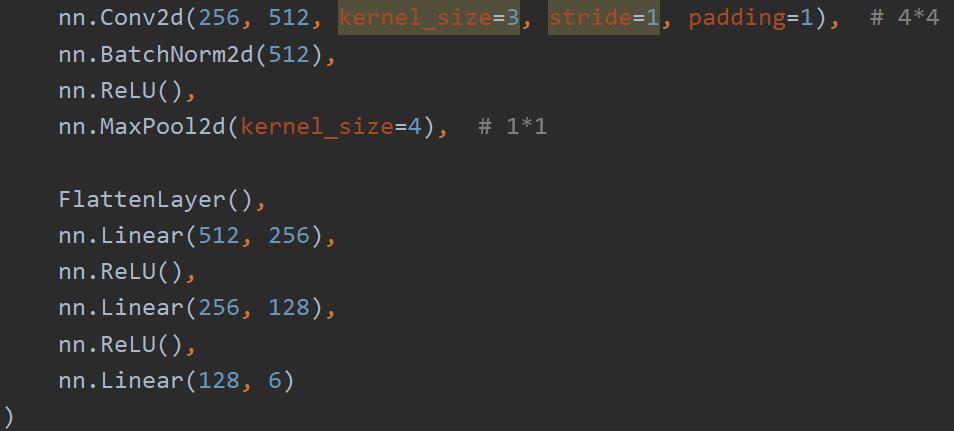


图3.3

学习率lr=0.0001，训练次数nums\_epoch=16以及Adam优化器的设置代码如图3.4所示：

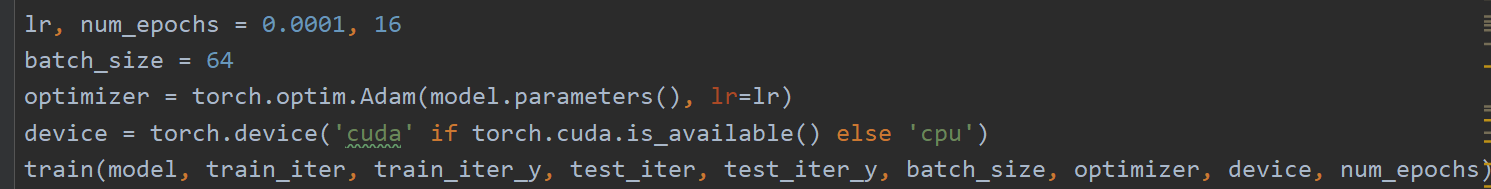
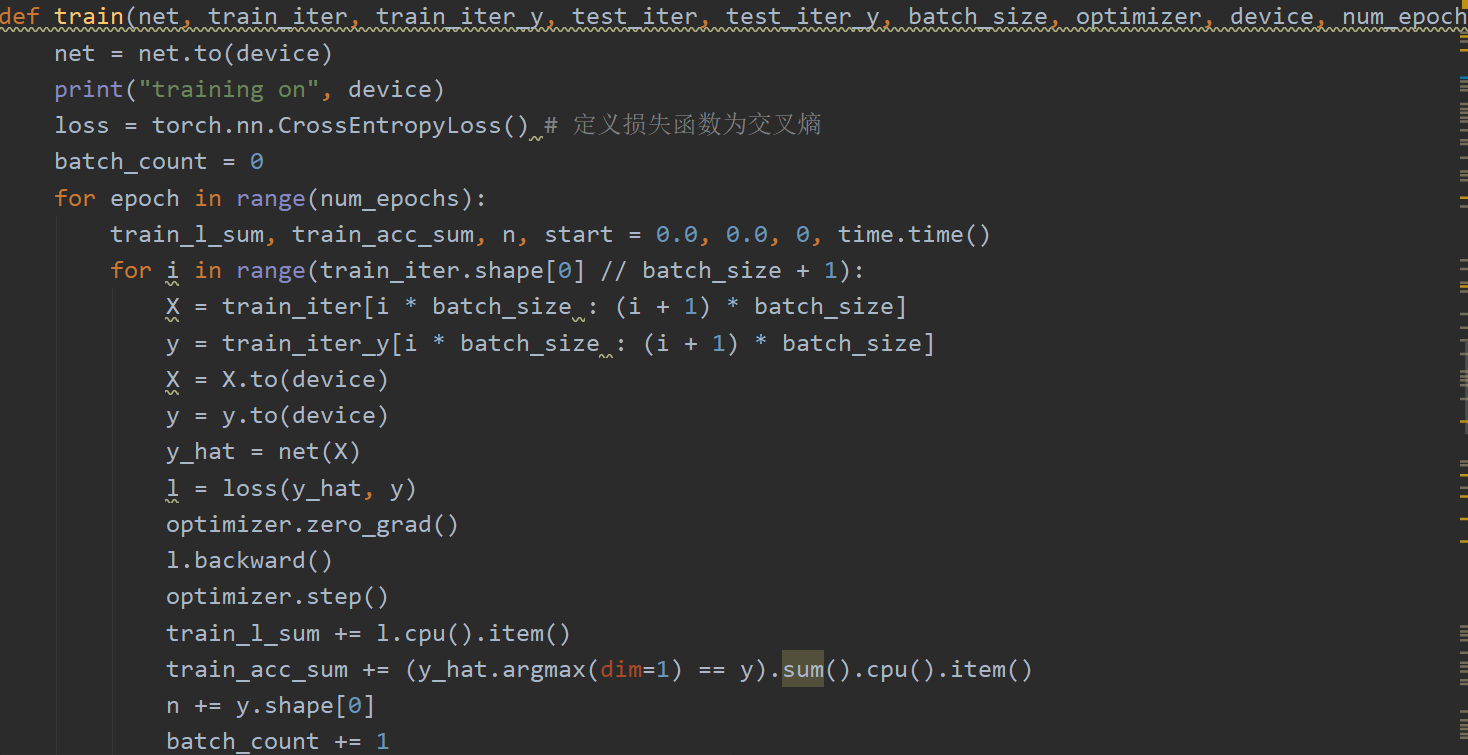


图3.4

采用pytorch框架的backword和step可以完成参数的训练更新，采用的交叉熵损失函数以及训练过程的具体实现代码如图3.5所示：



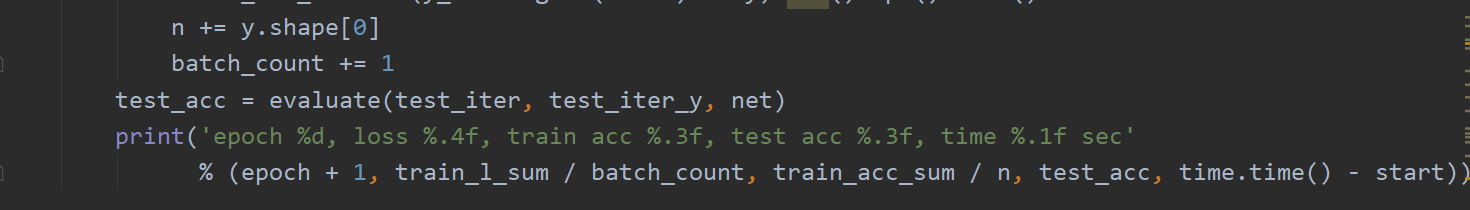


图3.5

**3.4 实验结果**

训练采用分类成功精确度指标判断模型好坏，训练过程的输出如图3.6所示：

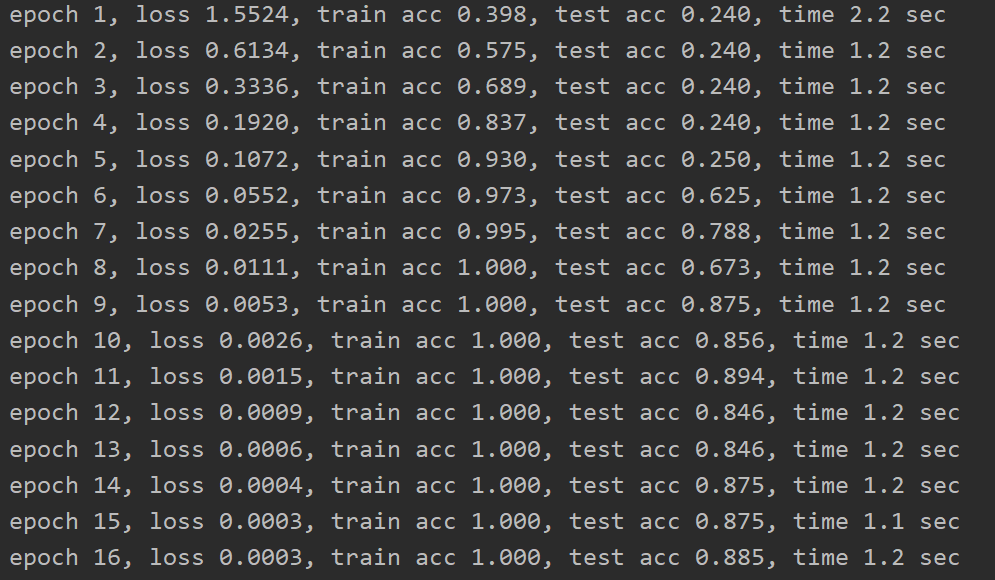


图3.6

可以将训练过程的损失值loss，训练集的精确度train\_acc以及验证集上的精确度test\_acc画图表示：

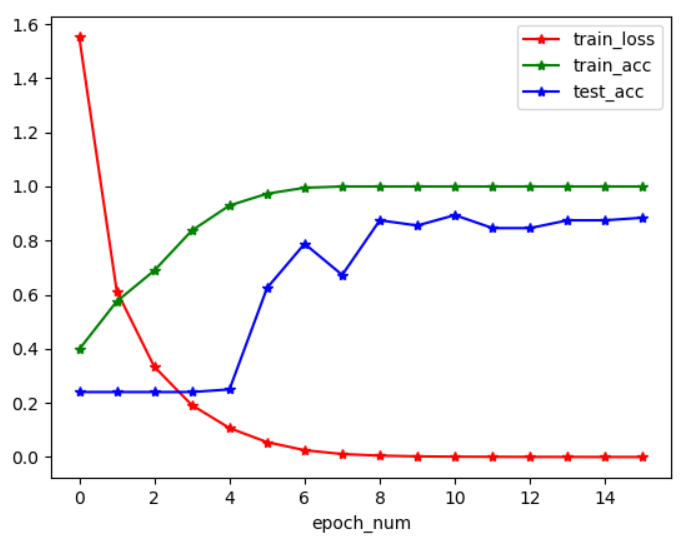


图3.7

由图3.7可知，随着训练轮次增加，训练的损失值逐渐降低，并最终趋于0.验证集和测试集的准确度均逐渐收敛，在训练集上的准确率趋近于1，测试集上的准确率可以达到88.5%，由于输入数据集较小，故收敛速度较快，在训练了10轮之后逐渐收敛。

在本地进行预测结果如图3.8所示：

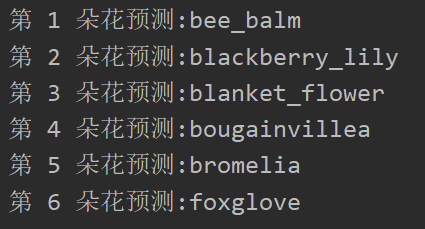


图3.8

对TestImage中的6张图片进行预测的结果正确率达到100%，因而可以认为该模型能够较好的完成本实验的花卉识别分类任务。

在modelarts训练、预测结果如图3.9，图3.10所示，

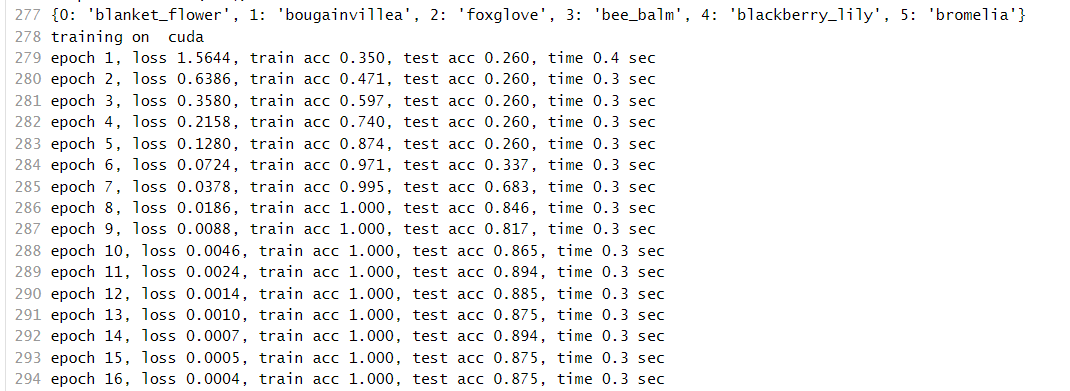


图3.9

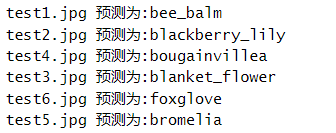


图3.10

在云端训练的结果均正确，可以将训练后保存的模型下载到本地训练，结果如图3.11所示，由于云端是乱序读入，故根据图3.9打印的映射关系可知，预测正确。



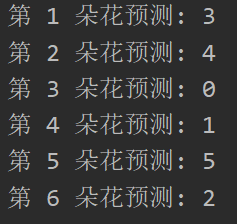
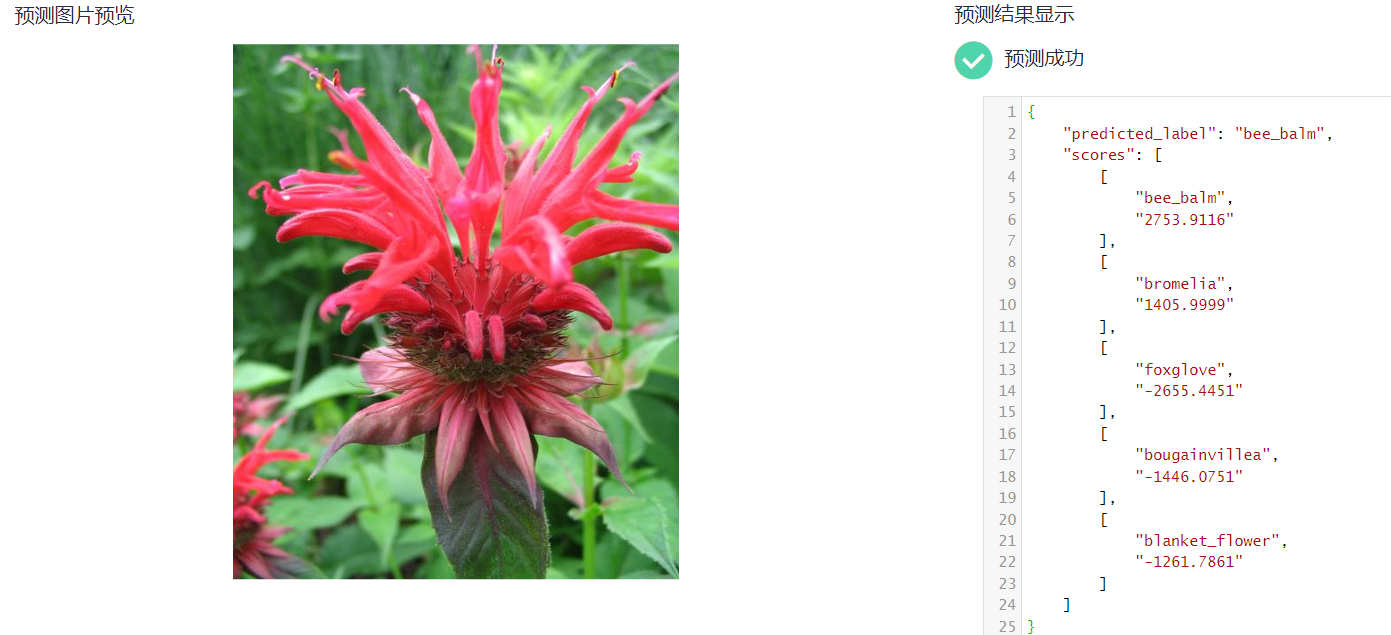
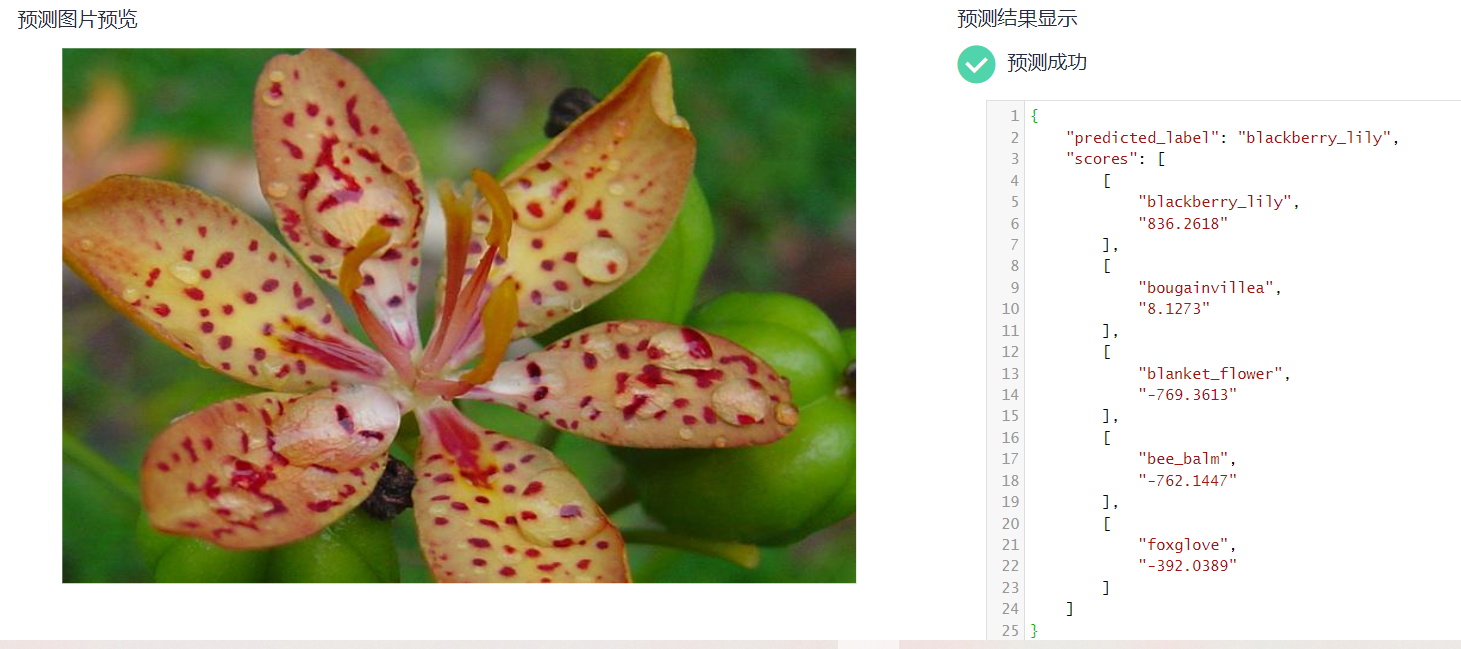
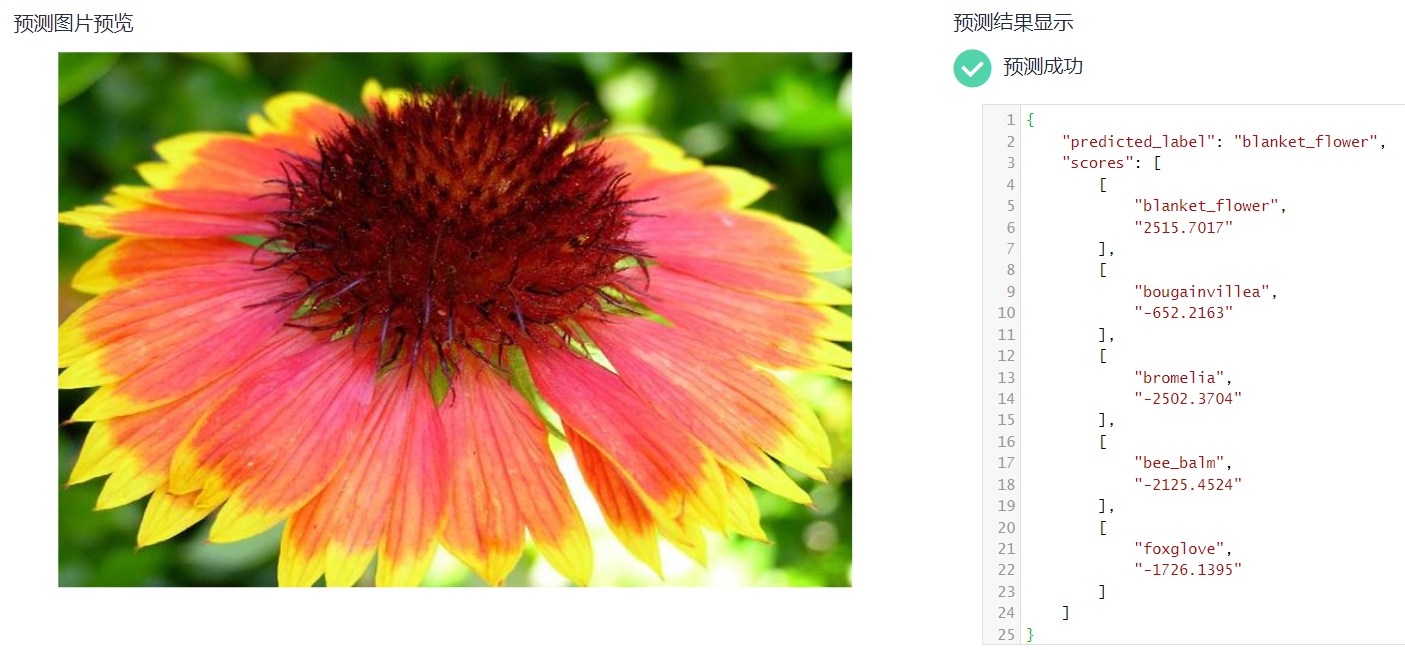


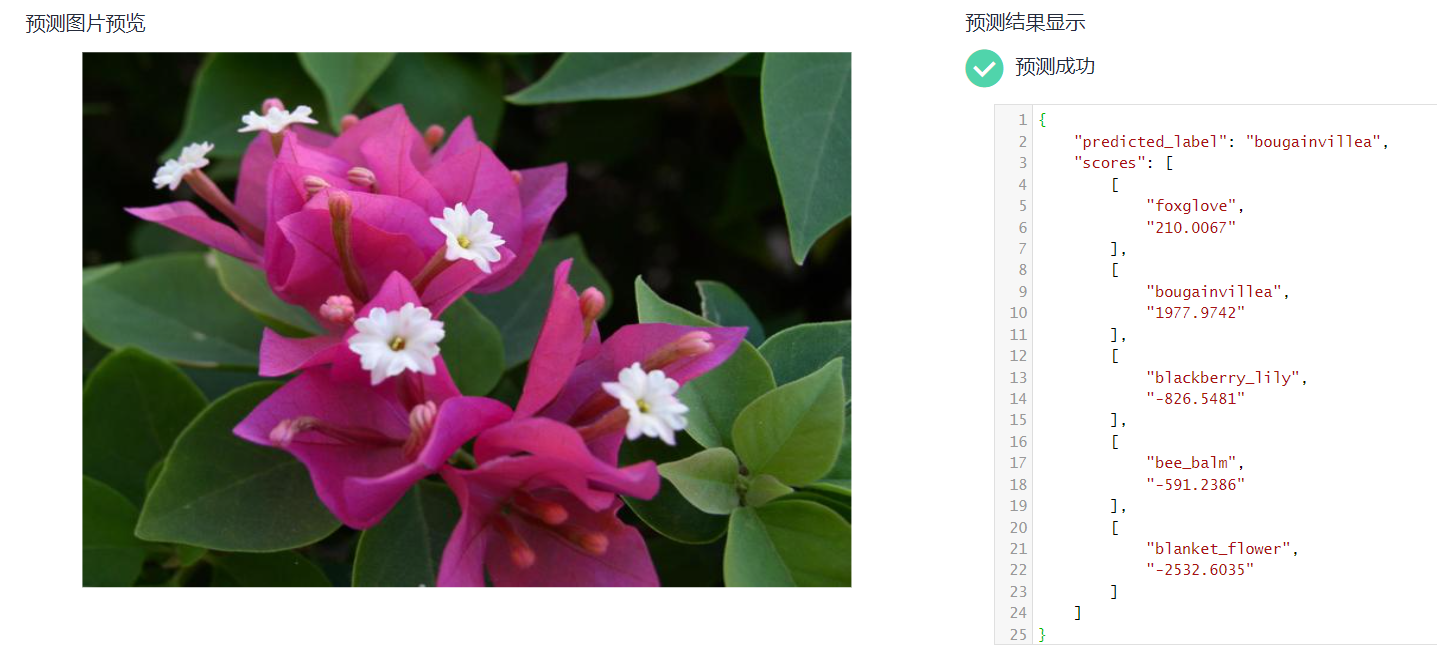
图3.11

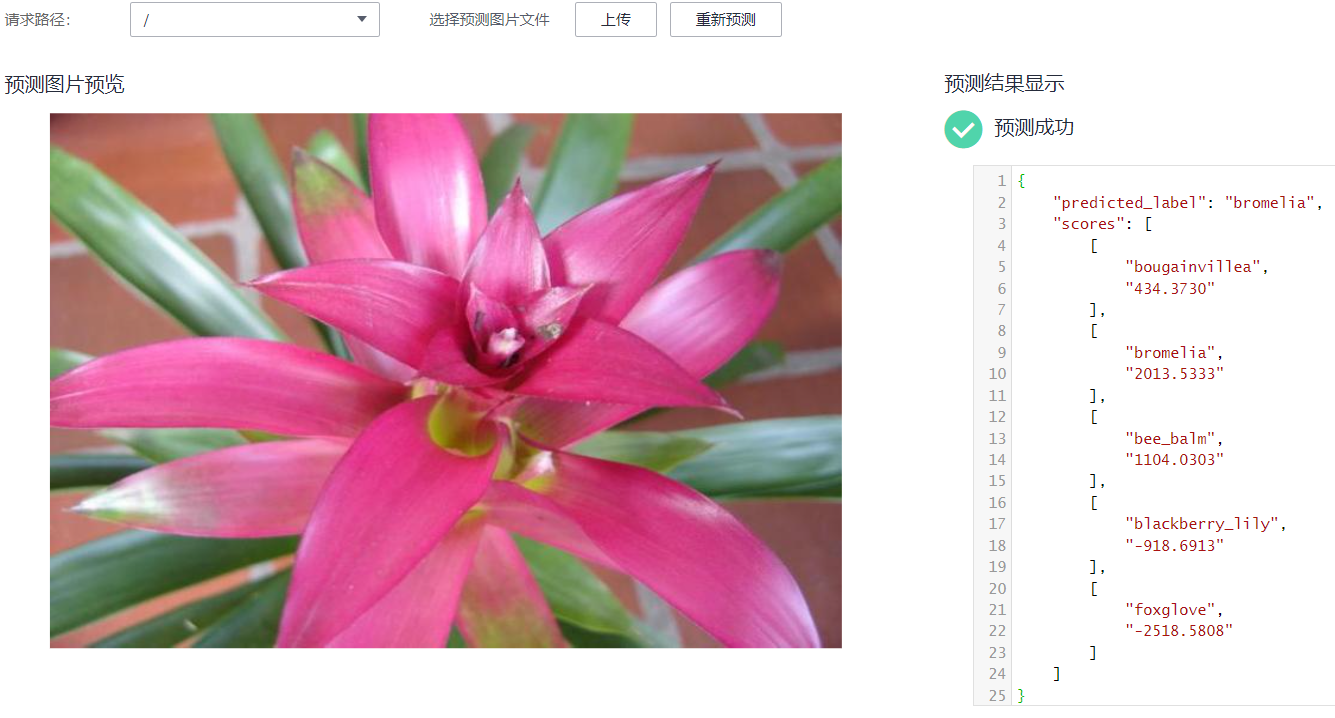
在ModelArts完成在线部署，最终的预测结果如图所示：

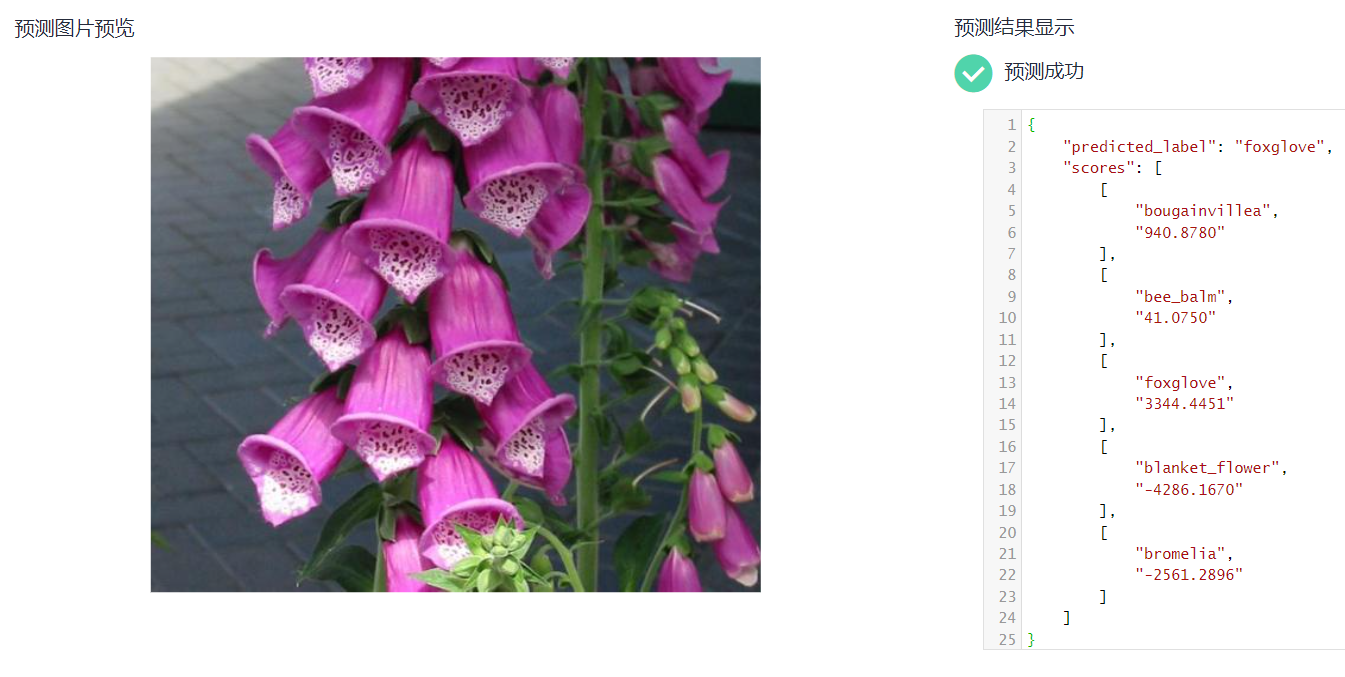












由图知，预测结果均正确。

**3.5 深度学习框架总结**

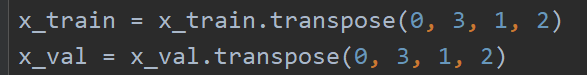
在完成实验过程中，相同模型的结构、超参、损失函数以及优化器在不同框架下训练的效果不同，将我的pytorch网络模型在tensorflow、mindspore中实现时，效果均不理想，可能与损失函数或优化器的具体实现上的不同有关。相同的pytorch模型可以通过modelarts，在云端训练好并保存模型的参数，再下载到本地进行训练，在训练过程中，显然云端的GPU性能更优，花费的时间更少，在进行一些大规模数据训练时采用云端训练模型会更节省时间。

# 四、总结

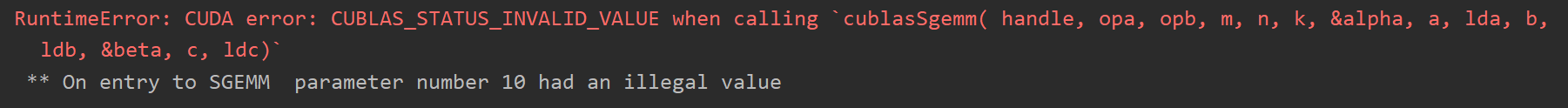
**4.1 碰到的问题及解决方法**

（1）问题：输入数据的维度与定义的模型不匹配

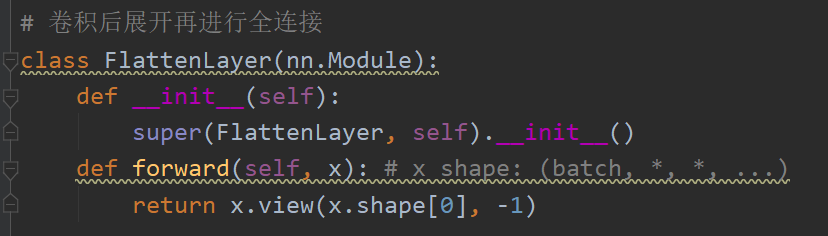
解决方法：查阅资料知，读入图片维度最后一维为通道数，而模型输入时，需要将通道维度置于第一维（batch维之后），利用transpose可以实现。



（2）卷积后连接全连接层前未进行展平操作，报错如下：



解决方法：在卷积池化操作完成之后，先添加展平层FlattenLayer将数据展平再与全连接层进行对接。

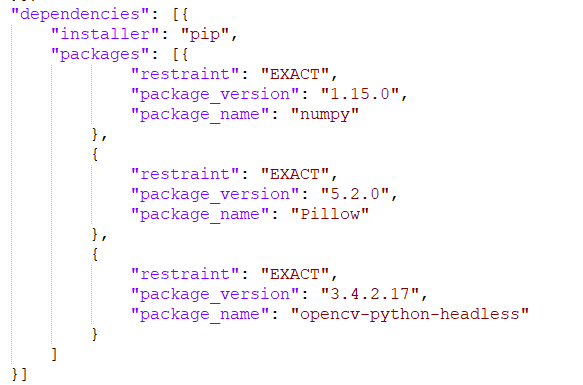


（3）在云端在线部署时，由于本版本的pytorch实现所用是opencv读入图片并进行训练，在推理代码的preproce中用PIL读取导致预测效果极差，但改用opencv时，出现错误。

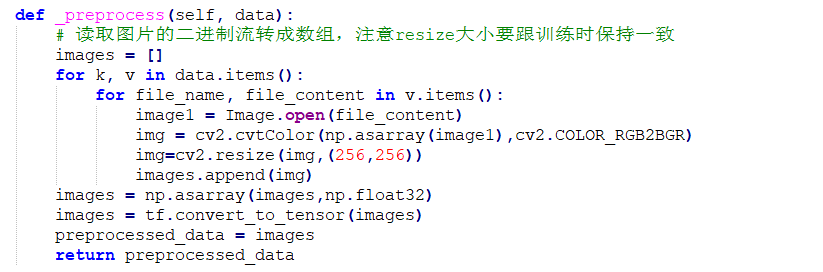


解决方法：

修改config.json文件pip安装opencv库，如图所示：



在预处理时，先用PIL处理图片，再转换为opencv的格式即可，代码如图所示：



**4.2 实验的启发、总结、建议**

通过本次小组实验，我了解了当前常用的三种深度学习框架的基本使用以及他们之间的区别，加深了我对理论学习的认识同时，为后续可能用到深度学习的学习打下了基础，同时加强了小组合作意识。

建议：本次实验中所用框架的环境搭建起来较为麻烦，且对于基础较为薄弱的同学上手较难，实验过程更偏向于调参，希望后续可以改进实验环境的配置，降低上手难度，扩展实验内容。